Lecture Notes in Computer Science 5666

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Paul S. Andrews Jon Timmis
Nick D.L. Owens Uwe Aickelin Emma Hart
Andrew Hone Andy M. Tyrrell (Eds.)

Artificial Immune Systems

8th International Conference, ICARIS 2009
York, UK, August 9-12, 2009
Proceedings

Springer

Volume Editors

Paul S. Andrews
University of York, Heslington, York, YO10 5DD, UK
E-mail: psa@cs.york.ac.uk

Jon Timmis
University of York, Heslington, York, YO10 5DD, UK
E-mail: jtimmis@cs.york.ac.uk

Nick D.L. Owens
University of York, Heslington, York, YO10 5DD, UK
E-mail: ndlo100@ohm.york.ac.uk

Uwe Aickelin
University of Nottingham, Nottingham, NG8 1BB, UK
E-mail: uxa@cs.nott.ac.uk

Emma Hart
Napier University, Edinburgh, EH10 5DT, UK
E-mail: e.hart@napier.ac.uk

Andrew Hone
University of Kent, Canterbury, CT2 7NF, UK
E-mail: A.N.W.Hone@kent.ac.uk

Andy M. Tyrrell
University of York, Heslington, York, YO10 5DD, UK
E-mail: amt@ohm.york.ac.uk

Library of Congress Control Number: 2009930954

CR Subject Classification (1998): F.1, I.2, F.2, H.2.8, H.3, J.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-03245-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-03245-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12722181 06/3180 5 4 3 2 1 0

Preface

Artificial immune systems (AIS) is a diverse and maturing area of research that bridges the disciplines of immunology and engineering. The scope of AIS ranges from immune-inspired algorithms and engineering solutions in software and hardware, to the understanding of immunology through modeling and simulation of immune system concepts. AIS algorithms have been applied to a wide variety of applications, including computer security, fault tolerance, data mining and optimization. In addition, theoretical aspects of artificial and real immune systems have been the subject of mathematical and computational models and simulations.

The 8th International Conference on AIS (ICARIS 2009) built on the success of previous years, providing a forum for a diverse group of AIS researchers to present and discuss their latest results and advances. After two years outside Europe, ICARIS 2009 returned to England, the venue for the first ICARIS back in 2002. This year's conference was located in the historic city of York, and was held in St. William's College, the conference venue of York Minster, northern Europe's largest Gothic cathedral.

Continuing the scope of previous ICARIS conferences, ICARIS 2009 was themed into three diverse areas: immune system modeling, theoretical aspects of AIS, and applied AIS. ICARIS this year saw the addition of published extended abstract submissions for the immune modeling stream, alongside full papers. Extended abstracts underwent the same rigorous review process, being checked for quality and relevance. In addition, we introduced a rebuttal system that allowed authors to respond directly to reviewers' comments. Based on the rebuttals, we were able to conditionally accept a number of papers that were revised and checked before full acceptance, resulting in an increased quality of these papers. From 55 submissions, we were pleased to accepted 30 high-quality full-length papers and extended immune modeling abstracts for publication, giving us an acceptance rate of 55%.

ICARIS 2009 was delighted to play host to two fascinating keynote speakers. David Harel from the Weizmann Institute of Science, Israel, presented ways in which techniques from computer science and software engineering can be applied beneficially to research in the life sciences, such as T cell development in the thymus and lymph node behavior. Dario Floreano from the School of Engineering at the Swiss Federal Institute of Technology in Lausanne, Switzerland, presented an alternative approach to the design of control systems for micro and unmanned aerial vehicles that are heavily inspired by insect vision and flight control.

To supplement the technical papers and keynotes, three tutorials were presented by Susan Stepney, Thomas Stibor and Tim Hoverd. Stepney demonstrated the usefulness of statistics for AIS algorithms, Stibor described how AIS can benefit from techniques used in the field of machine learning, and

Hoverd gave us an overview of how modeling techniques like the Unified Modeling Language should, and should not, be applied to AIS and related techniques. In addition, ICARIS 2009 played host to a DSTL-sponsored workshop on AIS for anomaly detection in real-time spectra, organized by Mark Neal of Aberystwyth University. The workshop included a competition requiring participants to perform anomaly detection on real-time mass-spectrometry data.

We would like to thank the keynote and tutorial speaks, Program Committee, ICARIS Vice and Publicity Chairs, Mark Neal and finally the authors for their input into creating such a high-quality conference. We would also like to thank Bob French and Mandy Kenyon from the Research Support Office in the Department of Computer Science, University of York, for their invaluable assistance behind the scenes, helping to make ICARIS 2009 a great success.

May 2009

Paul Andrews
Jon Timmis

Organization

Organizing Committee

General Chair	Jon Timmis (University of York, UK)
General Chair	Paul Andrews (University of York, UK)
Vice Chair	Emma Hart (Napier University, UK)
Vice Chair	Andy Tyrrell (University of York, UK)
Vice Chair	Andy Hone (University of Kent, UK)
Vice Chair	Uwe Aickelin (University of Nottingham, UK)
Publicity	Nick Owens (University of York, UK)

Program Committee

Uwe Aickelin
Paul Andrews
Iain Bate
Peter Bentley
Hugo van den Berg
Hugues Bersini
George Bezerra
Josh Carlin
Leandro de Castro
Ed Clark
Carlos Coello Coello
Vincenzo Cutello
Dipankar Dasgupta
Andries Engelbrecht
Stephanie Forrest
Juan Carlos Galeano-Huertas
Maoguo Gong
Fabio Gonzalez
Julie Greensmith
Emma Hart
Andy Hone
Christian Jacob
Colin Johnson
Henry Lau
Doheon Lee
Sang Wan Lee
Wenjian Luo
Chris McEwan
Thiago Masutti
Carmen Molina-Paris
Nikolaos Nanas
Giuseppe Nicosia
Luis Fernando Nino
Fabricio Olivetti
Nick Owens
Rodrigo Pasti
Mario Pavone
Fiona Polack
Mark Read
Peter Ross
Siti Mariyam Shamsuddin
Susan Stepney
Thomas Stibor
Alexander Tarakanov
Jonathan Timmis
Andy Tyrrell
Fernando Von Zuben
Andrew Watkins
Slawomir Wierzchon

ICARIS Steering Committee

Jon Timmis	University of York, UK
Emma Hart	Napier University, UK
Leando de Castro	McKenzie University, Brazil
Hugues Bersini	ULB, Belgium
Stephanie Forrest	University of New Mexico, USA
Christian Jacob	University of Calgary, Canada
Guiussepe Nicosia	University of Catania, Italy
Mark Neal	Aberystwyth University, UK
Peter Bentley	UCL, UK
Doheon Lee	KAIST, Korea

Keynote Speakers

David Harel	Weizmann Institute of Science, Israel
Dario Floreano	Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland

Tutorial Speakers

Susan Stepney	University of York, UK
Thomas Stibor	Technische Universität München, Germany
Tim Hoverd	University of York, UK

Sponsors

Intelligent Systems Group, Department of Electronics, University of York
Non-Standard Computation Group, Department of Computer Science, University of York
AISWeb: The on-line home of artificial immune systems

Table of Contents

Immune System Modeling

Theoretical Aspects of Artificial Immune Systems

Applied Artificial Immune Systems

Using UML to Model EAE and Its Regulatory Network

Mark Read[1], Jon Timmis[1,2], Paul S. Andrews[1], and Vipin Kumar[3]

[1] Department of Computer Science, University of York, UK
{markread,jtimmis,psa}@cs.york.ac.uk
[2] Department of Electronics, University of York, UK
[3] Laboratory of Autoimmunity, Torrey Pines Institute for Molecular Studies

Experimental Autoimmune Encephalomyelitis (EAE) is an autoimmune disease in mice which serves as a model for multiple sclerosis in humans [4,5]. The disease constitutes the direction of immunity towards myelin, an insulatory material that covers neurons. The consequential damage to the central nervous system (CNS) can lead to paralysis and death [6].

EAE can be spontaneously induced by immunisation with myelin basic protein (MBP, a myelin derivative) and complete Freund's adjuvant. The immunisation prompts the expression of MBP peptides on MHC molecules by antigen presenting cells (APCs), and the consequent activation of MBP-reactive T cells. The activated T cells migrate to the CNS parenchyma where their secretion of type 1 cytokines promotes the destruction of myelin.

A network of immune cell interactions operates to counter EAE. This regulatory network consists of $CD4^+$ and $CD8^+$ regulatory T cells (Tregs). The natural lifecycle of MBP-reactive CD4Th1 cells leads to their physiological apoptosis and subsequent phagocytosis by APCs. The peptides derived from CD4Th1 cells, when presented on MHC, prompt the activation of $CD4^+$ and $CD8^+$ Tregs. The CD8Tregs, with prior help from CD4Tregs, can induce the apoptosis of activated MBP-reactive CD4Th1 cells. The resulting population reduction permits the expansion of CD4Th2 cells which do not promote debilitating destruction of the CNS.

Our long term intention is to construct models and simulations of EAE and its regulatory network for the purposes of performing *in silico* experimentation. It is essential that a coherent understanding of the biological domain is obtained to construct an accurate representation of the system [7]. In line with others, for example [2,1,3], we have selected the UML as the tool with which to develop our models, before we move to an agent based simulation. We have completed a first pass in modelling the biological domain and wish to highlight certain issues that we have found when employing the UML in this context. We stress that this first pass of modelling serves only as a concise detail of our understanding of the biological domain, it is not a technical specification of a simulator.

We have found that the construction of our models has raised various questions of the biological domain; the immunological literature typically reports what *does* happen in a certain experimental setup for a particular event to manifest, it does not indicate all possibilities of what *can* happen under altered conditions.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 4–6, 2009.

To correctly model and simulate the system we must appreciate the latter as well as the former.

Capturing system wide behaviours with activity diagrams
As an appropriate starting point we suggest modelling the high level behaviours we intend our simulation to capture, from the interactions (at an abstract level) of the low-level components. Activity diagrams have proven to be a satisfactory technique with which to accomplish this. Any abstract concept can be expressed as an activity, and links between activities can span across multiple system entities; in our case cells. Furthermore, activity diagram semantics allow for the expression of concurrent activities; concurrency is a fundamental intrinsic quality of biological systems.

Represting static relationships with class diagrams
We have found the construction of class diagrams to be effective at generating questions relating to the quantities of entities that may partake in an activity at a particular time. These are valid questions, because they pertain to the dynamics of the system. However, reasoning about the behaviour of the system in a static manner is not as informative as it is from a dynamic viewpoint. *In vivo* the number of entities that can attempt to simultaneously interact with one another varies considerably. This usually manifests in '0..*' cardinalities on class diagrams, which are not particularly informative. Furthermore, biology is rich with entities that interact and influence large numbers of other entities, which leads to highly connected class diagrams that are difficult to interpret in a meaningful manner.

Sequence diagrams can be misleading
We have not used sequence diagrams in this stage of modelling as we consider them to a bad fit to our modelling needs; thinking about this biological domain in terms of entities that wait on other entities to complete some task is inappropriate. For example, a cell may require a series of signals to reach some state, but it does not lie in wait in between receipt of signals; it will continue to interact with its environment. Cells can be open to more than one path of events. The syntax of sequence diagrams does not communicate this well, and rather implies that an entity be temporarily 'locked' or suspended whilst activity proceeds elsewhere. A cell does not hold responsibility over sub-actions that result from its own.

Low level dynamics and state machine diagrams
State machine diagrams of individual system elements that depict low level dynamics have been very informative. Their provision of facilities to express orthogonality, concurrency, mutual exclusion, and containment of states renders them appropriate for expressing behaviour of cells. Though most behaviours in the system can be extrapolated through examination of state machine diagrams, higher level system dynamics that rely on interactions between several system components are difficult to comprehend through examination of state machine diagrams alone. This presents another use for activity diagrams; they tie low level dynamics of individual entities together into system wide behaviours.

Depicting feedback with the UML
There are aspects of the biological system that we have not been able to satisfactorily express using the UML. The biological system of interest is heavily governed by the interactions of feedback mechanisms. Activity diagrams can demonstrate the order in which critical interactions and events must take place for a high level behaviour to manifest, however they incorrectly imply that one activity stops and another starts. In reality the entity responsible for a preceding activity does not hand off control to that which follows, it continues and can potentially perfrom the same activity again. This concurrency amongst system elements can result in feedback, where an increasing number of elements engage in some activity. Relative population dynamics play a significant role in this biological system (for example the interplay between CD4Th1 and CD4Th2 cells) and it is important to communicate this information in the model. Owing to their ability to express any abstract concept across any number of system elements, we believe that the modification of activity diagram syntax and semantics can yield an appropriate medium for the expression of feedback. This forms ongoing research.

In conclusion
We have found UML to be a reasonably expressive medium in which to represent this biological system, however that are aspects of the system for which this is not the case. Future work entails the development of a simulation of the biological system with which we intend to integrate known biological data and perform in silico experimentation that can inform wet-lab experimentation.

References

1. Bersini, H.: Immune system modeling: The OO way. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 150–163. Springer, Heidelberg (2006)
2. Harel, D.: Statecharts: A visual formalism for complex systems. Science of Computer Programming 8(3), 231–274 (1987)
3. Johnson, C.G., Goldman, J.P., Gullick, W.J.: Simulating complex intracellular processes using object-oriented computational modelling. Progress in Biophysics & Molecular Biology 86(3), 379–406 (2004)
4. Kumar, V.: Homeostatic control of immunity by TCR peptide-specific Tregs. The Journal of Clinical Investigation 114(9), 1222–1226 (2004)
5. Kumar, V., Sercarz, E.: An integrative model of regulation centered on recognition of TCR peptide/MHC complexes. Immunological Reviews 182, 113–121 (2001)
6. Madakamutil, L.T., Maricic, I., Sercarz, E.E., Kumar, V.: Immunodominance in the TCR repertoire of a TCR peptide-specific CD4+ Treg population that controls experimental autoimmune encephalomyelitis. The Journal of Immunology 180(1), 4577–4585 (2008)
7. Polack, F.A.C., Andrews, P.S., Sampson, A.T.: The engineering of concurrent simulations of complex systems. To appear in CEC 2009 (2009)

Non-deterministic Explanation of Immune Responses: A Computer Model

Anastasio Salazar-Bañuelos

Hotchkiss Brain Institute
Department of Surgery, Division of Transplantation
University of Calgary, AB, Canada
salazara@ucalgary.ca

1 Introduction

Classically, immune responses are considered to be antigen-driven, implying a direct or indirect pattern recognition. This deterministic, instructionalist view of the immune system relies on molecular recognition; hence, the occurrence of immune responses depends exclusively on the differentiation between "Self" and "Non-Self" molecules[1,2]. A different approach is to consider immune responses as the emergent phenomena of a complex dynamic system as it is understood in System Biology[3]. Here, I present a computer simulation in NetLogo4.0.3[4] based on a theory that considers the emergence of inflammation as the defining phenomena of an immune response. Whereas the theory recognizes that microscopic events such as molecular recognition are part of the dynamics of the system, it maintains that immune phenomena cannot be understood from the study of individual events alone. The theoretical basis for this work is described elsewhere [5].

2 The Simulation

A two-dimensional space is formed by discrete units representing cells in a tissue. The space is divided into a central restricted area, which simulates the central lymphatic organs (principally the bone marrow), and remaining space, which simulates peripheral non-lymphatic tissue. The cells (patches) produce, dissipate, and diffuse *chemicals*, which simulate the production of mediators or lymphokines by neighbouring cells. Independent agents are created by simulating cells from the lymphatic system. These agents are divided into *stem cells* and *peripheral lymphocytes*, both of which are composed of two *sub-clones*: 1) an auto-reactive (+) *sub-clone* that interacts with the *cells* by fractionally increasing the *chemicals* in the patch where the *cells* are located and 2) a suppressive (-) *sub-clone* that decreases these *chemicals. Stem cells* are generated by recursion [6], starting from the creation of (+) and (-) *sub-clones* at a ratio of 3:1, which favors the predominance of (+) *sub-clones*. All *cells* and interactions are specific for one antigen, and all simulations take place from these initial conditions.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 7–10, 2009.

2.1 Clonal Expansion

This step simulates the origin of the lymphatic system from a few "mother cells" in early ontogeny, creating a colony of cells allocated to the central lymphatic system, mainly the *bone marrow*, from which all other lymphatic cells will be produced over the life of the individual (Fig. 1). Because the recursive stochastic process that generates this colony forms both (+) and (-) *sub-clones*, the proportion of *sub-clones* will fluctuate with decreasing amplitude as the number of agents increases, reaching stability in direct proportion to the numbers of agents created. This colony does not migrate and does not interact with cells in the *peripheral system*, resembling the population of progenitor cells in the bone marrow. The functions described below can be simulated by the program.

2.2 Clonal Selection

The initial condition was designed to favor a predominance of (+) *sub-clones* over (-) *sub-clones*, as seen when clonal selection does not take place. Clonal selection reverses this situation by eliminating a proportion of the (+) *sub-clones* while clonal expansion is taking place, thus simulating the elimination of auto-reactive clones in the thymus[7].

2.3 Proliferation

The *stem cells* in the *bone marrow* do not migrate to the periphery, but are the precursors of *peripheral lymphocytes*, which are identical to *stem cells* in all aspects except that they do migrate to the *periphery* and move randomly, interacting with the *cells* (patches) that they contact in their migration and decreasing or increasing the production of chemicals accordingly. The production of *peripheral lymphocytes* also is done by recursion; therefore the proportion of (+) to (-) *sub-clones* in the *peripheral lymphocytes* will mirror that proportion in the *stem cells*, only at higher numbers. The probability of discrepancy between the proportion of *sub-clones* in the *bone marrow* and the *periphery* will increase as the number of *stem cells* decreases.

2.4 Lymphatic-Ablation

This function eliminates all *peripheral lymphocytes*, leaving the *stem cell* colony intact in the *bone marrow*. This is used to illustrate that the system is robust, even for events affecting the entire *peripheral lymphocyte* population, since the recursive production of *peripheral lymphocytes* in the *bone marrow* will repopulate the *peripheral system*, maintaining *sub-clone* proportions similar to that of their progenitors. This highlights the fact that the system will be robust in direct proportion to the number of *stem cells* in the *bone marrow*.

2.5 Danger

This function produces an instant increase in the *chemicals* released by the *cells* of an specific and defined area in the *periphery*. This simulates the effect of an acute injury that produces a focus of inflammation and shows how the system behaves according to the several situations that can take place.

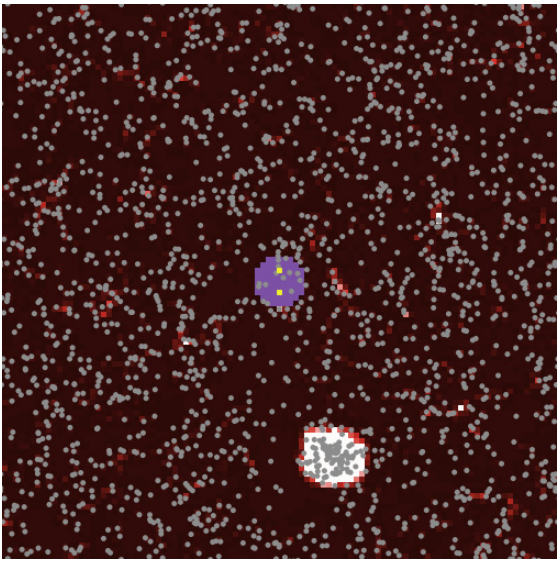


Fig. 1. Simulation view. The central region (purple) contains (+) and (-) *sub-clones* of *stem cells*. The peripheral region (black) shows agents (*peripheral lymphocytes*) migrating randomly but following a *chemical* concentration gradient. The white area shows the emergence of an spontaneous, escalated and self-propagated focus of an increase concentration of *chemicals*, which represents the origin of an inflammatory response. This occurred at proportion of 0.44 (-) : 0.56 (+) after 18544 iterations or tics and with 2034 agents.

3 Conclusions

The main characteristics of the theoretical model previously described [5], are reproduced in this simulation. It is considered to be non-deterministic because the recursive stochastic generation of agents can produce different outcomes from identical initial conditions. It does not depend on pattern recognition, since even when the simulation is specific and recognition of agents and patches is implicit, it can generate different outcomes. Therefore, it is not antigen recognition itself, but the random events that occur during the processing of the antigen that determine the occurrence of a response, as illustrated in Fig. 1. The model is also consistent with current knowledge in immunology, as all the components and steps have their biological equivalents. It suggests that the robustness of immunological memory is a function of the recursive process in conjunction with a large number of agents, and that changes in this memory may occur as a consequence of changing the central lymphatic system. This may explain paradoxical phenomena, such as generation or cure of autoimmune disorders in cases of bone marrow transplants and other conditions.

References

1. Klein, J., Sato, A.: The HLA system. First of two parts. N. Engl. J. Med. 343, 702–709 (2000)
2. Klein, J., Sato, A.: The HLA system. Second of two parts. N. Engl. J. Med. 343, 782–786 (2000)
3. Kitano, H.: Systems Biology: Toward System-level Understanding of Biological Systems. In: Kitano, H. (ed.) Fundations of System Biology, pp. 1–36. MIT Press, Cambridge (2001)
4. Wilensky, U.: NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (1999), `http://ccl.northwestern.edu/netlogo/`
5. Salazar-Bañuelos, A.: Immune Responses: A Stochastic Model. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 24–35. Springer, Heidelberg (2008)
6. Vaz, N.M., Varela, F.J.: Self and non-sense: an organism-centered approach to immunology. Med. Hypotheses 4, 231–267 (1978)
7. von Boehmer, H., Teh, H.S., Kisielow, P.: The thymus selects the useful, neglects the useless and destroys the harmful. Immunol. Today 10, 57–61 (1989)

Dendritic Cell Trafficking: From Immunology to Engineering

Emma Hart and Despina Davoudani

Edinburgh Napier University, Edinburgh, Scotland, UK
{e.hart,d.davoudani}@napier.ac.uk

The field of Artificial Immune Systems (AIS) has derived inspiration from many different elements of the natural immune system in order to develop engineered systems that operate in environments with constraints similar to those faced by the immune system [1]. A recent shift in thinking in AIS advocates developing a greater understanding of the underlying biological systems that serve as inspiration for engineering such systems by developing abstract computational models of the immune system in order to better understand the natural biology [2]. In this paper, we present results from a study in which agent-based modelling techniques were used to construct a model of dendritic-cell trafficking in the natural immune system with the aim of translating this model to an engineered system: a large-scale wireless sensor network. Our results highlight some generic issues which may arise when modelling biology with the intention of applying the results to AIS, rather than when modelling in order to replicate observed biological data. We suggest that the constraints of the engineered system must be considered when iterating the model, and that certain aspects of the biology may not be appropriate for the engineered system in question.

Our study is concerned specifically with modelling the trafficking of dendritic-cells and in understanding the role of these cells in serving as *sentinels* of the immune system[1]. In this respect, DCs circulate through the tissues sampling antigen, and when stimulated they migrate to the lymph node. There, they present a snapshot of the potentially infected site, and initiate either an immunogenic or tolerogenic response. Studying this aspect of DC functionality is motivated by previous work in *SpeckNets* [3], a type of wireless sensor network, in which it is required to capture information regarding the current state of parts of the network and react appropriately; radio messages circulating through the network act as DCs, sampling information stored at individual nodes, and return that information to more powerful specks positioned throughout the network which act as local *lymph nodes*.

Detailed information regarding the mapping of the dendritic cell behaviours to SpeckNets can be found in [4]. Although the analogy is on the surface appealing, further study of the literature quickly reveals immensely complex mechanisms, involving vast numbers of cytokines, chemokines and other cells. Examination of the Specknet and WSN literature reveals an equal number of difficulties associated

[1] Note that this is in contrast to the majority of work inspired by DCs in AIS which focuses on the differentiation capabilities of these cells and therefore their function as classifiers.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 11–13, 2009.

with working with such a constrained system. Making a leap from one complex system directly to another is not only practically challenging but also runs the serious risk of, on the one hand, misinterpreting or overlooking crucial aspects of the immunological system, and, on the other, creating a model or an algorithm which although immune-inspired is unfeasible due to the engineering constraints of the application. Providing a principled method of linking the two complex fields is therefore the main driver for this work.

We have chosen to implement a model of DC trafficking using NetLogo [5]. Forrest and Beauchemin [1] have discussed in depth the effectiveness of applying Agent-Based Modelling techniques (ABM) to immunology. ABM is highly applicable to modelling DCs, given that we wish to model populations of different cells circulating through an environment. Given that Specks themselves are still currently under development, work in SpeckNets is still performed mainly in simulation. A specialised simulator has been developed which mimics many aspects of Speck hardware and simulates random placement of Specks in either a 2D or 3D environment. This maps naturally to an ABM environment such as Netlogo which currently provides agent and graphical capabilites in 2D and in 3D in an experimental version.

A simplified model is developed which contains the following agents: DCs, mature DCs, semi-mature DCs, T cells, lymph nodes and two different cytokines. A single class of cytokine agents as a proxy for those cytokines expressed by matured and semi-matured DCs which attract them back to lymph nodes, and another single class of cytokine agent to represent cytokines which direct T cells to infected sites within the body. Lymphatic vessels are not physically represented due to the impossibility of replicating this in the engineered system. Instead, cytokine gradients are used to direct the trafficking of cells. Environmental information consists of infected sites, and signals (either *safe* or *danger*) which are generated in the tissue, and which can be collected by circulating DCs. We do not model most aspects of the adaptive response, as the main purpose of the model is to understand cell trafficking and information flow; we do however model helper T-cells which are produced by the lymph on receipt of matured DCs and travel back to areas of infection (to validate that a response can be directed). Agents move on a 2D grid in discrete time-steps. Movement of any agent is probabilistically determined by levels of cytokine at a location.

Using the model, we were able to determine that the simplified model displayed the high-level behaviours of DCs that we wished to capture, i.e. that circulating agents could effectively sample the environment and return a snapshot of that environment to a lymph node, and that the lymph was able to direct its adaptive response towards infected areas, see for example Fig. 1. Testing revealed however that some aspects would need to be modified in order to transfer the behaviour to a Specknet; this is necessitated by constraints occuring in relation to message transmission/receiving and the need to conserve energy in individual specks. For example, it became clear that mimicking DC generation from the tissue would pose severe problems in SpeckNet as it requires completely devolving control to invidual specks and leaves the network open to potential flooding; this

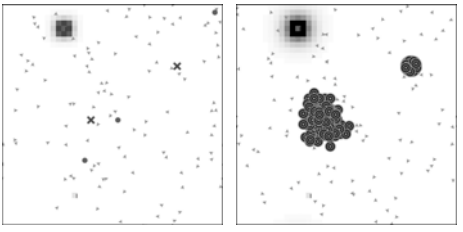


Fig. 1. The left-hand figure shows matured DCs (circles) travelling back to the lymph nodes (crosses) after detecting an infection (top left). The right hand figure shows the lymph node releasing T cells which are attracted back to the infected site via a cytokine gradient.

issue was resolved by delegating control of DC generation to the lymph specks in the engineered model which trades-off local vs global control. Another issue related to whether there was benefit in modelling trafficking of *immature* DCs back to the lymph, an issue which is unresolved in the biological literature,see [6] - this however has potential beneficial effects for the engineered system in that it returns a snapshot of the healthy system back to the lymph with little overhead to the network. The model was therefore modified, and re-tested to ensure that despite modifications made to mechanisms within the model, the same high-level behaviours were observed as in the more biologically faithful model. The final model was then used to implement a protocol in the Speck simulator; experimentation revealed that the protocol resulted in the desired behaviours, and as predicted by the model.

References

1. Forrest, S., Beauchemin, C.: Computer Immunology. Immunological Reviews 216(1), 176–197 (2007)
2. Timmis, J., Andrews, P., Owens, N., Clark, E.: An Interdisciplinary Perspective on Artificial Immune systems. Evolutionary Intelligence 1(1), 5–26 (2008)
3. Arvind, D., Elgaid, K., Krauss, T., Paterson, A., Stewart, R., Thayne, I.: Towards an integrated design approach to specknets. In: IEEE Int. Conf. on Communications, ICC 2007, pp. 3319–3324 (2007)
4. Davoudani, D., Hart, E., Paechter, B.: Computing the state of specknets: Further analysis of an immune-inspired model. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 95–106. Springer, Heidelberg (2008)
5. Wilensky, U.: Netlogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL (1999), `http://ccl.northwestern.edu/netlogo/`
6. Randolph, G.: Is maturation required for langerhans cell migration? J. Exp. Med. 196(4), 413–416 (2002)

A Hybrid Agent Based and Differential Equation Model of Body Size Effects on Pathogen Replication and Immune System Response

Soumya Banerjee and Melanie E. Moses

Dept. of Computer Science, University of New Mexico, Albuquerque, New Mexico
{soumya,melaniem}@cs.unm.edu

Many emerging pathogens infect multiple host species [1], and multi-host pathogens may have very different dynamics in different host species [2]. This research addresses how pathogen replication rates and Immune System (IS) response times are constrained by host body size. An Ordinary Differential Equation (ODE) model is used to show that pathogen replication rates decline with host body size but IS response rates remain invariant with body size. An Agent-Based Model (ABM) is used to investigate two models of IS architecture that could explain scale invariance of IS response rates. A *stage structured hybrid model* is proposed that strikes a balance between the detailed representation of an ABM and computational tractability of an ODE, by using them in the initial and latter stages of an infection, respectively.

The Immune System (IS) solves a search problem in both physical space and antigen space. The length of the search is determined by the time it takes for a cognate B-cell to encounter antigen. Our research suggests that this time is independent of the size of the organism [3]. This is counter-intuitive, since if we inject a sparrow and a horse with the same amount of antigen, the immune system of the horse has to search a larger physical space to find the pathogen, compared to the sparrow. This research attempts to explain how the time for the IS to search for antigen is independent of the size of the organism.

In addition to the immune system having to search larger spaces in larger organisms, larger body size can slow viral growth and immune system response times because the metabolic rate of cells is lower in larger species [4]. The metabolic rate of each cell is constrained by the rate at which nutrients and oxygen are supplied by the cardiovascular network. The rate at which this network supplies nutrients to each cell scales as the body mass (M) raised to an exponent of $-1/4$: $B_{cell} \propto M^{-1/4}$, such that individual cellular metabolic rates decrease as the body mass increases. The metabolic rate of a cell dictates the pace of many biological processes [4,5]. Metabolic rate is hypothesized to slow the speed of cell movement and proliferation in larger organisms. This could affect IS search times by reducing movement and proliferation of immune cells [6]. Rates of DNA and protein synthesis are also dependent on the cellular metabolic rate and could influence the rate at which pathogens replicate inside infected cells [2]. These two hypotheses, that IS search times and pathogen replication rates slow proportional to cellular metabolic rate and $M^{-1/4}$, lead to 4 possibilities, shown in Table 1 as originally proposed by Wiegel and Perelson [6].

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 14–18, 2009.

Table 1. Four scaling hypotheses of pathogen replication and immune system response rate [6]

H1: Pathogen replication rate $\propto M^0$ IS search time $\propto M^0$	H2: Pathogen replication rate $\propto M^{-1/4}$ IS search time $\propto M^0$
H3: Pathogen replication rate $\propto M^0$ IS search time $\propto M^{-1/4}$	H4: Pathogen replication rate $\propto M^{-1/4}$ IS search time $\propto M^{-1/4}$

In the first test of the effects of body size on pathogen replication and immune system response rates, we combine a differential equation model, an Agent Based Model and empirical results from an experimental infection study [7]. The same West Nile Virus (WNV) strain was used to infect multiple avian species with body mass ranging from 0.03 kg (sparrows) to 3 kg (geese), and the viral load was monitored each day in blood serum over a span of 7 days post infection (d.p.i.)[7].

A standard Ordinary Differential Equation (ODE) model was used to simulate viral proliferation and immune response, and model results were compared to empirical levels of virus in blood [3]. In the model, p = rate of virion production per infected cell, γ= innate IS mediated virion clearance rate, ω= adaptive IS proliferation rate, t_{pv} = time to attain peak viral load.

$$dT/dt = -\beta TV \tag{1}$$

$$dI/dt = \beta TV - \delta I \tag{2}$$

$$dV/dt = pI - c(t)V \tag{3}$$

$$c(t) = \gamma,\ t < t_{pv} \tag{4}$$

$$c(t) = \gamma e^{\omega(t - tpv)},\ t \geq t_{pv} \tag{5}$$

Target cells T are infected at a rate proportional to the product of their population and the population of virions V, with a constant of proportionality β. Infected cells I die at a rate δI, and virions are cleared by the immune system at the rate $c(t)V$. The action of the immune system is decomposed into an innate response before peak viremia (γ), and an adaptive immune response after peak viremia characterized by a proliferation rate ω. This paper focuses on p as pathogen replication, and γ and ω as IS response.

The parameters of the ODE model were fit to the viral load data for each of 25 species for days 1 – 7 d.p.i using non-linear least squares regression. According to the fits, pathogen replication rate (p, virions produced per infected cell per day) scaled as $p \propto M^{-0.29}$ (the predicted exponent of -0.25 is in the 95% CI, r^2 = 0.31, p-value = 0.0038). However, innate immune system mediated pathogen clearance rate (γ, day^{-1}) and adaptive immune system cell proliferation rate (ω, day^{-1}) were independent of host mass M (p-values of 0.4238 and 0.7242 respectively). These findings are consistent with hypothesis H2: pathogen replication rates decline in larger hosts, but immune response is independent of host mass.

Empirical data shows that time to peak viremia (t_{pv}) for WNV empirically occurs between 2 - 4 d.p.i., supporting the hypothesis that IS response rates are independent of M. If this peak were due to target cell limitation, then we would expect t_{pv} to increase with host mass M, since the rate of pathogen replication decreases with M and larger animals have more target cells. However, t_{pv} could be determined by WNV specific antibodies, which have a critical role in WNV clearance [8]. If the peak is determined by a threshold presence of antibodies, it implies that the time for cognate B-cells to recognize antigen, proliferate and produce antibodies is independent of host mass: $t_{pv} = t_{detect} + t_{prolif} \propto M^0$.

These findings raise the question: what mechanisms make IS rates independent of host body mass and metabolism? A lymph node could have "privileged metabolism" which is independent of host mass [6]. Lymph nodes also form a *decentralized detection network* in which they are evenly distributed throughout the organism, and each lymph node serves as a central place in which IS cells, antigen presenting cells (e.g. dendritic cells) and antigen encounter each other in a small local region of tissue. This decentralized network could lead to efficient and expedient antigen detection that is independent of organism size.

The ODE model supports H2: for WNV, viral replication is constrained by host mass, but immune response appears independent of host mass. However, ODEs are unable to explain how the spatial arrangement of lymph nodes affects the time for the immune system to respond to infection.

1 Two Competing Agent Based Models to Explore Mass Invariance of IS Response

In order to explore how the spatial arrangement of lymph nodes affects time to detect antigen, we used a spatially explicit Agent-Based Model (ABM). We used the Cy-Cells [9] ABM to explicitly represent each cell and virion, and simulated viral replication in a 3D compartment representing the lymph node and draining tissue. The model is informed with data on Dendritic Cell (DC) and T-cell movement from recent *in-vivo* microscopy experiments [10]. We simulated DCs, B-cells, viruses and lymph nodes, and explicitly modeled DC migration from tissue to lymph node, and random walk of DC and B-cells in lymph node. We simulated DC morphology by letting them have a high surface area and large dendrite sweep area [10].

In our first model, we assumed that the lymphatic network forms a *decentralized detection network*, such that each lymph node and its draining tissue function as a unit of protection, which is iterated proportional to the size of the organism (similar to the concept of a *protecton* [11]), i.e. lymph nodes in all organisms are of the same size, lymph nodes are distributed evenly throughout the volume of each organism, and an organism 100 times bigger will have 100 times more lymph nodes, each of the same size. We also assumed that lymph nodes have *preferential metabolism* [6] i.e. inside a lymph node, IS cells have speed and proliferation rates that are invariant with host mass M. The ABM was then used to simulate a cubic compartment of length 2000μm, representing a lymph node and its draining tissue region. The model showed that the time taken for a cognate B-cell to detect antigen on a DC (t_{detect}) is independent of

host mass and is slightly less than one day (20 simulations, mean = 16.43 hrs, SD = 13.83 hrs).

In the second model, we tested the alternative extreme that lymph nodes are arranged in a *centralized detection network* (bigger organisms have the same number of lymph nodes as smaller ones, however the size of an individual lymph node is larger) and they have preferential metabolism. We simulated 3 sizes of lymph nodes - a cubic lymph node of length 1000μm in a base animal, a cubic lymph node of length 2000μm in an animal 8 times bigger, and a cubic lymph node of length 5000μm in an animal 125 times bigger than the base animal. This model predicted that $t_{detect} \propto M^{0.93}$ (the exponent is theoretically predicted to be 1, and 1 is in the 95% CI, $r^2 = 0.99$, p-value = 0.01, 32 simulations) i.e. if a sparrow detects antigen in 12 hrs, then a goose would take 50 days, which is clearly unrealistic.

These results are consistent with our hypothesis that the decentralized nature of the lymphatic system, DC morphology and behavior, and privileged metabolism effectively reduce the antigen search time, so the classic search for a "needle in a haystack" problem can be solved in time independent of M. If $t_{pv} = t_{detect} + t_{prolif}$, and $t_{detect} \approx 1$ day independent of M, this implies that a fixed amount of time (1 to 3 days) is allocated to B-cell proliferation (t_{prolif}). Investigating how $t_{pv} = t_{detect} + t_{prolif}$ is independent of M, is ongoing.

2 A Stage-Structured Hybrid Model

ODE models implicitly assume that populations are homogeneously mixed, for example, that at initialization, each injected virion has the opportunity to come in contact with every normal cell. This is unrealistic since inoculated virions localize at the site of infection. Such spatial effects assume more importance during the onset of infection, when the number of virions that are carried to the lymph nodes is small and depends on the spatial arrangement of lymph nodes and the spatial interactions of multiple cell types within the LN and draining tissue. An ABM can be used to model these complex interactions. However, due to the level of detail at which individual entities are represented, ABMs can be prohibitively computationally expensive. This study outlines an approach that aims to strike a balance between the detail of representation of an ABM and the computational tractability of an ODE model.

We call this a *stage-structured hybrid modeling* approach, which uses a detailed and spatially explicit, but computationally intensive Agent-Based Model (ABM) when spatial interactions matter, and a coarse-grained but computationally tractable Ordinary Differential Equation (ODE) model when the ODE assumptions of homogeneous mixture of population are likely to be satisfied and spatial effects can be ignored. We utilize this modeling approach to elucidate dependence of pathogen replication rates, and IS search times and rates, on host body size.

Our models support the hypothesis that pathogen replication rates decline with body mass, but IS detection is independent of mass. The ABM shows that the latter can be explained by the decentralized architecture of the lymphatic network.

References

1. Woolhouse, M.E.J., et al.: Population Biology of Multihost Pathogens. Science 292, 1109–1112 (2001)
2. Cable, J., Enquist, B., Moses, M.E.: The Allometry of Host-Pathogen Interactions. PLoS ONE 2(11), e1130 (2007)
3. Banerjee, S., Moses, M., Perelson, A.S.: A Mathematical Model of Body Size Effects on Pathogen Replication and Immune System Response (2009) (in preparation)
4. Brown, J.H., et al.: Toward a Metabolic Theory of Ecology. Ecology 85, 1771–1789 (2004)
5. West, G.B., et al.: A general model for the origin of allometric scaling laws in biology. Science 276, 122–126 (1997)
6. Wiegel, F.W., Perelson, A.S.: Some Scaling Principles for the Immune System. Immunology and Cell Biology 82, 127–131 (2004)
7. Komar, N., et al.: Experimental infection of North American birds with the New York 1999 strain of West Nile virus. Emerg. Infect. Dis. 9, 311–322 (2003)
8. Diamond, M.S., et al.: A Critical Role for Induced IgM in the Protection against West Nile Virus Infection. Journal of Experimental Medicine (2003), doi:10.1084/jem20031223
9. Warrender, C.E.: CyCells simulator (Open source software) (2005), `http://sourceforge.net/projects/cycells`
10. Miller, M.J., et al.: T cell repertoire scanning is promoted by dynamic dendritic cell behavior and random T cell motility in the lymph node. Proc. Natl. Acad. Sci. USA 101, 998–1003 (2004)
11. Cohn, M., Langman, R.E.: The Protecton: the evolutionarily selected unit of humoral immunity. Immun. Rev. (1990)

Nonself Detection in a Two-Component Cellular Frustrated System

F. Vistulo de Abreu and P. Mostardinha

Departamento de Física, Universidade de Aveiro, 3810 Aveiro
{fva,pmostardinha}@ua.pt

The cellular frustration concept was first introduced in an ICARIS conference [1] as an alternative approach to accomplish specific and prompt intrusion detection in highly diverse systems. Cellular frustration uses two main assumptions: 1) that cells' activation are better modeled as cellular decisions and 2) that these cells' decisions require a finite amount of time to be triggered. These two assumptions have been gaining experimental support. It is now well established that T cells activation requires a considerable amount of time [2]. Experimental observations also suggest that interactions of T cells with APCs could be better modeled as cellular decisions rather than conventional probabilistic reactions [3],[4]. As a result from these assumptions the cellular frustration framework (CFF) is capable of reconciling two apparently opposing phenomena, namely high reactivity against nonself with total tolerance towards self [4].

The cellular frustration framework differs essentially from other approaches in how detection events (or cellular activations) are triggered [5]. During the time two cells start interacting and maturing this interaction, any of the two cells can contact a third cell. If it senses a stronger interaction, and the same happens with the third cell, then previous interactions are terminated so that a stronger interaction is established with the third cell. As a result of this decision dynamics, intrusion detection becomes an emergent phenomenon: cellular activation depends on interactions with other cells in the system, which may, or may not, frustrate the previously formed interactions. This contrasts with classical reactive models, such as Negative selection (NS) algorithms [5],[6]. NS algorithms are extremely intuitive. They assume that the space of patterns can be divided in two sets, self or nonself. The first step in the algorithm is to find a set of detectors with detection domains covering the whole nonself space. This is not always a computationally simple task and alternative methods exist leading to different performances [7]. Having defined the set of detectors, then a pattern is matched against all of them, and if it fits in a detector's domain a detection event is triggered. Hence, intrusion detection events are triggered strictly depending on the attributes of single two-cell contacts. The two approaches are thus extremely different and it is important to understand how both tackle similar simple problems.

To understand how the cellular frustration approach can be used to solve the same conceptual problem addressed by NS algorithms, the problem can be stated as follows. Given a set of arbitrarily diverse self patterns $S=\{s_i\}$ $(i=1,\ldots,N_s)$, define an algorithm and a set of detectors $D=\{d_i\}$ $(i=1,\ldots,N_d)$, such that the probability that at least one detector triggers a detection event against an arbitrary nonself pattern, p, is

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 19–21, 2009.

always higher than the probability of triggering a detection event against *any* self pattern. If $P(s_i)$ is the probability that the s_i self pattern triggers a detection event, then we want to define D that guarantees that:

$$P(s_i) < P(p),\ \forall s_i \in S,\ \forall p \notin S\ . \tag{1}$$

We should emphasize that the CFF can even trigger detection events against self patterns if they are presented at frequencies that differ substantially from those defined as self [4]. This contrasts to what happens in NS algorithms, where self patterns can be persistently presented in an abnormal manner and may consequently harm the hosts' normal functioning, without being detected. Homeostatic responses against unregulated presentation of self patterns can thus be an important advantage of the CFF, since it can respond to non-regulated presentation of self patterns, i.e. to cancer in the real immune system.

The example presented in [4] of Circular Frustrated Systems (CFS) is paradigmatic in how cellular frustration can perform specific and prompt detections. Each cell dynamics is established by their interaction lists (ILists). Motivated by the Principle of Maximal Frustration, ILists in CFS were built in [4] so that 1) if the i^{th} cell that was on the top of the j^{th} cell IList, then the j^{th} would be on the bottom of the i^{th} cell IList; 2) all cells appeared on the top of the IList of some cell. The same rationale is followed in the subsequent positions of each cell IList.

Here we report results from simulations similar to those obtained in CFS [4]. However now the model uses two cell types, APCs (or, in a language closer to artificial immune systems applications, presenters) and T cells (the detectors). Ligands and receptors of both cell types are associated to real numbers, x_L and x_R, in a bounded domain. In the example presented below their coordinates lie within -2 and 2. Furthermore, periodic boundary conditions are assumed. APC ligands have arbitrary coordinates, and it is assumed that each self APC (i.e., an APC presenting no pathogen) bear receptors that correlate with their own ligands according to a specific rule. Without loss of generality it was assumed that coordinates of APC ligands and receptors are the same (although this is not strictly required). It should be stressed this assumption does not mean that APC ligands and receptors are the same - which would be nonsense. In fact, both coordinates are defined in different spaces and consequently the previous assumption implies only that a correlation exists between the two. I.e., it is assumed that a correlation exists between the patterns presented by an APC, and the APC's receptors.

To each different APC it is assumed that m T cells exist, having ligands that match the APC receptor. When $m \neq 1$, then small perturbations are introduced to avoid having exactly identical cells and to increase frustration. T cell receptors follow the same rationale as in CFS: their coordinates are at the maximal distance from the APC receptors coordinates that exactly match a given T cell. Consequently, as in CFSs, frustration is maximized. This model can be called a two-component CFS, since it follows from the CFSs presented in [4], but it has only two different cell types.

We performed numerical simulations on systems with arbitrary diversity, using an algorithm similar to the one presented in [4]. In figure 1 we show a typical result. On the left the cumulative distributions of conjugate lifetimes are presented. Distributions for each self APC are presented in thin lines, and for the nonself APC in bold dots. Two examples are shown corresponding to two nonself detections. On Figure 1c we

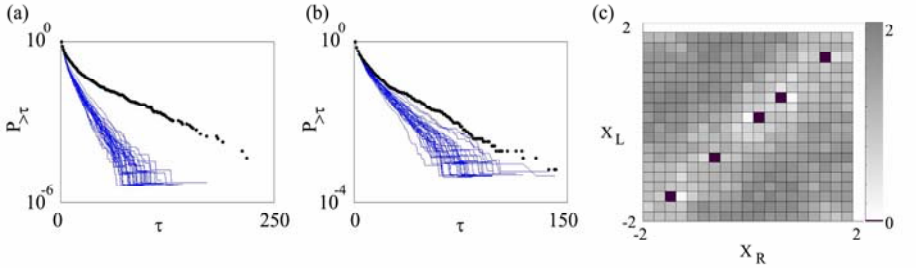


Fig. 1. (a,b) Cumulative distributions for conjugates lifetimes for self APCs (thin lines) and a pathogenic APC (dots), when the pathogenic APC is at (a) x_L=-1.8, x_R=0 and (b) x_L=x_R= -1.8. (c) The detection intensity χ as a function of the pathogenic APC coordinates. In black are the locations of self APCs. Clusters with 5 self APCs were placed at x_L=x_R= -1.6, 0, 0.4 and 1.2 and a cluster with 15 cells was placed at x_L=x_R= -0.8. We used m=4.

display $\chi = \ln\left(P_\tau^{Pathogen} / < P_\tau^{Self\ APC} >\right)$, where τ=40. Here $< P_\tau^{Self\ APC} >$ corresponds to the cumulative distribution of conjugates lifetime for the average of all self APCs. Hence it is possible to perform perfect self-nonself discrimination; only in the regions where self APCs lie is $\chi \sim 0$. Hence this model shows that nonself detection can be performed in a very different way as prescribed by NS algorithms. This work further calls the attention to the CFF, as an alternative intrusion detection method.

Acknowledgments. P.M. acknowledges support from FCT (SFRH/BD/37625/2007).

References

[1] de Abreu, F.V., Nolte-'Hoen, E.N.M., Almeida, C.R., Davis, D.M.: Cellular frustration: A new conceptual framework for understanding cell-mediated immune responses. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 37–51. Springer, Heidelberg (2006)

[2] Celli, S., et al.: Decoding the dynamics of T cell-dendritic cell interactions in vivo. Immunological Reviews 221, 182–187 (2008)

[3] Depoil, D., et al.: Immunological synapses are versatile structures enabling selective T cell polarization. Immunity 22, 185–194 (2005)

[4] de Abreu, F.V., Mostardinha, P.: Maximal frustration as an immunological principle. Journal of the Royal Society Interface 6, 321–334 (2009)

[5] Bentley, P.J., Kim, J., Aickelin, U., Greensmith, J., Tedesco, G., Twycross, J.: Immune system approaches to intrusion detection – a review. Natural Computing 6, 413–466 (2007)

[6] Ji, Z., Dasgupta, D.: Revisiting negative selection algorithms. Evolutionary Computation 15, 223–251 (2007)

[7] Hart, E.: Not all balls are round: An investigation of alternative recognition-region shapes. In: Jacob, C., Pilat, M., Bentley, P., Timmis, J. (eds.) ICARIS 2005. LNCS, vol. 3627, pp. 29–42. Springer, Heidelberg (2005)

Questions of Function: Modelling the Emergence of Immune Response

Tom Hebbron and Jason Noble

University of Southampton, Southampton, SO17 1DJ, UK
tomhebbron@zepler.net

1 Part One: Theoretical Background and Motivation

The practice of borrowing designs from nature to solve problems is increasingly prevalent in our technology development. The reductionist paradigm that allowed technology to advance to its present state fails when faced with many richly interacting components: a hallmark of complex systems. The behaviour of such systems is not solely determined by the aggregate behaviour of its components, but by emergent effects. The divide-and-conquer strategy is insufficient as the whole is greater than the sum of the parts.

Happily, the natural systems that surround us have been solving this class of complex problem for several billion years, optimising solutions through natural selection. By idealising these solutions we can aim to transfer successful strategies from biology to our own problem domains.

When borrowing ideas from biology in this way, there can be pitfalls in both under- and over-reaching. For example, an artificial immune system (AIS) algorithm that performs binary classification (e.g., "safe" / "dangerous") is not really doing what a real immune system does. The algorithm may do well at the practical task we set for it, but it is not embedded in real biochemistry in the way an immune system is. There is a superficial similarity but we would be wrong to draw conclusions from this understated algorithm about real immunobiology. Conversely, building a high-definition model of immune system components might be a valuable scientific goal, but as engineering it would be a misplaced effort if the complex contextual constraints of the real immune system were not present in the target problem. It is important to look closely at the differences between the practical problems tackled by AIS researchers (e.g., computer security) and the biological problem faced by real immune systems, as a false assumption of congruence will lead to either models that are too simple to be good science, or tools that are too byzantine to be good engineering.

This is the real crux of the problem: how to separate the useful functionality of a biological system from the legacy of its evolutionary history. Whilst evolution may appear to produce general purpose solutions, this is not the case; solutions in real biological systems are *nichiversal* not universal. Organisms evolve to solve the problems their ancestors faced, but not to solve a general class of problem [1]. Solutions are tightly bound to the context in which they arose, and represent a compromise balancing multiple conflicting constraints on the

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 22–26, 2009.

organism. A fundamental constraint is that the solution is made of the same biochemical substrate as the rest of the ecosystem it exists in. This is not the case in the relationship between artificial immune systems and the material that they work on; the AIS is able to step outside of the constraints of the data it is working with, breaking out of the bounds of the reflexive relationship that underlies biological immune systems.

This question of compatibility between a biological solution and a computation problem is a question of function, one of Tinbergen's "Four Questions": a schema of four equally valid categories of enquiry into biological or ecological systems [2]. In brief:

- **Mechanism:** how does the proximate system work? What are the component parts, and how do they interact?
- **Ontogeny:** in an organism, how does development of the mechanism proceed from origin to maturity?
- **Phylogeny:** what is the evolutionary history of the mechanism and its ancestors? What innovations appeared, what scaffolding disappeared? What is the particular path taken through the space of possible phenotypes by the ancestors of the modern organism?
- **Function:** what is the selective advantage in being equipped with such a mechanism? What ecological problem does it solve? The adaptations in the phylogenetic route taken by a species are specific responses to selection pressures. A truly explanatory account of an evolved mechanism must address the deeper structure of the problem for which the specific mechanism is one possible solution.

Questions of mechanism and ontogeny are of primary interest to the medical sciences, and it seems a great deal of AIS work draws from these models of proximate mechanism. Comparative genomics helps answer questions on phylogeny: when and where did recombination activating genes (RAG) arrive? How conserved are pattern recognition receptors (PRRs) across the eukaryota? The question of function however, is often presumed to be implicit — immune systems are *for* protecting against pathogens. This question warrants further exploration: for there to be pressure for an immune system in the first place requires certain dynamics in the ecology to hold — pathogenic behaviour must exist, and defences against it must be available and evolutionarily findable. Once an immune system emerges it makes fundamental changes to the future evolutionary pathways available to a lineage; some adjacent possible phenotypes will be incompatible with the defenses now encoded in the genome.

Why should we be interested in answering questions of function? High-resolution, predictive models of biological mechanisms are certainly useful, for instance in developing therapeutics; but such models are opaque. We can see how they work, but not why they work the way that they do, and such models are open to misinterpretation. For instance, low iron levels or a fever may not be a symptom of infection but an immune response, creating an unfavourable environment for a pathogen. Without knowing the function of this mechanism, there is the danger that therapeutics are employed to lower the temperature or

restore iron levels, working against the biological mechanisms [3]. This need for functional explanations to understand a mechanism in context is also an example of the "no free lunch" theorem: in short, you cannot evaluate an algorithm for solving a problem without some knowledge of the problem and how it arose, and there are no universal solutions — the success of an algorithm on one problem does not necessarily translate to another. This is significant for artificial immune system practitioners, who are trying to isolate the design principles of a system particularly tightly embedded in its biological context.

2 Part Two: Details of the Proposed Model

To investigate functional questions on the evolution of the immune system, we must consider the major transitions in immunity, and why they arose. We see these transitions as follows:

1. The protection of metabolism from random external perturbations (the vast majority of which are fatal), through some form of membrane.
2. Predation as a strategy; exploiting other organisms as local concentrations of resources and the corresponding coevolution of defensive counter strategies.
3. The accumulation in the genome of these defences — the emergence of innate immunity. The continuing selective advantage of these defences relies on the conservation of features of pathogens; features crucial to their metabolism or membrane and diffcult to mutate to non-recognised alternatives.
4. Asymmetry of pathogen / host evolutionary rate, and an increase in pathogenic load selecting for the emergence of somatic mutation: adaptive immunity to counter within-lifetime diversity of pathogens.

The emergence of a membrane to isolate an auto-catalytic set from the random perturbations of the outside world will be assumed in our model, as modelling at the level of physicochemical interactions that would be required to capture the emergence of membranes and proto-cells is below the level of abstraction that we believe will be necessary to observe higher level strategic events.

The substrate that our ecology will be built from is metaphorically closest to the metabolome in real biology. We require a level of abstraction that is higher than that involved in building an artificial chemistry, but that still allows for an organism to be made up of genetically specified phenotypic components that interact to provide energy payoffs. The model should be general enough to represent differing levels of epistasis in the landscape of energy payoffs for these component interactions.

We took inspiration from Kauffman's NK model of tunably rugged fitness landscapes [4]. An N by N interaction matrix defines the energy gain or loss $[-1.0, 1.0]$ of all possible interactions between metabolic components. There is no concept of concentration, so a metabolic 'state' in the model is represented simply by a bitstring of length N, which serves as an index to the interaction matrix: a 1 or 0 indicates the presence or absence of a particular metabolic complex.

Fitness will depend on the ability to reproduce, a function of genome size and energy gained from interactions. A membrane will be a given in the model, and organisms will be represented by two binary indices to the interaction matrix: internal and surface. The internal components are able to interact with the environment only via the surface. An abiotic background solution of nutrients will also be represented by a binary index to the interaction matrix. Energy gain or loss at each timestep is a function of the interaction between the internal and surface components and the surface and external state. Death occurs when energy falls below zero, and reproduction occurs when energy is sufficient to divide, and have both mother and daughter remain alive.

Reproduction with a small mutation rate will see adaptation; the population optimising surface and internal components according to the interactions available from environmental nutrients: a search for optima on a fitness landscape whose ruggedness is defined by the distribution of values in the interaction matrix. Of course, organisms do not exist independently, and Kauffman extended the NK model to consider coevolution in the NKC variant, where fitness landscapes are dynamic; deformed by the exploration of the linked fitness landscapes of other species. This separation of the fitness landscapes breaks with our idea that the substrate must be common to all members of the ecology: we want to see life converge on some parts of the landscape, and become contingently locked into others.

We expect that in some regions of parameter space, a number of the components available will be universal; they have such a high payoff that individuals employing them will dominate the population, and find it hard to mutate away from this dependence. The interesting dynamics will come from secondary components, those that have a high payoff, but are not universal. When contingencies of evolution lock these components into a species genome, and in particular if they are used as surface proteins, they become potential pathogen associated molecular patterns (PAMPs): conserved features that are difficult to mutate to unrecognised alternatives because of their importance to the pathogen's metabolic network, but not present in non-pathogen surfaces.

Parasitic and subsequent coevolutionary behaviour has been observed in artificial life systems before, notably in the Tierra system [5], which was not explicitly designed to exhibit the phenomenon. Parasitism in Tierra is an interesting example for us because it shows that a fixed and quite brittle substrate defined by Tierra's virtual machine instruction set, and which represents a single point in our parameter space, gave rise to parasitism and the beginnings of an immune response. This gives us confidence that regions of biological interest do exist in our own modelling space and will be found without too much difficulty.

We have several different goals in running the simulation. First we would like to find regions in parameter space that give rise to the evolution of a coevolutionary arms race between parasitic and host species, i.e., the beginnings of an immune response.

Our procedure will be to initially run the ecology with no organism / organism interactions — they optimise to background nutrients, essentially finding optima

on a static fitness landscape. We will then switch on organism / organism interactions, allowing the emergence of predatory behaviour. Under what parameters do we see some species become predatory - i.e. now specialise to feed on other species rather than the background nutrients?

Do we see a coevolutionary arms race between species? Or is it transitory? What regions of the PPI parameter space lead to the behaviours of interest?

We expect to see that species well adapted to the background nutrients end up on different optima than those who experience the pressures of predation and immune development. If we begin with a static landscape and allow species to explore and find optima, they will end up on different peaks when the landscape remains static (no interactions) than if the landscape has undergone a dynamic phase (the introduction and later removal of interactions and immune response). This will illustrate the way in which the pressure to adopt an immune system drives species along different phylogenetic routes and ultimately to different optima.

In future work, we intend to extend this modelling methodology to explore the functional questions that underlie the fourth transition identified; from innate germline immune responses to somatic adaptive responses.

References

1. Bullock, S.: The fallacy of general purpose bio-inspired computing. In: Rocha, L.M., Yaeger, L.S., Bedau, M.A., Floreano, D., Goldstone, R.L., Vespignani, A. (eds.) Tenth International Conference on Artificial Life, pp. 540–545. MIT Press, Cambridge (2006)
2. Tinbergen, N.: On aims and methods of ethology. Zeitschrift für Tierpsychologie 20, 410–433 (1963)
3. Nesse, R.M., Williams, G.C.: Why We Get Sick: The New Science of Darwinian Medicine, 1st edn. Vintage (1996)
4. Kauffman: The Origins of Order: Self-organization and Selection in Evolution. Oxford University Press, Oxford (1993)
5. Ray, T.S., Xu, C.: Measures of evolvability in Tierra. Artificial Life and Robotics 5(4), 211–214 (2001)

Object-Oriented Refactoring of Existing Immune Models

Hugues Bersini

CODE/IRIDIA – ULB
CP 194/6
50, av. Franklin Roosevelt
1050 Bruxelles, Belgium
bersini@ulb.ac.be

Abstract. Extending a previous plea of the author for adopting the OO practices in the modelling of immunological systems, this paper explains the process of restructuring an existing, interesting and complex immune model of T cell responses by adopting OO good practices, essentially the drawing of UML state and class diagrams and the implementation of the "State Design Pattern". This pattern associates to each state in which a T cell can be found, a single class responsible for describing both the internal transition taking place while in this state and the switching to the next state. Its exploitation entails a natural decoupling of the code, facilitating its comprehension and its re-use. This exercise aims at showing that both UML and this design pattern adoption greatly improves the readability, communication, and thus the possible modification of existing codes. Generalizing this process to all exploitable and existing immune models will allow the constitution of an utilizable library of understandable and reproducible simulations, something that seems to miss these days and hampers the software side of theoretical immunology to take off.

1 Introduction

Although there is a more than 20 years old tradition of immune software models, very few of them have been the object of further running and exploitation once their authors published the paper explaining it and event made the code easily available. Some simple very abstract models have received further attention but mainly due to their oversimplification, their pedagogical character or unexpected qualitative outcome, such as simulations grounded into cellular automata [2][3][11]. Once the model aims at getting beyond interesting but still very qualitative phenomena, and tries as much as possible to capture more refined biological knowledge, the resulting code turns out to be much too complex to attract further researchers. Roughly saying, the price to pay to understand and get into the software is not worth the benefits gained by running the code. The charge for admission is far too much. The amount of new knowledge acquired by understanding and running the code does not deserve such cognitive expanses. In a previous paper [1], I argued that theoretical immunologists could gain from the members of our ICARIS community the adoption of professional programming practices, rendering their software modelling more readable, scalable, usable and thus allowing to change this current frustrating situation of "write once run

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 27–40, 2009.

only once". Although algorithmic writing is less demanding than mathematical developments, still a great degree of rigour and a sharper clarification of biological mechanisms is required. The modelling of complex systems such as the immune system demands algorithmic ways to decouple the model so as to be able to pay attention to each of its part in turn without losing the global picture. This is exactly what the adoption of OO practices amounts to: making the "emergent" phenomena easily emerges out of the classes relationship network while keeping a perfect understanding and control of what happens in the system parts i.e. the classes.

OO software are simultaneously easy to read and to understand (even for non programmers), simple to build, easy to extend thanks to their inherent modularity, easier to maintain and to adapt. From its very origins, OO computation has simplified the programming of complex reality by allowing the programming to come closer to the way human perceives this reality (the first OO language was indeed called "SIMULA") instead of being constrained by the processor set of elementary instructions. Furthermore, the decoupling of this complex reality in simpler software classes, both in a vertical (classes inheritance) and in an horizontal direction (classes association), inherent to the OO practice, is a very old "Cartesian" trick that has never been contradicted to handle complex systems. UML proposes a set of well defined and standardized diagrams (transcending any specific OO programming language) to naturally describe and resolve problems with the high level concepts inherent to the formulation of the problem. It is enough to discover and draw (the UML language is essentially graphic) the main actors of the problem and how they do mutually relate and interact in time to build the algorithmic solution of this problem. Departing from these diagrams and in agreement with the software community which wants to substitute the programming practice (fully dependent of the computer platform) with the modelling one (fully independent), more and more automatic code generation tools appear on the market, contributing to make this computational practice evolution even more appealing to biologists. Finally, Design Patterns (DP) [9, 10, 11] are very convenient and well experimented software recipes to face and resolve programming difficulties often encountered during the development of complex software. One of these DP, the so-called "state pattern" (well explained in [9]), and originating from the UML state-transition diagram, will be amply used in this paper.

To convince the reader of the benefits gained by adopting OO practices, this paper will justify and describe the refactoring in an OO way of immune models by tackling one of them: a very interesting model of cytotoxic T cell responses proposed by Chao et al some years ago [4][5]. The original code is available and downloadable at http://www.cs.unm.edu/~dlchao/imm/download.html although I have been told, discussing with Chao, that he were unaware of many researchers having moved on, modifying or extending the code. This very same frustrating situation could apply to another as much interesting immune modelling effort by Efroni, Cohen and Harel [6][7]. The next section will remind both the two UML diagrams and the "state pattern" that are necessary to comprehend in order to follow this OO refactoring process. The third section will sketch the key parts of the Chao et al's model that has gone through this reorganisation exercise. The final one will justify and explain this restructuring and show pieces of the resulting updated software.

2 UML Diagrams and the State Design Pattern

It is impossible to even briefly give an overview of the hundred modeling symbols composing the 13 UML diagrams. A very simple and didactical introduction to UML is the purpose of Fowler's book [8]. As illustrated in the figure 1 below, the class diagram is essentially composed of the classes and the different types of relationships among them, such as association, composition and inheritance. Two classes are associated when objects of the first one need to run method declared in the second class so as to exploit or modify the current state of objects of the second classes. In the figure, a "TrafficLight" object is able to interact with the "Car" class (i.e. the many instances of car objects in front of it) so as to make them slow down or accelerate (thus modifying their speed attribute). A class is physically composed of another one, like in the figure the engine which really resides "into" the car, when the disappearance of the containing object entails the disappearance of its content. Whenever a car object leaves the RAM memory, an engine object automatically leaves it too. Finally the car class gives raise to two subclasses through the inheritance type of relationship, making the subclasses, first to be an exact replica of the superclass (thus the inheritance), but allowing additional specific attributes, methods, or redefining some methods already present in the superclass.

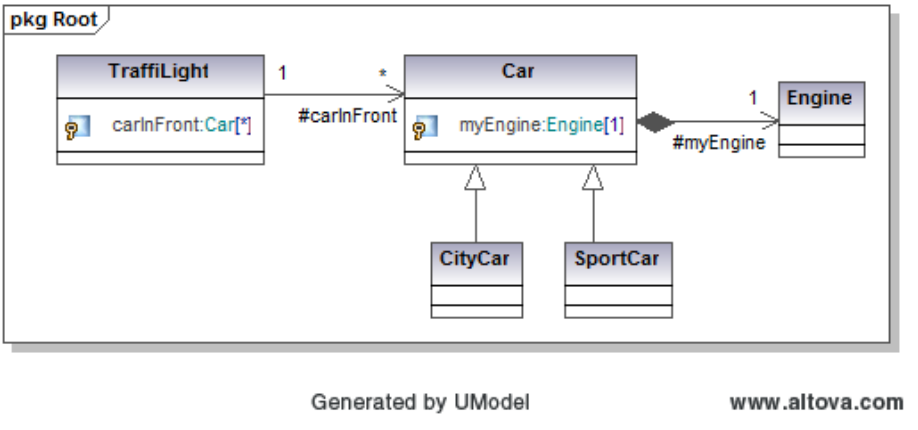


Fig. 1. Example of class diagram illustrating the three main types of relation among classes: association, composition and inheritance

All OO programmers know that there is a perfectly isomorphic code which is automatically generated by most of UML programming environments. A sketch of the Java version of the five classes would go as follow: 1) `class TrafficLight { private ArrayList<Car> carsInFront; public TrafficLight () { carsInFront = new ArrayList<Car>();} public void addCar(Car c) {carsInFront.add(c);}} – 2) class Car{ private Engine myEngine; public Car() {myEngine = new Engine();}} – 3) class Engine{} – 4) class CityCar extends Car{} – 5) class SportCar extends Car{}.`

The figure 2 below illustrates the state-transition diagram applied on the "Traffic Light" object and the associated class diagram which results from a straightforward application of the "state pattern". The traffic light can be in five elementary states regrouped into two composite states. The composite state is useful when a same transition (for instance the one indicated by [event3]) equally applies to all internal states. This diagram indicates all possible states an object can be in and all possible transitions (which can result from an event and/or be conditioned) between these states.

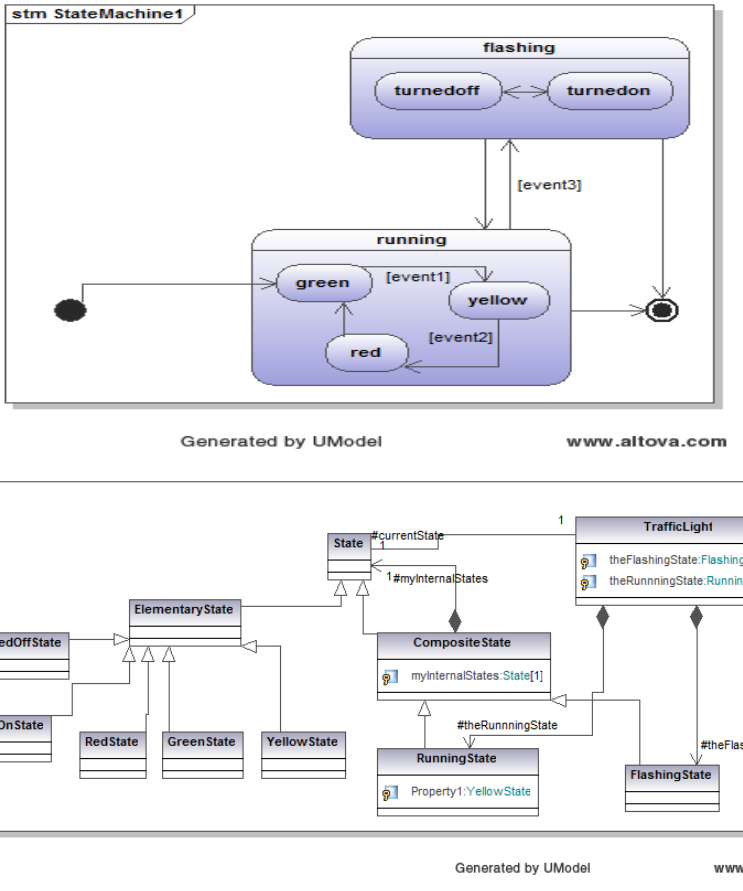


Fig. 2. The state-transition diagram of the TrafficLight and the Class diagram which results from a straightforward application of the "State Pattern"

In order to decouple the code as much as possible, the state pattern associates each state with one class responsible for what happens while the object is in this state . Part of the java code of the "TrafficLight" class goes as follows:

```
class TrafficLight {
      private ElementaryState currentState;
```

```
        private CompositeState currentCompositeState;
        private RunningState runningState;
        private FlashingState flashingState;
        public TrafficLight (ElementaryState currentState){
                carsInFront = new ArrayList<Car>();
                this.currentState = currentState;
                runningState = new RunningState(this);
                flashingState = new FlashingState(this);
        }
        public ElementaryState getYellowState() {
                return runningState.getYellowState();
        }
        public void changeState(ElementaryState newState){
                currentState.exitState();
                currentState = newState;
                currentState.enterState();
        }
        public void simulate() {
            while (true) {
                currentState.leaveState();
            }
        }
    }
```

The "changeState" method is responsible for switching the traffic light from one elementary state to another. In each state, it is possible to specify what should happen as long as the object stays in that state and just before exiting it. The abstract "State" class is shown below:

```
abstract class State {
        private TrafficLight theTrafficLight;
        public State (TrafficLight theTrafficLight){
                this.theTrafficLight = theTrafficLight;
        }
        public TrafficLight getTheTrafficLight(){
                return theTrafficLight;
        }
        public abstract void enterState();
        public abstract void exitState();
        public abstract void leaveState();
    }
```

It has three abstract methods (their name should reflect what they try to capture) to be specified for each state, either being composite or elementary. For instance, the "GreenState" will define these methods in the following way.

```
class GreenState extends ElementaryState {
        public GreenState(TrafficLight theTrafficLight){
                super(theTrafficLight);
        }
        public void enterState() {}
        public void exitState() {}
```

```
        public void leaveState() {
               if (event1){
                      greenToYellow();
               }
        }
        public void greenToYellow (){
            getTheTrafficLight().changeState(getTheTrafficLight().
                      getYellowState());
        }
}
```

Notice that the "leaveState" method regroups all possible transitions (each transition will then call the "changeState" method on the traffic light), depending here on the occurrence of a given event (event 1 in the example). This transition could be probabilistic in nature and thus associated with a probability of transition. In a stochastic version of this same diagram, the "leaveState" method would choose which transition to execute.

Interestingly enough, both the Java code and the class diagram have been automatically generated departing from the state-transition diagram. For some years now, there exits an XML description of all UML diagrams called XMI and equally standardized by the OMG. For instance, parts of the XMI description of the traffic light state-transition diagram is given below:

```
<subvertex xmi:type="uml:State" xmi:id="U2d38031e-88c0-4487-a358-
10033319382f" xmi:uuid="2d38031e-88c0-4487-a358-10033319382f"
name="running">
        <region xmi:type="uml:Region" xmi:id="U9975393a-c217-402a-8c28-
b4ad044f4921" xmi:uuid="9975393a-c217-402a-8c28-b4ad044f4921"
name="Region1">
        <subvertex xmi:type="uml:State" xmi:id="U0d02c7cb-1cc5-4cdb-
9951-f98db987fa6b" xmi:uuid="0d02c7cb-1cc5-4cdb-9951-f98db987fa6b"
name="yellow">
        <outgoing xmi:idref="U255848fd-71c6-4c37-9320-49f74f21bf91"/>
        <incoming xmi:idref="U89943d6b-b6dd-4e18-aa2d-1998dd625af9"/>
        </subvertex>
<subvertex xmi:type="uml:State" xmi:id="U95ab8769-8478-41a6-b00f-
51caa61d343b" xmi:uuid="95ab8769-8478-41a6-b00f-51caa61d343b"
name="green">
```

From this document, it is immediate to exploit a Java DOM parser in order to extract all information required to the automatic construction of the code: the "states" becoming the classes, the "transitions" ("outgoing" of the first state to "incoming" of the second one) constituting the body of the various transition methods conditionally called by the "leaveState" method. In my laboratory, a version of the parser has been developed for this specific problem so that a class diagram is automatically produced out of the state transition one.

3 The Chao et al's Stochastic Model of Cytotoxic T Cell Responses

The simulation software to be transformed is a very classical model of primary and secondary immune responses. It is important to precise that no functional modification

at all has been made on the original model, apart from a complete refactoring of the code in the several classes, as a result of a straight application of the State Pattern. It runs exactly in the same way but hopefully should be much easier to understand and transform in the future. The model roughly works that way (a complete description can be found in various publications [4][5]). Once a target cell is infected by a virus, it presents parts of the virus to T cells circulating around that become activated as an outcome of this recognition. Once activated and following a certain delay, T cell proliferate and destroy the infected target cell. Some of the effector T cells will later turn into memory cells which can, once again, respond to the virus but this time in a much more efficient way. All cells in the system naturally die at a certain rate. One very convenient way to understand the model is by attentively observing the two UML state diagrams shown below in figure 3 and 4.

The state-transition diagram of figure 3 shows how the population of uninfected target cells become infected once they meet the virus. Thus, a second transition makes them disappearing once they do encounter the T cell. It is clearly shown in the diagram how the fate of these target cells is linked both to the population of viruses and of T cells. Therefore three populations are involved here which, in the original study, are modeled by three differential equations responsible for their concentration evolution in time. Two general key aspects of the model need to be discussed further: its population-based and stochastic nature.

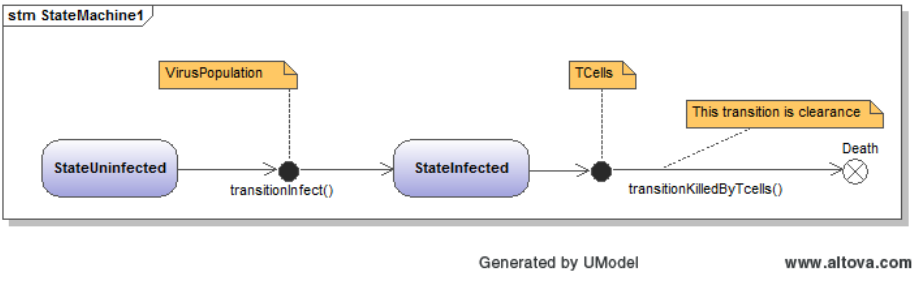


Fig. 3. State transition diagram of the target cell infected by the virus and cleared by T cell

In the UML original state transition diagram, only a single object and all its possible transitions are represented (like in the case of the traffic light). In Chao et al's model, we rather are in presence of various populations of objects whose transitions are followed in time. There is a long going debate in many scientific communities regarding the use of an agent-based approach (in which an agent is a single object) versus a population-based one (in which an object is a population of agents of the same type and thus contains a "size" attribute). Due to the huge number of immune cells present in an organism, and following the example of chemists who rather privilege the use of population-based kinetics models to agent-based ones, a population-based approach has been adopted in this model so as to be more faithful to reality and much more computationally effective. Notice that the state pattern is not affected by this choice and the resulting class diagram would look very much the same whatever choice is adopted: the single object or the population-based version. Like illustrated by the class diagrams below, a T cell object will now contain as many populations as states a T cell can be found in.

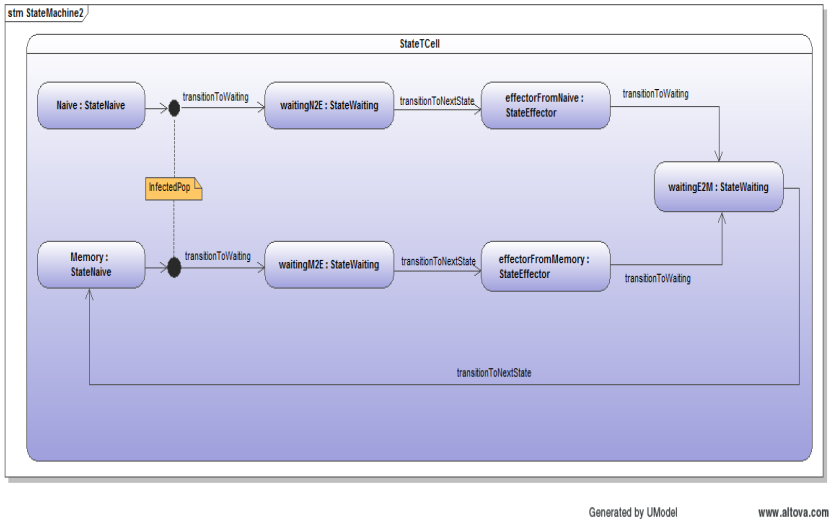


Fig. 4. The State transition diagram associated to the T cell

The second key aspect of the model is its stochastic nature. There are two reasons justifying the presence of probabilities in the model. One is again computational efficiency, such as when replacing the deterministic updating of the three differential equations describing the evolution in time of the uninfected, the infected target cells and the viruses by a stochastic version (this is classically done in chemical kinetics when the Gillespie algorithm is favored to the classical numerical integration). The second more fundamental reason is the reflection of the uncertainty in the description of the transition. Like shown in fig. 5, only a subpart of the cells in a given state transit to the next state, so that a random sampling of the population of the previous state disappears from that same state to move to the next one. With respect to the original UML state transition diagram, the transition guard will be stochastically defined.

Fig. 5. The stochastic transition between two states

The T cell state transition diagram shown in figure 4 is slightly more subtle. Initially the T cells are in the naïve state. Once they do meet an infected cell, they enter a waiting state, representing the time needed for the cell to become effector (proliferating and killing infected cells). This waiting state is implemented in the code by an array of populations (the size of the array is given by the number of time steps to wait). At each time step, new T cells enter the waiting state (in the first compartment of the array), while all cells in a stage (in an array compartment) move to the next stage (the next compartment). The cells in the last stage (the last compartment of the array) enter the next state i.e. the effector state (that they will have reached in exactly *t*

time steps). Notice that there are three instance objects of this same StateWaiting in the diagram: one from "naïve to effector", one from "effector to memory" and one from "memory to effector" which only differ by the waiting time (i.e. the size of the array). The effector state is slightly more complex than the other states and is indeed represented by a composite state like shown in figure 6 to be discussed later. Following the effector state, the cell enters a new waiting state before turning into a memory cell. These memory cells are represented by a second instance of the same naïve class since they behave exactly like the naïve one, although their stimulation is much more effective (some attributes change between the naïve cells and the memory cells increasing their reactivity). Again like for the naïve state, the effector state class gives raise to two objects (one coming from naïve and one from memory) which behave in the same way and differ only by some attributes values. Something not shown in the diagram is the spontaneous natural dying of the T cells in whatever state they are (and again this transition to death is of a stochastic nature).

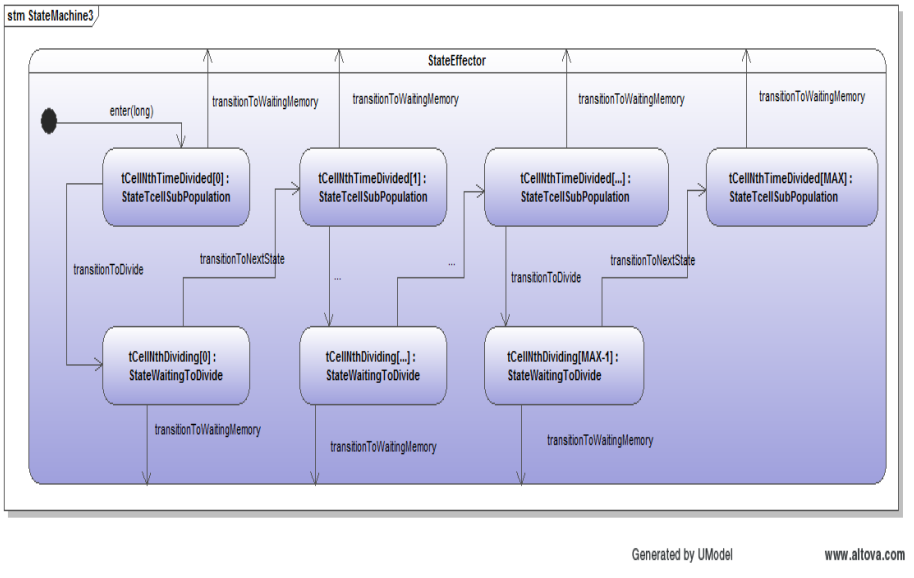


Fig. 6. The state transition diagram of the state effector which is a composite state

The state effector represented in figure 6 is a composite state composed of two classes of state: "StateWaitingToDivide" which is a subclass of "StateWaiting" and the "StateTcellSubPopulation". Let *n* be the number of times a T cell can successively divide and let *t* be the number of time steps necessary to perform one division. In the composite effector state, there will be n instances of objects "StateWaitingToDivide" and n+1 objects of "StateTcellSubPopulation". Each object "StateWaitingToDivide" is responsible for the division process and contains an array of populations of size t. The class "StateWaitingToDivide" inherits from the class "StateWaiting" by just redefining the transition to the next state which supplies two times the number of T cell in this next state instead of one. The T cell enters in this composite state by first

entering in a TcellSubPopulation state. This class precedes any division process and receives the result of the previous division process. In fact, *n* successive times, a T cell transits to the state TcellSubPopulation then to the state WaitingToDivide in order to divide, to finally attain the last instance of TcellSubPopulation i.e. the n+1th one which concludes the whole period spent in the state Effector. In addition, from whatever internal state, a stochastic transition is possible to the memory state.

4 OO Refactoring of the Model

The three class diagrams of the new reorganized version of the code are discussed in this section. Provided what the UML community and the OMG members keep telling is adequate, these diagrams should easily allow to understand the whole simulation with no real need to enter the code.

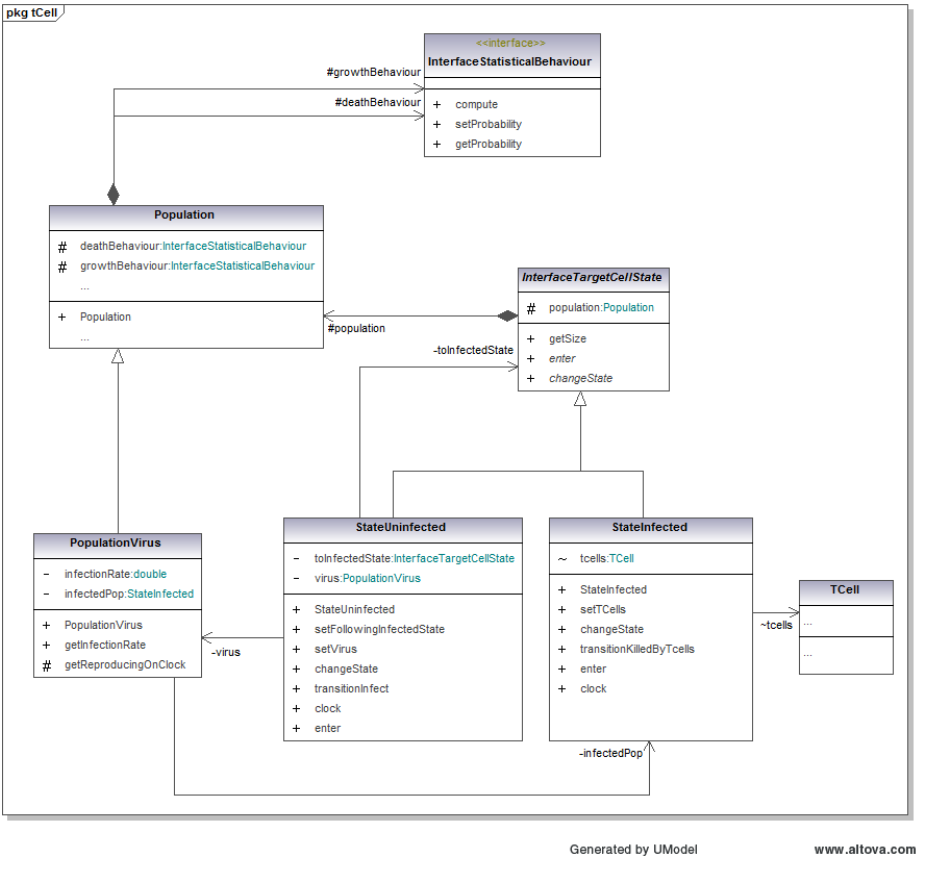


Fig. 7. The class diagram of the non infected, infected target cells and of the virus

Let's first discuss the classes shown in figure 7. The population class contains a "size" attribute. Its method "transitionToReproduce" accounts for the natural reproduction of the cell and "transitionToDie" for the natural dying process. The different statistical behaviors i.e. "Poisson" or "binomial", responsible for the two previous transitions, are declared separately and associated to that class in order not to confuse the biology and the algorithmic tricks. The virus population is a subclass. The rest of the figure is the part of the class diagram that automatically results from the application of the "state pattern" on the state-transition diagram of the target cell. The two states are then: "infected" and "noninfected", the first one requiring an association with the virus population and the second one with the T cell class. On account of the "population-based" nature of the simulation, each state of the target cell contains an object of type population which represent all T cell in that specific state. The state class contains the two methods "changeState" and "enter". The uninfected class contains the additional "transitionToInfected" method.

The following figure is more complete and more representative of the whole simulation.

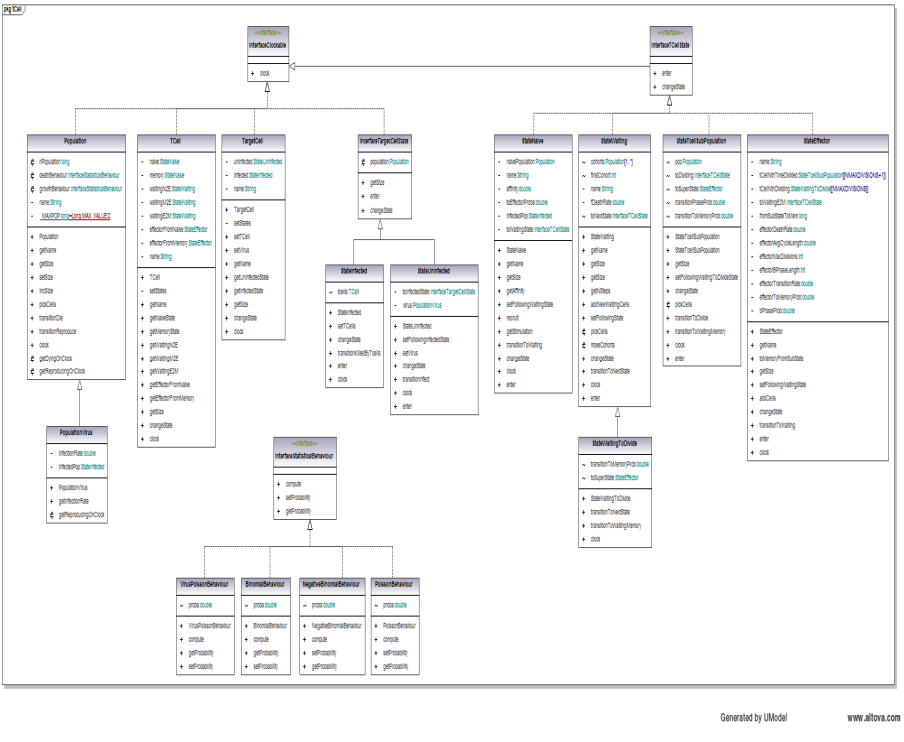


Fig. 8. A nearly complete class diagram of the reorganized code, with just shown the inheritance relationship among the classes

Four interfaces can be seen in the figure. A first one just contains the method "clock", which is called at each time step on each class implementing it. The "clock" method is responsible for every processes occurring at each time step. All biological classes implement this interface. Two other interfaces result from the application of the state pattern, first on the target cell then on the T cell. The last interface is just responsible for the different possible statistical processes describing the transitions between states. Implementing the interface "TcellState" are the five states already sketched above: "naïve", "waiting", "effector", "waitingToDivide" and "TcellSub-Population", the last two being contained in the effector composite state. The last class diagram of figure 9 describes in a more precise way the classes needed to run the T cell state transition diagram.

The T cell class contains 3 instances of the "waitingState", since the state transition diagram repeats three times the same waiting period but with different durations, and 2 instances of the "naïveState", the second one for the memory version of the T cell. A waiting state contains an array of populations of size the number of time steps to wait. The "StateEffector", for reasons already explained above, contains n instances of "StateWaitingToDivide" and n+1 instances of "StateTcellSubPopulation" (each object of this later class contains one "population" instance). In order to verify the perfect matching of the code and the class diagram, some parts of the Java code of the class "StateNaive" is shown below:

```
public class StateNaive implements tCell.InterfaceTCellState {

        private Population naivePopulation;
        private StateInfected infectedPop;
        private InterfaceTCellState toWaitingState;

        public void setFollowingWaitingState(InterfaceTCellState
        nextState){
                this.toWaitingState = nextState;
        }
        public long recruit(double fProb){
                return naivePopulation.pickCells(fProb);
        }
        /** transition to the next state */
        public long transitionToWaiting(InterfaceTCellState
        nextState){
                 long leaving = recruit(1.0 - Math.exp(-
        getStimulation()* toEffectorProba/ Con
        stants.TIMESTEPSPERDAY));
                nextState.enter(leaving);
                return leaving;
        }

        /** internal transition - here cells are dying */
        public long changeState(){
                return naivePopulation.clock();
        }
        public long clock() {
                long variation = 0;
                variation += changeState();
```

```
                variation += transitionToWaiting(toWaitingState);
                return variation;
        }
        /** entering from the previous state */
        public void enter(long entering) {
                this.naivePopulation.incSize(entering);
        }
}
```

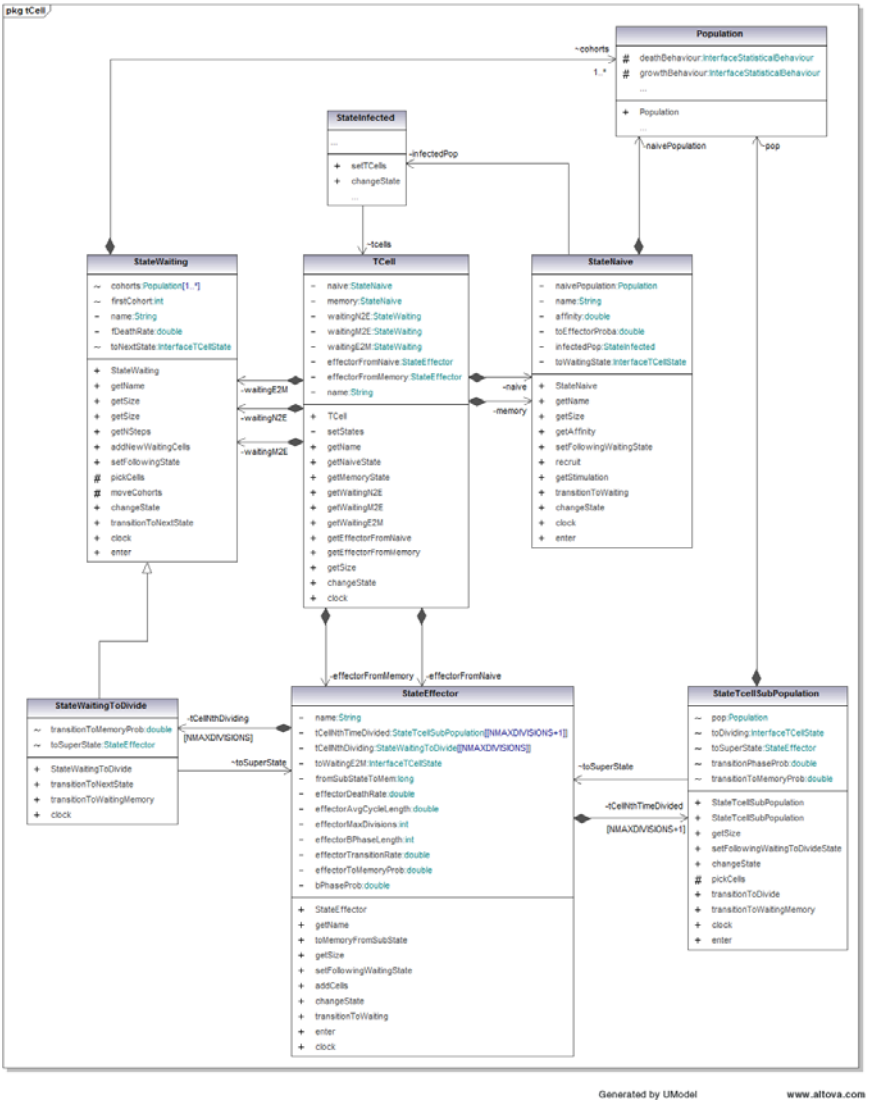


Fig. 9. The class diagram describing all classes necessary to code the T cell state transition diagram

5 Conclusions

OO languages, UML and Design Patterns all together allow to tackle the simulations of the immune system in a much more comprehensible, adaptable and effective way. Immunologists should be able to understand what the code does through the UML diagrams and even figure out how to change or to adapt it to their specific needs. They could easily re-use some of the classes described in the paper such as the "population" or the "waitingState". Through the use of UML diagrams, the necessary communication between programmers and between programmers and non programmers, in order to modify or extend existing immune simulations, should be greatly facilitated. I personally interacted with Denis Chao and members of the Cohen and Harel's team and in both cases decided to watch into the code of their T cell class. They honestly recognized that, due to the number of states transitions affecting this class, so far the coding of this class was too long, ugly, not very understandable, and particularly difficult to modify. This is why I decided to make this refactoring exercise and to write this paper to improve the situation and to encourage future programmers to adopt better software practices.

References

1. Bersini, H.: Immune system modeling: The OO way. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 150–163. Springer, Heidelberg (2006)
2. Bersini, H.: Self-Assertion vs Self-Recognition: A Tribute to Francisco Varela. In: Proc. of the first ICARIS Conference, pp. 107–112 (2002)
3. Celada, F., Seiden, P.E.: A computer model of cellular interactions in the immune system. Immunol. Today 13, 56–62 (1992)
4. Chao, D.L., Davenport, M.P., Forrest, S., Perelson, A.S.: Stochastic stage-structured modelling of the adaptive immune system. In: Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB 2003), pp. 124–131 (2003)
5. Chao, D.L., Davenport, M.P., Forrest, S., Perelson, A.S.: A stochastic model of cytotoxic T cell responses. Journal of Theor. Biol. 228, 227–240 (2004)
6. Efroni, S., Harel, D., Cohen, I.R.: Reactive animation: Realistic Modeling of Complex Dynamic Systems. Computer 38(1), 38–47 (2005)
7. Efroni, S., Harel, D., Cohen, I.R.: A theory for complex systems: reactive animation. In: Patton, R., Namara, L.M. (eds.) Studies in Multidisciplinarity, vol. 3, pp. 309–324. Elsevier, Amsterdam (2006)
8. Folwer, M.: UML Distilled, 3rd edn. Addison Wesley, Reading (2004)
9. Freeman, E., Freeman, E., Sierra, K., Bates, B.: Head First Design Patterns. O'Reilly Media, Sebastopol (2004)
10. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns. Addison Wesley, MA (1995)
11. Kleinstein, S.H., Seiden, P.E.: Computer simulations: simmulating the immune system. Comput. Sci. Eng. 2, 69–77 (2000)
12. Larman, C.: Applying UML and Patterns, 2nd edn. Prentice-Hall, Inc., Englewood Cliffs (2003)

Mathematical Model of HIV Superinfection and Comparative Drug Therapy

Anil Sorathiya[1,⋆], Pietro Liò[1], and Luca Sguanci[2]

[1] Computer Laboratory, University of Cambridge,
Cambridge CB3 0DG UK
[2] Dipartimento di Energetica, Università di Firenze,
Via S. Marta 3, 50139 Firenze, Italy

Abstract. We have modeled the within-patient evolutionary process during HIV infection. During the HIV infection several quasispecies of the virus arise. These quasispecies are able to use different coreceptors, in particular the CCR5 and CXCR4 (R5 and X4 phenotypes, respectively). The switch in coreceptor usage has been correlated with a faster progression of the disease to the AIDS phase. As several pharmaceutical companies are getting ready to start large phase III trials for their R5 blocking drugs, models are needed to predict the co-evolutionary and competitive dynamics of virus strains. Moreover, we have considered CTLs response and effect of TNF. We present a model of HIV early infection and CTLs response which describes the dynamics of R5 quasispecie and a model of HIV late infection, specifying the R5 to X4 switch and effect of immune response. We report the following findings: quasispecies dynamics after superinfection or coinfection have time scales of several months and become even slower in presence of the CTLs response. In addition, we illustrate dynamics of HIV quasispecies on HAART, Maraviroc and Zinc-finger nucleases(ZFN) therapies. Our model represents a general framework to study the mutation and distribution of HIV quasispecies during disease progression, and can be used to design vaccines and drug therapies.

Keywords: HIV, viral dynamics, quasispecies, coinfection, superinfection, HAART, Maraviroc, Zinc-finger nucleases.

1 Introduction

Human Immunodeficiency Virus(HIV) infection kills or impairs cells of the immune system, destroys resistance ability against infections and it leads to AIDS. According to UNAIDS/WHO, there are over 33 million people living with HIV and AIDS worldwide(`http://www.who.int/hiv/en/`). HIV, one of the deadest disease, is difficult to control because of very high mutational capability of the virus. Past years have seen major advances in the understanding of the interactions between viral envelope glycoproteins and hot receptors, and of the

⋆ Correspondence author.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 41–53, 2009.

conformational changes involved in the entry of HIV-1 into host cells [1]. The Chemokine receptors, CCR5 and CXCR4, have been identified as major coreceptors for HIV-1 entry into CD4+ T cells, playing critical role in the cell entry process and lead to a new classification of HIV-1. Virus entry through CCR5 is termed as *R5-tropic* (no-syncytium-inducing), and virus that exclusively uses CXCR4 is termed as *X4-tropic* (syncytium-inducing). Virus, able to use both CCR5 and CXCR4 receptors, is termed as *R5/X4-* or *dual-tropic*.

HIV-1 infection is characterised by the progressive loss of CD4+ T cells. Infection by most strains of HIV-1 requires interaction with CD4+ T cell and a chemokine receptor, either CXCR4 or CCR5. In early stage of HIV-1 infection, viruses that use CCR5 to enter into the CD4+ T cells and make them infected, are known as R5 HIV-1. In later stage of HIV-1 infection, viruses bind to the CXCR4 chemokine receptor (X4) in addition to CCR5 (known as R5X4) or CXCR4 alone in 50% patients [2,3,4]. These strains are infecting not only memory T lymphocytes but also naive CD4+ T cells and thymocytes through the CXCR4 coreceptor, which induce syncytium. Furthermore, R5-tropic virus for memory and X4-tropic for naive T cells may drive evolution of phenotypes with diseases progression [5]. Switching of CCR5 to CXCR4 has been linked to an increased virulence and with progression to AIDS, probably through the formation of cell syncytia and killing the T cell precursors. CXCR4 is expressed on a majority of CD4+ T cells and thymocytes, whereas only about 5 to 25% of mature T cells and 1 to 5% of thymocytes express detectable levels of CCR5 on the cell surface [6]. It is noteworthy that X4 HIV-1 strains stimulate the production of cellular factor called Tumor Necrosis Factor (TNF), which is associated with immune hyperstimulation, a state often implicated in T-cell depletion [7]. TNF seems to be able to both, inhibit the replication of R5 HIV strains while having no effect on X4 HIV and down regulate the number of CCR5 co-receptors that appear on the surface of T-cells [8].

A powerful concept in understanding the HIV variability and its consequences is that of quasispecies [9,10]. Quasispecies are the combined result of the mutations and recombination, that originates variability, and of the co-infection (simultaneous infection), superinfection (delayed secondary infection) and selection, that keeps variability low. HIV-1-infected individuals show heterogeneous viral populations of related genomes best described as viral quasispecies [11].

In fact, the infection capacity of mutants may vary, and also their speed of replication [12]. Moreover, since the number of targets (the substrate) is limited, fitter clones tend to eliminate less fit mutants, which are subsequently regenerated by the mutation mechanism [13]. While mutations are essential ingredients for exploring the genetic space in the search for the fitness maximum, they also lower the average fitness of the strain, that generally is formed by a cloud of mutants around the fitness maximum, the quasispecies. It is worth noting that, for a given fitness landscape, there is a maximum tolerable mutation rate above which the quasispecies structure is lost(see [9] for a recent work on error threshold).

In addition, Cytotoxic T Lymphocytes(CTLs) play vital role in controlling infection and variability of HIV-1. The pressure on HIV-1 progression is the

existence of either weak CTL selection pressure or viral mutations [14]. A recent study shows that higher degree of immunodominance leads to more frequent escape with a reduced control of viral replication but a substantially impaired replicative capacity of the virus [19]. We aim at modeling viral multi strain short and long term evolutionary dynamics especially transition of R5 to X4 phenotype switch and CTLs response.

The use of mathematical models is an insightful and essential complement to *in-vivo* and *in-vitro* experimental design and interpretation. Indeed mathematical models of HIV dynamics have been proved valuable in understanding the mechanisms of several observed features in the progression of the HIV infection [19,20,21,22,23,24,25,26,27]. The mathematical models are even more effective in predicting the time and space patterns, the effects of drug therapies and, the design of vaccines.

Here, we address the issue of studying the coevolutive and the competitive dynamics of CTL response and R5 to X4 phenotype switch during HIV-1 infection. In the next section we introduce: a quasispecies model focuses on the R5 to X4 shift, the hyperstimulation of T cell precursors through TNF [28] and the CTLs selection pressure of escape. We describe the decreasing dynamics of CD4+ T cells after the appearance of X4 strains and make predictions on the results of HAART, Maraviroc and Zinc-finger nucleases(ZFN) in coinfection and superinfection scenarios at different times of the disease progression. Finally we discuss comparison among therapies which might be helpful in medication of HIV-1 patients. In addition, immune response model may help in designing vaccines.

2 Models

In a quasispecies model for R5 phase mutations, co-infection and superinfection cause several R5 strains. In the limit of one quasispecies we found the same values observed in experiments and in other models (most notably Perelson's standard model). We tested the model in the scenarios of co-infection and superinfection using parameters derived from biological literature. The model has been discussed in our previous published literatures [10,28]. The model is as follows:

$$\frac{dU}{dt} = N_U - \delta^U U - \delta_F^U U F \tag{1}$$

$$\frac{dT_i}{dt} = \delta^U U - (\sum_k \beta_k V_k) T_i - \delta^T T_i \tag{2}$$

$$\frac{dI_k}{dt} = \left(\sum_{k'} \mu_{kk'} \beta_{k'} V_{k'} \right) \left(\sum_i T_i \right) - \delta^I I \tag{3}$$

$$\frac{dV_k}{dt} = \pi I_k - c V_k \tag{4}$$

$$\frac{dF}{dt} = K_F \sum_{k \in X4} V_k \tag{5}$$

Table 1. Parameters introduced in the R5 to X4 phenotypic switch model

Parameter	Symbol	value	Units
Production of immature T cells	N_U	100	$cell/\mu l\ t^{-1}$
Death rate of immature T cells	δ^U	0.1	t^{-1}
Death rate of immature T cells upon the interaction with TNF	δ_F^U	10^{-5}	$\mu l/cell\ t^{-1}$
Decreasing infectivity of R5 phenotype due to TNF	k_{R5}	10^{-7}	$\mu l/cell^2\ t^{-1}$
Increasing infectivity of X4 phenotype due to TNF	k_{X4}	10^{-7}	$\mu l/cell^2\ t^{-1}$
Increasing death rate of immature T cells due to TNF	δ_{X4}^I	0.0005	$\mu l/cell\ t^{-1}$
Rate of production of TNF	k_F	0.0001	t^{-1}

Model for the R5 to X4 phenotypic switch: a summary of the additional parameters is introduced. The values of the other parameters are from medical literature referred(see also [29]).

The studies have shown that the equilibrium abundance of the infected cells depend on immunological parameter [30,31]. Therefore, we introduce another variable, immune responce (Cytotoxic T Lymphocytes), in above model. Even in future, we intend to check model on vaccination and drug therapy.

2.1 Modeling of CTLs and Transition of R5 to X4

As shown in several studies, Cytotoxic T Lymphocytes (CTLs) play an important role in controlling HIV infection specially at initial stage [32,33]. However, there are many challenges to model immune response because HIV is difficult to diagnose in the early stage. Additionally, it is the challenge to acquire data on CTLs escape since the virus diversity within a host has to be followed over a long time. It is also difficult to analyse the intersection in viral replication, the selection and the mutation by different CTLs response. To better understand these processes, we model transition of R5 to X4 with immune response.

The model contains six variables: immature T cells(U), uninfected mature T cells(T), infection of mature T-cells (I), viral quasispecies (V), TFN (F), and CTL response (Z). The changes in the population over time can be described by following differential equations model:

$$\frac{dU}{dt} = N_U - \delta^U U - \delta_F^U U F + k_Z I Z \tag{6}$$

$$\frac{dT_i}{dt} = \delta^U U - (\sum_k \beta_k V_k) T_i - \delta^T T_i \tag{7}$$

$$\frac{dI_k}{dt} = \left(\sum_{k'} \mu_{kk'} \beta_{k'} V_{k'}\right) \left(\sum_i T_i\right) - \delta^I I - \delta^z I Z \tag{8}$$

$$\frac{dV_k}{dt} = \pi I_k - c V_k \tag{9}$$

$$\frac{dF}{dt} = K_F \sum_{k \in X4} V_k \tag{10}$$

$$\frac{dZ}{dt} = rIZ - bZ \tag{11}$$

Equation (6) describes the constant production of immature T cells by the thymus N_U and their transformation into mature T cells at rate δ_U. If X4 viruses are present, upon the interaction with TNF, immature T-cells are cleared at fixed rate δ_F^U, and added due to CTLs at fixed rate k_Z.

Equation (7) describes how uninfected mature T cells of strain i are produced at fixed rate δ^U by the pool of immature T cells. Those cells, upon the interaction with any strain of the virus, V_k, become infected at rate $\beta_k = \beta \quad \forall k$. The infectiousness parameter, β, is not constant over time, but depends on the interplay between R5 and X4 viruses. In particular, due to the presence of TNF, the infectivity of the R5 strains reduces ($\beta_{R5}(t) = \beta - k_{R5}F(t)$), while infectivity of the X4 strains increases($\beta_{X4}(t) = \beta + k_{X4}\ F(t)$).

Equation (8) describes the infection of mature T-cells. Infected T-cells of strain k arise upon the interaction of a virus of strain k with any of the mature T-cell strains. The infected cells, in turn, are cleared out at rate δ^I. In addition, the infected cells are cleared out by CTLs response at fixed rate δ^z. When TNF is released, this value increases linearly with constant δ_{X4}^I, $\delta^I(t) = \delta^I + \delta_{X4}^I\ F(t)$.

Equation (9) describes the production of viral strains from infected cells at fixed rate π, viruses are cleared out at fixed rate c.

Equation (10) models the dynamics of accumulation of TNF by assuming the increase in TNF level to be proportional, via the constant K_F, to the total concentration of X4 viruses present.

Finally, in the equation (11), we model CTLs response which is proportional to the concentration of the infected cells at constant rate r, CTLs are cleared out at fixed rate b.

Table 2. Additional parameters for CTL response Model

Parameter	Symbol	Value	Units
Increasing immature T cell due to CTL	k_Z	0.01	$\mu l/cell\ t^{-1}$
Rate of Immune response due to Infected T cells	r	0.004	$\mu l/cell\ t^{-1}$
Death rate of Infected cell due to CTL cells	δ^z	0.02	$\mu l/cell\ t^{-1}$
Natural death rate of CTL cells	b	0.2	t^{-1}

The values of the parameters are from literature referred, see also [34,19].

3 Results

3.1 Dynamics of CTLs on Transition of R5 to X4 Switch

The evidences suggest that CTLs response contributes to control HIV-1 infection *in-vivo* [14,15,16,17,18]. However, diseases progression rate to AIDS varies person to person which indicates that CTLs response plays protective role in progression. Here, we simulate CTLs response during HIV-1 superinfection and

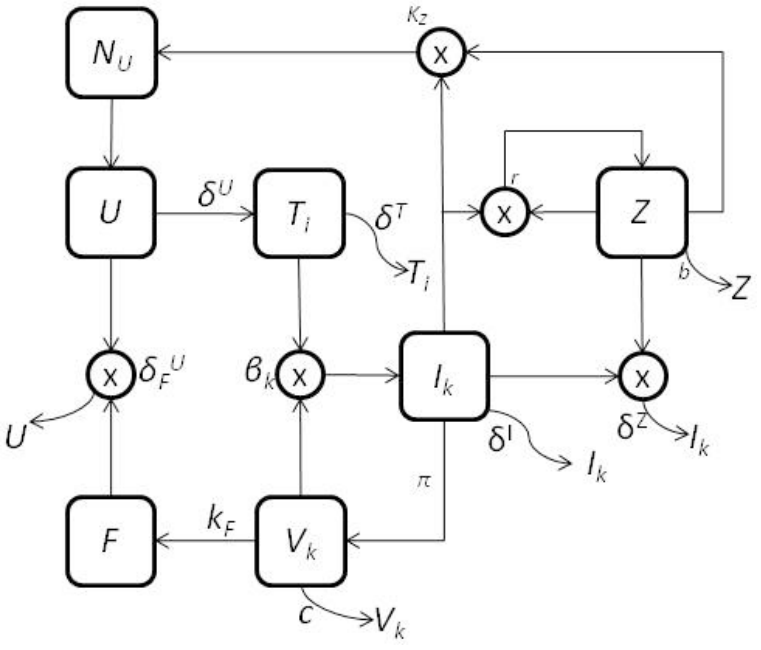


Fig. 1. Schematic description of the updated model for the switching from R5 to X4 viral phenotype with immune response. Naive T-cells U, are generated at constant rate N_U and removed at rate δ^U. Also, Naive T-cells are produced at fix rate k_Z due to CTLs response. They give birth to differentiated, uninfected T-cells, T. These in turn are removed at constant rate δ^T and become infected as they interact with the virus. Infected T-cells, I, die at rate δ^I and contribute to the budding of virus particles, V, that are cleared out at rate c. In addition, infected cells are cleared out at δ^Z by CTLs, Z. CTLs response depends on number of infected cells(I) and is cleared out at fixed rate b. As soon as the X4 phenotype arises, the production of the TNF starts, proportional to the X4 concentration and contributes to the clearance of naive T-cells, via the δ_F^U.

R5 to X4 phenotype switch. We have varied parameter k_Z to analyse the effect of CTLs and infected cells on immature cells production. We have assumed that infected cells are killed by CTLs, at constant rate δ^Z and CTLs response is proportional to the population of infected cells at fixed rate r.

In Figure 2, we varied parameter k_Z to observe the dynamics of viruses and CD4+ cells. Viral load and CD4+ T cells behave normally in presence of weaker immune system, do not make much difference in the appearance of X4 phenotype (see Fig. 2(a)). At $k_Z = 0.1$, immune system tries to resist during initial stage of infection and during appearance of X4 phenotype (see Fig. 2(b)). We observed strong competition among viruses and CD4+ T cells at 0.3 of k_Z (see Fig. 2(c)). We observed that CTLs response plays a key role in controlling HIV-1.

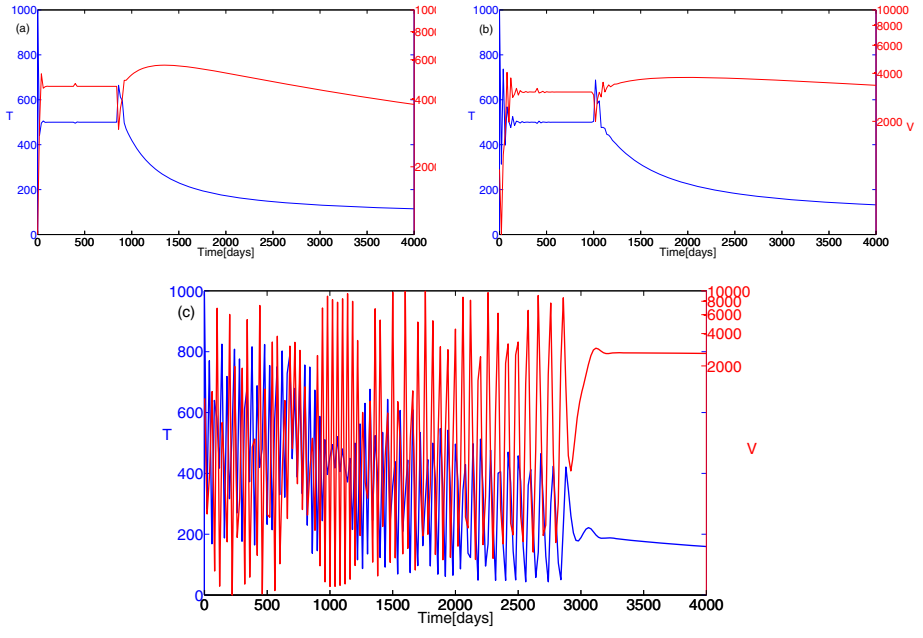


Fig. 2. Effect of immune response on the model due to k_Z parameter: (a) $k_Z = 0.01$, for weak immune system, (b) $k_Z = 0.1$, system shows damped oscillations at initial stage and at appearance of R5 and X4, (c) $k_Z = 0.3$, oscillations are observed between CD4+ T cells and viral load at initial stage and they become stable at later stage of infection. Mutation rate $\mu = 2.5 \times 10^{-3}$.

3.2 The Key Drug Therapies: HAART, Maraviroc, ZFN

HAART Therapy. HAART therapy is composed of multiple anti-HIV drugs and is prescribed to many HIV-positive patients. The therapy usually includes one nucleoside analog (DNA chain terminator), one protease inhibitor and either a second nucleoside analog ("nuke") or a non-nucleoside reverse transcription inhibitor (NNRTI) [37]. HAART is one of the way suppressing viral replication in the blood while attempting to prevent the virus rapidly developing resistance to the individual drug. Our model in the Section 3.1 suggests that phenotype switch depends on mutation rate μ and increasing production of immature T cells due to CTLs response parameter k_Z. To simulate HAART therapy on the model, we have decreased viral load at certain time interval to analyse drug effect.

Fig. 3 describes HAART treatment, which is usually able to decrease the concentration of the virus in the blood and delay the appearance of X4. However, this model suggests a possible scenario in case of a sudden interruption in the therapy. If the different R5 strains experience the same selection pressure, as soon as the therapy is stopped, the X4 strain may appear sooner. In fact during the treatment, concentration of the different strains of R5 viruses is kept to a

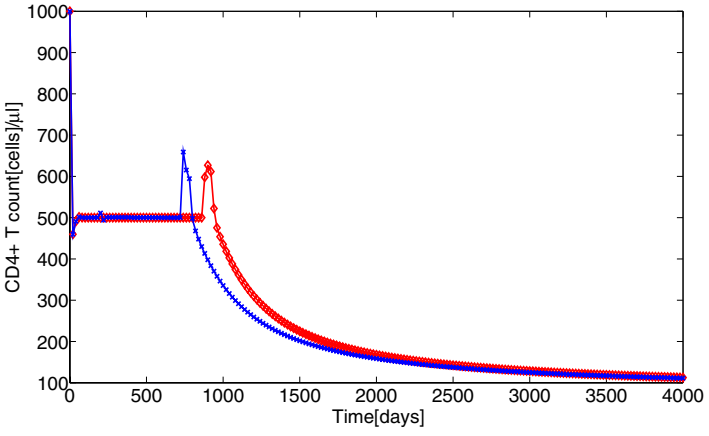


Fig. 3. Model outcome with HAART therapy: CD4+ T cells concentration during HIV-1 superinfection by R5 viral strain. Drug treatment simulated from time $t = 200$ to $t = 400$. For superinfection R5 to X4 phenotype switch, the time is shorter in drug treatment as compared to the time without drug treatment. The concentration of CD4+ T cells is plotted using red line ($\diamond$) and blue line ($\times$) denoting with drug therapy and without drug therapy respectively. Parameters are same as in Fig. 2.

very low level while T-cell abundances increase. As the therapy is interrupted, all the strains give rise to a renewed infection, but now also the strains closer to the X4 co-receptor using viruses are populated, and a mutation leading to an X4 strain occurs sooner.

Maraviroc. Maraviroc is the first member of a new class of antiretroviral medications, the CCR5-receptor antagonists which is approved by the US Food and Drug Administration (FDA). It targets host protein, a CCR5 coreceptor rather than viral strains so it can block viral strains entering into CD4+ T cells. Maraviroc can be used as combination with other antiretroviral agents in treatment-experienced patients infected with multidrug-resistant, CCR5 tropic HIV-1. The study conducted on Maraviroc has shown promising results compared to other antiretroviral agents (placebo). Clinical trial phase III studies shown that Marvarioc, as compared with placebo, resulted in greater suppression in HIV-1 and greater increase in CD4+ T cells count at 48 weeks [35,36].

In Fig. 4, we considered Maraviroc treatment, which is usually able to decrease viral load in R5 tropic phase and greater increase of CD4+ T count. To simulate Maraviroc treatment we mutate CD4+ T cells with the effect of drug(Maraviroc) so that R5 virus particles can not interact with them. Drug treatment is simulated from time $t = 200$ to $t = 400$. The model suggests that after discontinuing the treatment, viral load increases again hence concentration of CD4+ T cells decreases. However, there is no time delay in appearance of X4 viral strain.

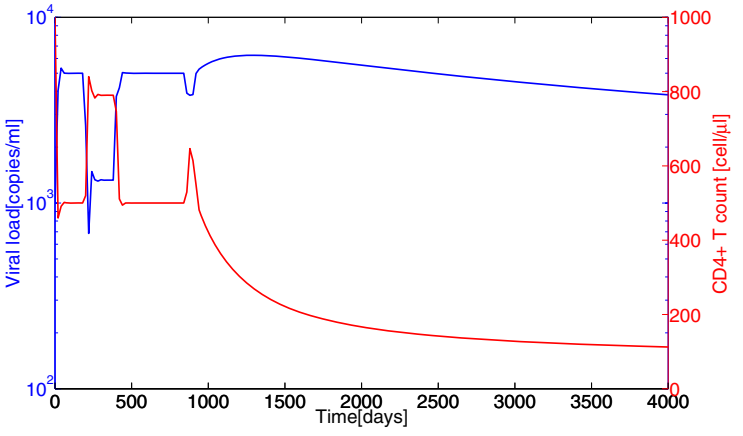


Fig. 4. Maraviroc therapy dynamics: Dynamics of CD4+ T counts and Viral load during drug therapy. Simulation of treatment from time $t = 200$ to $t = 400$. Parameters are same as in Fig. 2.

R5 to X4 switch and Zinc-finger nucleases(ZFN) therapy. Several different anti-HIV-1 gene therapy approaches have been tested in cells over the last 10 years. This therapy is mainly classified into two categories (i) RNA-based agent; (ii) protein-based agents [38]. Zinc-finger nucleases use different protein-based approach. Disruption in gene sequence by ZFN therapy, generates nonfunctional CCR5 mutant, known as $CCR5\Delta32$ which is resistant to CCR5-tropic of HIV-1. Recent study on a mouse model has shown that ZFN-modified CD4+ T cells had increased number of CD4+ T cells and a statistically significant seven-fold reduction in viral load in their peripheral blood [39].

In Fig. 5, we simulate CD4+ T cells and viral load dynamics by considering several CD4+ T mutation percentage, introduced at time $t = 200$, using ZFN gene therapy. We observed substantial increase in concentration of CD4+ T cells and a

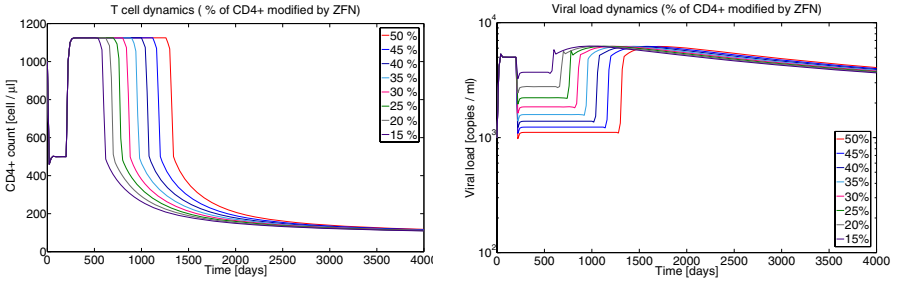


Fig. 5. Model outcome with Zinc-finger nucleases therapy: percentage of CD4+ T cells mutated by ZFN gene therapy (left)CD4+ T cells count(in cells per microliter) (right) viral load (in copies per milliliter)

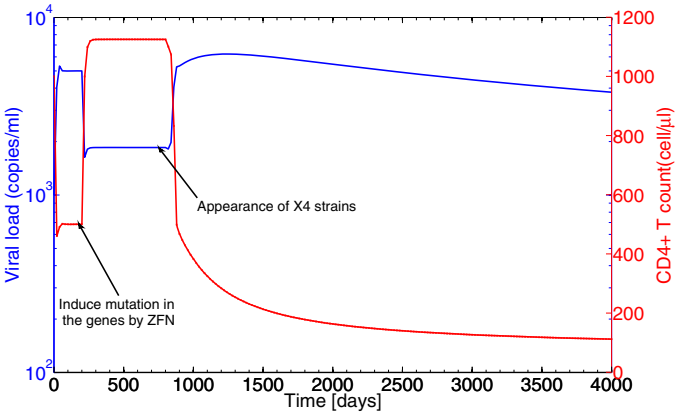


Fig. 6. Snapshots of competitive dynamics between CD4+ T cells and viral load at 30% mutated CD4+ T cells by ZFN therapy

great decrease in viral load(see Fig. 5(a),Fig. 5(b)). In addition, R5 to X4 phenotype switch appears later as a result of increase in percentage of mutated CD4+ T cells by ZFN(see Fig. 5(a),Fig. 5(b)). Fig. 6 represents dynamics of CD4+ T cells and viral load during HIV-1 infection using ZFN treatment. Our model suggests that ZFN gene therapy may affect R5 to X4 phenotype switch that appears later as we increase percentage of mutated CD4+ T cells in the system.

4 Conclusion

There are several strains of HIV present worldwide and often HIV patients carry multiple strains simultaneously in their body. During the HIV infection several quasispecies of the virus arise which are able to use different chemokine coreceptors particularly CCR5 and CXCR4.

Here, we represent the model of CTLs response during evolutionary quasispecies superinfection of HIV-1, especially transition of R5 to X4 phenotype that helps to gain better insight of the complexity of the HIV-1 infection process. We observed that CTLs response is a key aspect in HIV infection. However, viruses escape from immune system by mutation. Simulation results show different behaviors of the system by varying CTLs response parameter.

There are many options available for drug therapy in the treatment of HIV infection such as Highly Active Antiretroviral Therapy, Maraviroc and gene therapy(Zinc-finger nucleuses). We noticed, interruption and discontinuation of HAART therapy may result into early appearance of R5 to X4 phenotype switch. Maraviroc therapy can substantially decrease viral load and significantly increase CD4+ T cells count during R5-tropic phase. But, when viruses start to enter through CXCR4 chemokine coreceptor, we can not see any effect of Maraviroc on transition of the R5 to X4 phenotype switch. Zinc-finger nucleuses gene

therapy may become helpful for HIV patients because our results on ZFN has shown that emergence of R5 to X4 phenotype switch depends on CCR5 mutated by ZFN (see Fig. 5 and Fig. 6).

Our model also represents a general framework to investigate switching dominance of strains and arising new dominant strains during different phases of therapies.

Acknowledgments. This project is supported by EC IST SOCIALNETS - Grant agreement number 217141.

References

1. Nagashima, K.A., Thompson, D.A., Rosenfield, S.I., et al.: Human Immunodeficiency virus type 1 entry inhibitors PRO 542 and T-20 are potently synergistic in blocking virus-cell and cell-cell fusion. J. Infect. Dis. 183, 1121–1125 (2001)
2. Karlsson, I., Antonsson, L., Shi, Y., Oberg, M., Karlsson, A., Albert, J., Olde, B., Owman, c., Jansson, B., Fenyo, E.M.: Coevolution of RANTES sensitivity and mode of CCR5 receptor use by human immunodeficiency virus type 1 of the R5 phenotype. J. Virol. 78, 11807–11815 (2004)
3. Gorry, P.R., Churchill, M., Crowe, S.M., Cunningham, A.L., Gabuzda, D.: Pathogenesis of macrophage tropic HIV. Curr. HIV Res. 3, 53–60 (2005)
4. Koot, M., Keet, I.P., Vos, A.H., de Goede, R.E., Roos, M.T., Coutinho, R.A., Miedema, F., Schellekens, P.T., Tersmette, M.: Prognostic value of HIV-1 syncytium-inducing phenotype for rate of CD4+ cell depletion and progression to AIDS. Ann. Intern. Med. 118, 681–688 (1993)
5. Davenport, M., Zaunders, J., Hazenberg, M., Schuitemaker, H., Rij, R.: Cell turnover and cell tropism in HIV-1 infection. Trends Microbiol. 10, 275–278 (2002)
6. Gray, L., Sterjovski, J., Churchill, M., Ellery, P., Nasr, N., Lewin, S.R., Crowe, S.M., Wesselingh, S.L., Cunningham, A.L., Gorry, P.R.: Uncoupling coreceptor usage of human immunodeficiency virus type 1 (HIV-1) from macrophage tropism reveals biological properties of CCR5-restricted HIV-1 isolates from patients with acquired immunodeficiency syndrome. Virology 337, 384–398 (2005)
7. Herbeuval, J.P., Hardy, A.W., Boasso, A., Anderson, S.A., Dolan, M.J., Dy, M., Shearer, G.M.: Regulation of TNF-related apoptosis-inducing ligand on primary CD4+ T cells by HIV-1: Role of type I IFN-producing plasmacytoid dendritic cells. Proc. Nat. Acad. Sci. USA 102, 13974–13979 (2005)
8. Pavlakis, G.N., Valentin, A., Morrow, M., Yarchoan, R.: Differential effects of TNF on HIV-1 expression: R5 inhibition and implications for viral evolution. In: Int. Conf. AIDS 2004, July 11-16, 15:(abstract no. MoOrA1048) (2004)
9. Eigen, M., McCaskill, J., Schuster, P.: The Molecular Quasi-Species. Adv. Chem. Phys. 75, 149–263 (1989)
10. Sguanci, L., Bagnoli, F., Li, P.: Modeling viral coevolution: HIV multi-clonal persistence and competition dynamics. Physica A 366, 333–346 (2006)
11. Biebricher, C.K., Eigen, M.: The error threshold. Virus Res. 107, 117–127 (2005)
12. Neumann, A.U., Lam, N.P., Dahari, H., Gretch, D.R., Wiley, T.E., Layden, T.J., Perelson, A.S.: Hepatitis C Viral Dynamics in Vivo and the Antiviral Efficacy of Interferon-alpha Therapy. Science 282, 103–107 (1998)
13. Eigen, M., Schuster, P.: The hypercycle. Naturwissenschaften 64, 541–565 (1977)

14. Carrington, M., OBrien, S.J.: The influence of HLA genotype on AIDS. Annu. Rev. Med. 54, 535–551 (2003)
15. Kalams, S.A., Goulder, P.J., Shea, A.K., Jones, N.G., Trocha, A.K., et al.: Levels of human immunodeficiency virus type 1-specific cytotoxic T-lymphocyte effector and memory responses decline after suppression of viremia with highly active antiretroviral therapy. J. Virol. 73, 6721–6728 (1999)
16. Borrow, P., Lewicki, H., Wei, X., Horwitz, M.S., Peffer, N., et al.: Antiviral pressure exerted by HIV-1specific cytotoxic T lymphocytes (CTLs) during primary infection demonstrated by rapid selection of CTL escape virus. Nat. Med. 3, 205–211 (1997)
17. Phillips, R.E., Rowland-Jones, S., Nixon, D.F., Gotch, F.M., Edwards, J.P., et al.: Human immunodeficiency virus genetic variation that can escape cytotoxic T cell recognition. Nature 354, 453–459 (1991)
18. Jin, X., Bauer, D.E., Tuttleton, S.E., Lewin, S., Gettie, A., et al.: Dramatic rise in plasma viremia after CD8() T cell depletion in simian immunodeficiency virusinfected macaques. J. Exp. Med. 189, 991–998 (1999)
19. Althaus, C.L., Be Boer, R.J.: Dynamics of Immune escape during HIV/SIV Infection. PLOS Computational Biology 4(7), e1000103 (2008)
20. Ho, D., Neumann, A.U., Perelson, A.S., Chen, W., Leonard, J.M., Markowitz, M.: Rapid turnover of plasma virions and CD4 lymphocytes in HIV-1 infection. Nature 373, 123–126 (1995)
21. Chao, L., Davenport, M.P., Forrest, S., Perelson, A.S.: A stochastic model of cytotoxic T cell responses. J. Theor. Biol. 228, 227–240 (2004)
22. Celada, F., Seiden, P.E.: Affinity maturation and hypermutation in a simulation of the humoral immune response. Eur. J. Immunol. 26, 1350–1358 (1996)
23. De Boer, R.J., Perelson, A.S.: Towards a general function describing T-cell proliferation. J. Theor. Biol. 175, 567–576 (1995)
24. Wodarz, D., Nowak, M.A.: HIV dynamics and evolution. BioEssays 24, 1178–1187 (2002)
25. Perelson, A.S., Neumann, A.U., Markowitz, M., Leonard, J.M.D., Ho, D.: HIV-1 Dynamics in Vivo: Virion Clearance Rate, Infected Cell Life-Span, and Viral Generation Time. Science 271, 1582–1586 (1996)
26. Wiegel, F.W., Perelson, A.S.: Some scaling principles for the immune system. Immunol. Cell. Biol. 82, 127–131 (2004)
27. Wei, X., Ghosh, S.K., Taylor, M.E., Johnson, V.A., Emini, E.A., Deutsch, P., Lifson, J.D., Bonhoeffer, S., Nowak, M.A., Hahn, B.H., Saag, M.S., Shaw, G.M.: Viral dynamics in human immunodeficiency virus type 1 infection. Nature 373, 117–122 (1995)
28. Sguanci, L., Bagnoli, F., Li, P.: Modeling HIV quasispecies evolutionary dynamics. BMC Evolutionary Biology 7, S5 (2007)
29. Bruch, C.L., Chao, L.: Evolvability of an RNA virus determined by its mutational neighborhood. Nature 406, 625–628 (2000)
30. Riddell, S.R., Watanabe, K.S., Goodrich, J.M., Li, C.R., Agha, M.E., Greenberg, P.D.: Restoration of viral immunity in immunodeficient humans by the adoptive transfer of T cell clones. Science (Wash. DC) 257, 238–241 (1992)
31. Cannon, M.J., Openshaw, P.J., Askonas, B.A.: Cytotoxic T cells clear virus but augment lung pathology in mice infected with respiratory syncytial virus. J. Exp. Med. 168, 1163–1168 (1988)
32. Kaslow, R., Carrington, M., Apple, R., Park, L., Munoz, A., et al.: Influence of combinations of human major histocompatibility complex genes on the course of HIV-1 infection. Nat. Med. 2, 405–411 (1996)

33. Schmitz, J., Kuroda, M., Santra, S., Sasseville, V., Simon, M., et al.: Control of viremia in simian immunodeficiency virus infection by CD8+ lymphocytes. Science 283, 857–860 (1999)
34. Asquith, B., Edwards, C.T.T., Lipsitch, M., McLean, A.R.: Inefficient Cytotoxic T LymphocyteMediated Killing of HIV-1 Infected Cells In Vivo. PLOS Biology 4(4), 90 (2006)
35. Gulick, R.M., Lalezari, J., James, G.J., et al.: Maraviroc for Previously Treated Patients with R5 HIV-1 Infection. The New England Journal of Medicine 359, 14 (2008)
36. Lieberman-Blum, S.S., Fung, H.B., Bandres, J.C.: Maraviroc: A CCR5-Receptor Antagonist for the Treatment of HIV-1 Infection. Clinical Therapeutics 30, 7 (2008)
37. Dybul, M., Fauci, A.S., Bartlett, J.G., Kaplan, J.E., Pau, A.K.: Panel on Clinical Practices for Treatment of HIV (September 2002). "Guidelines for using antiretroviral agents among HIV-infected adults and adolescents". Ann. Intern. Med. 137(5 Pt 2), 381–433 (2002)
38. Rossi, J.J., June, C.H., Kohn, D.B.: Genetic therapies against HIV: approximate methods. Nature Biotechnology 25, 1444–1454 (2007)
39. Perez, E.E., Wang, J., Miller, J.C., Jouvenot, Y., Kim, K.A., Liu, O., et al.: Establishment of HIV-1 resistance in CD4+ T cells by genome editing using zinc-finger nucleases. Nature Biotechnology 26, 808–816 (2008)

Exploration of the Dendritic Cell Algorithm Using the Duration Calculus

Feng Gu, Julie Greensmith, and Uwe Aickelin

School of Computer Science, University of Nottingham
Nottingham, NG8 1BB, UK
{fxg,jqg,uxa}@cs.nott.ac.uk

Abstract. As one of the newest members in Artificial Immune Systems (AIS), the Dendritic Cell Algorithm (DCA) has been applied to a range of problems. These applications mainly belong to the field of anomaly detection. Real-time detection, a new challenge to anomaly detection, requires improvement on the real-time capability of the DCA. To assess such capability, formal methods in the research of real-time systems can be employed. The findings of the assessment can provide guideline for the future development of the algorithm. Therefore, in this paper we use an interval logic based method, named the Duration Calculus (DC), to specify a simplified single-cell model of the DCA. Based on the DC specifications with further induction, we find that each individual cell in the DCA can perform its function as a detector in real-time. Since the DCA can be seen as many such cells operating in parallel, it is potentially capable of performing real-time detection. However, the analysis process of the standard DCA constricts its real-time capability. As a result, we conclude that the analysis process of the standard DCA should be replaced by a real-time analysis component, which can perform periodic analysis for the purpose of real-time detection.

1 Introduction

Artificial Immune Systems (AIS) [3] are computer systems inspired by both theoretical immunology and observed immune functions, principles and models, which can be applied to real world problems. As the natural immune system is evolved to protect the body from a wealth of invading micro-organisms, *artificial* immune systems are developed to provide the same defensive properties within a computing context. One of these immune inspired algorithms called the Dendritic Cell Algorithm (DCA) [6] is based on the function of the dendritic cells of the innate immune system. An abstract model of the behaviour of natural dendritic cells is used as the foundation of the developed algorithm. Currently, the DCA has been applied to numerous problems, including port scan detection [6], Botnet detection [1] and a classifier for robotic security [12]. These applications refer to the field of anomaly detection, which involves discriminating between normal and anomalous data, based on the knowledge of the normal data. The success of the applications has suggested that the DCA shows not only good performance on detection rate, but also the ability to reduce the rate of false alarms

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 54–66, 2009.

in comparison to other systems including Self Organising Maps [7]. However, one problem with the DCA has been pointed out in [9], that is, the analysis process of the algorithm is performed offline rather than online in real-time. This results the delays between when potential anomalies initially appear and when they are correctly identified. Such delays can be problematic for applications with strict time constraints, as they are often speed-critical. To solve this problem, it is desired to improve the real-time capability of the DCA, in order to develop an effective real-time detection system.

A *real-time system* [14] is a reactive system which, for certain inputs, has to compute the corresponding outputs within given time bounds (real-time criteria). The design of real-time systems generally requires high precision due to their particular application areas. The high precision is achieved by using formal methods that are based on the mathematical models of the systems being designed. The formal methods make it possible to specify the system properties at different levels and abstractions, as well as formally verify the specifications before implementing. One of the formal methods for specifying real-time systems is known as the Duration Calculus (DC) [17], which is a temporal logic and calculus for describing and reasoning about the properties of a real-time system over time intervals. The DC can specify the safety properties, bounded responses and duration properties of a real-time system, which can be logically verified through proper induction. Unlike predicate calculus [5] using time points to express time-dependent state variables or observables of the specified system, the DC uses time intervals with the focus on the implicit semantics level rather than the explicit syntactic level. As a result, it is more convenient and concise to use the DC to specify patterns or behaviour sequences of a real-time system over time intervals, compared to predicate calculus.

The real-time capability of the DCA should be assessed before making any improvement on the algorithm. For this purpose, the DC is used to specify the behaviours of the DCA over particular time intervals. First of all, the DC specifications of the algorithm can be further induced by applying available proof rules in the DC. The mathematical aspects of the algorithm that have not been discovered might be revealed through the induction. In addition, the DC specifications include the temporal properties of the algorithm, which provide the insight of the duration required for each individual behaviour. The duration of each behaviour can be compared with the real-time criteria, to evaluate whether a behaviour can be performed in real-time or not. As a result, we can identify the properties of the algorithm that can satisfy the real-time criteria and those cannot at the behavioural level. The findings can be used as the basis for the further development of the real-time detection system based on the DCA.

The aim of this paper is to use the DC to specify the properties of the DCA, with the focus on the development of an effective real-time detection system. As a result, we would be able to identify the properties of the DCA that can be inherited for future development, and those need improved. Proper proof is included in this paper to support the conclusions derived from the DC specifications. The paper is organised as follows: the DCA is briefly described in section 2; the

background information of the DC is given in section 3; the DC specifications of the single-cell model are given in section 4; the discussion of the analysis process of the DCA is provided in Section 5; finally the conclusions and future work are drawn in Section 6.

2 The Dendritic Cell Algorithm

2.1 Algorithm Overview

As previously stated the blueprint for the DCA is the function of the dendritic cells of the innate immune system. Natural dendritic cells are capable of combining a multitude of molecular information and then interpret this information for the adaptive immune system, to induce appropriate immune responses towards perceived threats. Signal and antigen are the two types of molecular information processed by dendritic cells. Signals are collected from their local environment and consist of indicators of the health of the monitored tissue. Denrditic cells exist in one of three states of maturation to perform their immune function. In their initial immature state, dendritic cells are exposed to a combination of signals. They can differentiate into either semimature or fully mature state based on the concentrations of signals. Additionally, during their immature phase dendritic cells also collect debris in the tissue which are subsequently combined with the molecular environmental signals. Some of the debris collected are termed antigens, and are proteins originating from potential invading entities. Eventually dendritic cells combine evidence in the form of signals with the 'suspect' antigens to correctly instruct the adaptive immune system to respond, or become tolerant to the antigens in question. For more detailed information of natural dendritic cells, please refer to Lutz and Schuler [10].

The resulting algorithm incorporates the state transition pathway, the environmental signal processing procedure, and the correlation between signals and antigens. In the algorithm signals are represented as continuous real-number values and antigens are the categorical values of possible categories. The algorithm is based on a multi-agent framework, where each cell processes its own environmental signals and collects antigens. Diversity is generated within the cell population through the application of a 'migration threshold' - this value limits the number of signal instances an individual cell can process during its lifespan. This creates a variable time window effect, with different cells processing the signal and antigen input streams over a range of time periods [13]. The combination of signal/antigen correlation and the dynamics of a cell population are responsible for the detection capabilities of the DCA.

2.2 The Single-Cell Model

The DCA consists of a population of artificial cells, each of which is capable of performing a set of identical behaviours, to accomplish its function as a detector. In order to understand the properties of the algorithm, we start with describing a simplified single-cell model. A one-cell model is demonstrated in [13] for

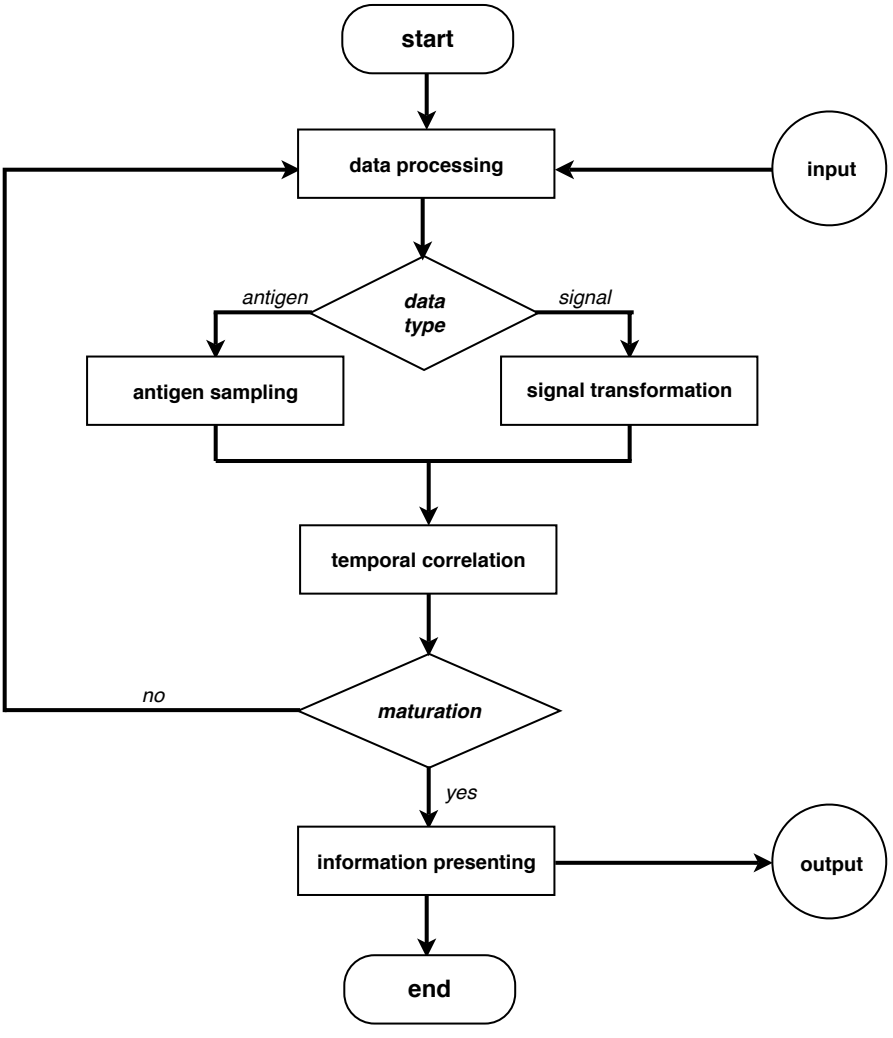


Fig. 1. The behavioural flowchart of the single-cell model

the purpose of analysing the effect of signal frequency. Whereas, the single-cell model here is focused on the behavioural level of a generic cell of the DCA from a temporal perspective, rather than a quantitative level. The flowchart of the behaviours involved in the single-cell model is displayed in Fig. 1. The time-dependent behaviours of a cell are termed 'events' in this paper, and they are performed by the cell in each state during particular time intervals. The state and event mentioned in this paper are similar to those defined in temporal logic. Therefore, states must hold over any subintervals of an interval in which they hold, conversely events do not hold over any subintervals of an interval in which

they hold. In other words, states can be broken down over multiple subintervals of an interval, whereas events cannot. The states, the events in each state, and the relevant time intervals of the single-cell model are described as following.

- **Immature state:** this is the initial state of the cell, where the cell is fed with input data instances. All the input data instances are handled by the *data processing* event, to determine their types. If the type of a data instance is 'signal', it is passed to the *signal transformation* event. Otherwise, if the type of data instance is 'antigen', the data instance is passed to the *antigen sampling* event. In each iteration of the system, only one signal instance but multiple antigen instances can be fed to the cell. The processed signals and sampled antigens are correlated by the *temporal correlation* event according to their time stamps. The cell keeps performing the events above cyclically, until the migration threshold is reached. This indicates that the cell has acquired sufficient information for decision making.
- **Matured state:** once the cell reaches its migration threshold, it changes from immature state to either semimature state or fully mature state. As same event takes place in both semimature state and mature state, they are called 'matured state' in this paper. Based on the correlated signals and antigens by the *temporal correlation* event, the cell makes a decision on whether any potential anomalies appeared within the input data. Such decision is termed 'processed information' that is presented by the *information presenting* event as the output of the cell. Up to this point one lifespan of the cell is finished, and then the cell is reinitialised to immature state for new incoming data instances.

At the population level, the DCA can be seen as a system in which multiple single-cell models are executed in parallel. The output of each matured cell is accumulated with the outputs of others in the population by the analysis process of the algorithm. From the accumulated outputs of all matured cells, the analysis process produces the final detection result in which the anomalies within the input data can be identified. In the standard DCA, the analysis process is performed after all instances of the input data are processed. This could make it difficult for the system to satisfy the real-time criteria if the size of the input data is large. The details will be discussed in section 5.

3 The Duration Calculus

The DC was firstly introduced by Zhou and Hansen [16] as an extension of the Interval Temporal Logic [11]. It uses continuous time for specifying desired properties of a real-time system without considering its implementation. The specifications are presented by DC formulas which express the behaviours of time-dependent variables or observables of a real-time system within certain time intervals. In DC specifications, not only abstract high-level but also detailed low-level specifications can be formulated according to the selected variables or observables. This makes it possible to specify the system from different perspectives at various levels. There are different versions of the DC [17], including the

classic DC, the extended DC and mean-value calculus. The work in this paper uses the classic DC, as it is sufficient for specifying the system presented.

In order to introduce the DC, the syntax defining the structure of DC specifications and the semantics explaining its meaning are described in this section. The DC specifications often consist of three elements, which are state assertions, terms and formulas. Their formal definitions given in [14] are as following.

Definition 1. *state assertions* are Boolean combinations of basic properties of state variables, as defined in 1.

$$P ::= 0 \mid 1 \mid X = d \mid \neg P \mid P_1 \wedge P_2 \tag{1}$$

As a state assertion, the Boolean value of the observable of P can be either 0 or 1; It can have a state variable X whose value is d of data type D; There are situations where P does not hold; There are also situations where the substates of P, P_1 and P_2, both hold. The semantics of a state assertion involves the interpretation of time-dependent variables that occur within it. Let $\mathcal{I}$ be an interpretation, the semantics of a state assertion P is a function defined in 2.

$$\mathcal{I}[\![P]\!] : \mathsf{Time} \longrightarrow \{0, 1\} \tag{2}$$

where 0 or 1 represents the Boolean value of P at $t \in \mathsf{Time}$, which can be also written as $\mathcal{I}(P)(t)$.

Definition 2. *terms* are expressions that denote real numbers related to time intervals, as defined in 3.

$$\theta ::= x \mid l \mid \int P \mid f(\theta_1, ..., \theta_n) \tag{3}$$

The expression above states that giving an interval l during which the state assertion P holds, there is a global variable x that is related to the valuation a n-ary function f. The semantics of a term depends on the interpretation of state variables of the state assertion, the valuation of the global variables, and the given time interval. The semantics of a term θ is defined in 4.

$$\mathcal{I}[\![\theta]\!] : \mathsf{Val} \times \mathsf{Intv} \longrightarrow \mathbb{R} \tag{4}$$

where Val stands for the valuation ($\mathcal{V}$) of the global variables, and Intvl is the given interval which can be defined in 5.

$$\mathsf{Intv} \stackrel{\text{def}}{\Longleftrightarrow} \{[b, e] \mid b, e \in \mathsf{Time} \text{ and } b \leq e\} \tag{5}$$

So this term can also be written as $\mathcal{I}[\![\theta]\!](\mathcal{V}, [b, e])$.

Definition 3. *formulas* describe properties of observables depending on time intervals, as defined in 6

$$F ::= p(\theta_1, ..., \theta_n) \mid \neg F_1 \mid F_1 \wedge F_2 \mid \forall x \bullet F_1 \mid F_1 \ ; \ F_2 \tag{6}$$

This expression shows that there is a n-ary predicate with the terms of $\theta_1, ..., \theta_n$ defined in the interval of l, during which F_1 does not hold or F_1 and F_2 hold. The quantitative part of the expression is separated by the symbol of '•'. It states that for all x, F_1 holds in the interval of l, or there are situations where F_1 and F_2 hold respectively in the subintervals of l. The symbol '**;**' is the chop operator used for dividing the given time interval into subintervals. The semantics of a formula involves an interpretation of the state variables, a valuation of the global variables and a given time interval, defined in 7. The relevant state variables and global variables all appear in the terms of this formula.

$$\mathcal{I}[\![F]\!] : \mathsf{Val} \times \mathsf{Intv} \longrightarrow \{\mathsf{tt}, \mathsf{ff}\} \tag{7}$$

where tt stands for true and ff for false. It can also be written as $\mathcal{I}[\![F]\!](\mathcal{V}, [b, e])$, which stands for the truth value of F under the interpretation $\mathcal{I}$, the valuation $\mathcal{V}$, and the interval $[b, e]$.

4 DC Specifications of the System

Before going into the details of the DC specifications, we want to introduce the notations that are used in this section, listed as following.

- $I : \mathsf{Time} \longrightarrow \{0, 1\}$ is the Boolean observable indicating that the cell is in immature state.
- $M : \mathsf{Time} \longrightarrow \{0, 1\}$ is the Boolean observable indicating that the cell is in matured state.
- $E_i : \mathsf{Time} \longrightarrow \{0, 1\}$ is the Boolean observable representing the ith event is being performed, where $i \in \mathbb{N}$.
- $l_i \in \mathbb{R}$ is the duration of E_i.
- $l_a \in \mathbb{R}$ is the duration of the analysis process.
- $b \in \mathbb{R}$ is the real-time bound, if a process is completed within b, then it can be performed in real-time, and vice versa.
- $r \in \mathbb{R}$ is the duration of one iteration in the system.
- $c \in \mathbb{R}$ is the duration of one lifespan during which the cell experiences both immature state and matured state.
- $\bar{m} \in \mathbb{N}$ is the average number of processed signal instances within one lifespan of the cell.
- $\bar{n} \in \mathbb{N}$ is the average number of sampled antigen instances within one lifespan of the cell.

To be more specific, the definition of each event E_i is shown as follows.

- E_1 is the *data processing* event with an interval l_1.
- E_2 is the *signal transformation* event with an interval l_2.
- E_3 is the *antigen sampling* event with an interval l_3.
- E_4 is the *temporal correlation* event with an interval l_4.
- E_5 is the *information presenting* event with an interval l_5.

4.1 Specifications of the Single-Cell Model

According to the description of the single-cell model in section 2, the cell performs a set of particular events in each state. So the states of the cell can be indicated by the combination of whether E_i is being performed (the Boolean observable of E_i), as shown in 8.

$$\begin{aligned} I &::= 0 \mid 1 \mid \neg I \mid E_1 \vee (E_2 \wedge \neg E_3) \vee (\neg E_2 \wedge E_3) \vee E_4 \\ M &::= 0 \mid 1 \mid \neg M \mid E_5 \end{aligned} \tag{8}$$

In immature state (I), the cell is fed with input data instances whose type can be either signal or antigen. The immature state can be indicated by E_1 holds, $E_2 \wedge \neg E_3$ holds, $\neg E_2 \wedge E_3$ holds, or E_4 holds. Conversely, in matured state (M), the cell presents the processed information from correlated signals and antigens. The matured state can be indicated by E_5 holds.

The specifications in 8 can be expanded by including the time interval of each event, expressed in the form of formulas, in which the temporal dependencies between events are included. For example, E_2 or E_3 depends on the completion of E_1, and only either of them can be performed at one point; E_4 depends on the completion of E_2 and E_3, as the temporal correlation requires both processed signals and sampled antigens; E_5 is performed as soon as the cell changes to matured state, it is not dependent on any other events. Two formulas that are corresponding to the immature state and matured state of a cell are shown in 9.

$$\begin{aligned} F_1 &::= \lceil I \rceil \mid \neg E_5 \mid (E_1 \;;\; E_2 \wedge \neg E_3) \;;\; (E_1 \;;\; \neg E_2 \wedge E_3) \;;\; E_4 \\ F_2 &::= \lceil M \rceil \mid \neg(E_1 \vee E_2 \vee E_3 \vee E_4) \mid E_5 \end{aligned} \tag{9}$$

where $\lceil I \rceil$ stands for that I holds almost everywhere within the time interval constrained by formula F_1, and $\lceil M \rceil$ stands for that M holds almost everywhere within the time interval constrained by F_2. So in the interval constrained by F_1, it is certain that E_5 does not hold. This interval can be divided into multiple subintervals in which E_1, $E_2 \wedge \neg E_3$, $\neg E_2 \wedge E_3$, or E_4 holds respectively. In the interval constrained by F_2, none of E_1, E_2, E_3 or E_4 holds, but only E_5 holds. To illustrate this, assuming the overall length of the time interval while F_1 and F_2 holds is six, and the time interval of each event is equal to one, Fig. 2 shows the interpretation of the two formulas.

As the cell can process multiple signal instances and antigen instances before it gets matured, some of the events can be performed for more than once within one lifespan of the cell. To generalise this, the average numbers ($\bar{m}$ and $\bar{n}$) of processed data instances are used. Therefore, the pattern of '$E_1 \;;\; E_2 \wedge \neg E_3$' appears $\bar{m}$ times, and the pattern of '$E_1 \;;\; \neg E_2 \wedge E_3$' appears $\bar{n}$ times. Additionally, E_4 is performed $\bar{m}$ times, as the number of performed temporal correlations is equivalent to the number of processed signal instances. However, E_5 is only performed once, as it is irrelevant to the number of processed signal instances or

sampled antigen instances. As a result, the duration of a cell being in immature state and the duration of a cell being in matured state can be formalised as in 10. The duration of one lifespan of the cell is defined as $c = \int I + \int M$.

$$\begin{aligned} \int I &= \bar{m} \cdot (l_1 + l_2) + \bar{n} \cdot (l_1 + l_3) + \bar{m} \cdot l_4 \\ \int M &= l_5 \end{aligned} \tag{10}$$

4.2 Evaluation of the Real-Time Capability

Based on the DC specifications above, we conduct a test to examine the real-time capability of an individual cell of the DCA. If the cell completes at least one lifespan within the given real-time bound (b), it suggests that the cell can perform its function in real-time. This test is formalised as a requirement in 11.

$$\mathsf{Req} \overset{\text{def}}{\Longleftrightarrow} \Box(b \geq (\bar{m} + 1) \cdot r \Longrightarrow \int I + \int M \leq b) \tag{11}$$

where $\Box$ is the dual modal operator of interval logic, defined as $\Box F$ holds in an interval of $[b, e]$ only if F holds in every subinterval of $[b, e]$. The *condition* of the Req is the left side of the logical connective '$\Longrightarrow$', while the *conclusion* is on the right side. If this requirement is satisfied, then we conclude that each individual cell in the DCA is capable of operating in real-time.

As mentioned in section 2, a cell exists in either immature state or matured state, and all the events within each state should be performed in each iteration. According to the definition, one system iteration is equal to the duration between two successive updates of signal instance. In each iteration the cell processes one signal instance but a number of antigen instances. Therefore, the cumulative duration of E_1, E_2 and E_4 should not be greater than the duration of one iteration, and the duration of E_5 should also not be greater than the duration of one iteration. Such properties can be formalised as two design decisions of the single-cell model shown in 12.

$$\begin{aligned} \mathsf{Des\text{-}1} &\overset{\text{def}}{\Longleftrightarrow} \Box(\lceil I \rceil \Longrightarrow l_1 + l_2 + l_4 \leq r) \\ \mathsf{Des\text{-}2} &\overset{\text{def}}{\Longleftrightarrow} \Box(\lceil M \rceil \Longrightarrow l_5 \leq r) \end{aligned} \tag{12}$$

The two design decisions are the extra preconditions that determine whether the system can satisfy the real-time criteria or not, as defined in **Theorem 1**.

Theorem 1. $\models (\mathsf{Des\text{-}1} \wedge \mathsf{Des\text{-}2}) \Longrightarrow \mathsf{Req}$
It expresses that if both design decisions Des-1 and Des-2 hold, the requirement Req can be satisfied.

Proof

$$
\begin{aligned}
& b \geq (\bar{m}+1)\cdot r \\
\Longrightarrow & \{\text{state transition pathway}\} \\
& \lceil I \rceil \ ; \ \lceil M \rceil \\
\Longrightarrow & \{\text{by formula 9}\} \\
& (\int I) \ ; \ (\int M) \\
\Longrightarrow & \{\text{by formula 10}\} \\
& (\int I = \bar{m}\cdot(l_1+l_2)+\bar{n}\cdot(l_1+l_3)+\bar{m}\cdot l_4) \ ; \ (\int M = l_5) \\
\Longrightarrow & (\int I = \bar{m}(l_1+l_2+l_4)+\bar{n}(l_1+l_3)) \ ; \ (\int M = l_5) \\
\Longrightarrow & \{\text{by Des-1 and Des-2}\} \\
& (\int I \leq \bar{m}\cdot r) \ ; \ (\int M \leq r) \\
\Longrightarrow & \{\text{by the addition rule of calculus}\} \\
& \int I + \int M \leq \bar{m}\cdot r + r = (\bar{m}+1)\cdot r \\
\Longrightarrow & \int I + \int M \leq b
\end{aligned}
$$

Thus Req holds on every interval of $b \geq (\bar{m}+1)\cdot r$, and **Theorem 1** is proved. As the increase of iterations is not affected by the events for processing the antigen instances, the duration of relevant events is eliminated in the induction above.

Based on **Theorem 1**, as long as the real-time bound is not smaller than the duration of '$(\bar{m}+1)\cdot r$', the cell can at least complete one lifespan. According to the experiments performed in [9], the value of $\bar{m}$ is normally smaller than 10. Therefore, the single-cell model can satisfy the real-time criteria if the real-time bound is not less than the duration of 11 iterations, which can be easily satisfied in most applications. This suggests that a single cell in the DCA can be performed in real-time. As a consequence, the DCA can perform all the events except the analysis process in real-time, since the algorithm employs a population of such cells that operate in parallel.

5 Discussion of the Analysis Process

The DCA also involves an analysis process that produces the final detection result from the accumulated outputs of matured cells in the population. As mentioned before, the analysis process of the standard DCA is performed offline after all the instances of the input data are processed. In the case that the input data consist of m ($m \in \mathbb{N}$) signal instances, by formula 10, $\frac{m}{\bar{m}}$ lifespans of the cell are required to process the whole input data. To satisfy the real-time bound,

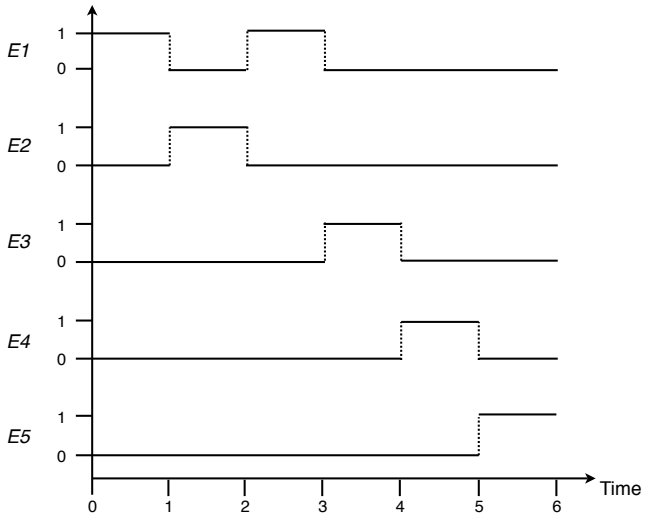


Fig. 2. Interpretation for E_1, E_2, E_3, E_4, and E_5, and the whole interval is divided into subintervals by the events

the duration needed for the standard DCA to get the final detection result can be formalised as in 13.

$$c \cdot \frac{m}{\bar{m}} + l_a \leq b \tag{13}$$

As c, $\bar{m}$ and l_a are constants, whether formula 13 can hold or not is determined by the quantity of m. The value of m is derived from the number of signal instances contained within the input data. As the size of the input data grows, the number of signal instances is getting bigger and bigger. This can cause that the duration for getting the final detection result increases dramatically and exceeds the real-time bound. Therefore, as the size of the input dataset increases, it is becoming more and more difficult for the standard DCA to satisfy the real-time criteria. Therefore, the analysis process of the standard DCA is the weakness of the algorithm in terms of real-time detection.

In order to satisfy the real-time criteria, the analysis process of the standard DCA should be replaced by a real-time analysis component that performs periodic analysis during detection. This can be achieved by segmenting the current output of the DCA, which is performed in a variety of ways, as suggested in [8]. Segmentation involves slicing the output data of the DCA into smaller segments with a view to generating finer grained results and to perform analysis in parallel with the detection process. Segmentation can be performed based on a fixed quantity of output data items or alternatively on a basis of a fixed time period. As the analysis process is performed within each segment, the sub-duration for each segment to produce the detection result is much shorter than the whole duration. It highly possible for the sub-duration to satisfy the real-time bound.

Moreover, the sub-duration can be made to satisfy the real-time bound, by manipulating the segment size that determines the length of each sub-duration. Segmentation is the initial step of developing the real-time analysis component, and eventually an approach that can deal with the online dynamics is required. This approach should be able to adapt and evolve during detection, so that it can deal with the new situations that have not been previously seen. This leads to the future work of dynamic segmentation.

6 Conclusions and Future Work

In this paper, we used the DC specifications to formally describe a simplified single-cell model of the DCA. The temporal properties of the events performed by the cell in each state are included, indicating the dependencies between events. To explore the real-time capability of the DCA, we conduct a test from which **Theorem 1** is derived. The conclusion of **Theorem 1** suggests that each cell of the DCA can operate in real-time, based on the single-cell model. As the DCA employs a population of such cells that operate in parallel, the events functioning detection can be performed in real-time. Therefore, the DCA is potentially capable of performing real-time detection. However, the analysis process of the standard DCA is performed after all the instances of the input data are processed. As a result, if the size of input dataset grows, the duration required for the algorithm to produce the final detection result increases dramatically. This make it more and more difficult for the algorithm to satisfy the real-time criteria. Therefore, in order to develop an effective real-time detection system based on the DCA, the analysis process of the algorithm needs to be replaced by a real-time analysis component, which is capable of performing periodic analysis during detection. Preliminary work on segmentation has been done in [8], and the result appears promising. Eventually an adaptive real-time analysis component that incorporates with dynamic segmentation will be developed.

The DC specifications in this paper focus on the behavioural level of the single-cell model without going into any further details, which is sufficient for the scope of the paper. However, for future work the population level of DCA should also be covered to better present the algorithm. In addition, the DC is mainly used for specifying the requirement of real-time systems, to design and implement real-time systems, other formal methods are also required. These methods include the Timed Automata [2] and the PLC Automata [4], which can be used for modelling cyclic behaviours of interacting objects in real-time systems. Therefore, they are ideal for formally modelling the systems like the DCA that is based on multi-agent framework. Moreover, there are existing tools, such as UPPAAL [15] and so on, which facilitate the automatic verification of the systems modelled in the Timed Automata and PLC Automata. As a result, the designed real-time system can be formally verified before its implementation.

References

1. Al-Hammadi, Y., Aickelin, U., Greensmith, J.: DCA for Bot Detection. In: Proceedings of the IEEE World Congress on Computational Intelligence (WCCI), pp. 1807–1816 (2008)
2. Alur, R., Dill, D.L.: A theory of timed automata. Theoretical Computer Sicence 126, 183–235 (1994)
3. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligent Approach. Springer, Heidelberg (2002)
4. Dierks, H.: PLC-Automat: A new class of implementable real-time automata. Theoretical Computer Sicence 253, 61–93 (2000)
5. Dijkstra, E.W., Scholten, C.S.: Predicate calculus and program semantics. Springer, Heidelberg (1990)
6. Greensmith, J.: The Dendritic Cell Algorithm. PhD thesis, School of Computer Science, University of Nottingham (2007)
7. Greensmith, J., Feyereisl, J., Aickelin, U.: The DCA: SOMe Comparison A comparative study between two biologically-inspired algorithms. Evolutionary Intelligence 1(2), 85–112 (2008)
8. Gu, F., Greensmith, J., Aickelin, U.: Further Exploration of the Dendritic Cell Algorithm: Antigen Multiplier and Time Windows. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 142–153. Springer, Heidelberg (2008)
9. Gu, F., Greensmith, J., Aickelin, U.: Integrating Real-Time Analysis With The Dendritic Cell Algorithm Through Segmentation. In: Genetic and Evolutionary Computation Conference (GECCO), page in print (2009)
10. Lutz, M.B., Schuler, G.: Immature, semi-mature and fully mature dendritic cells: which signals induce tolerance or immunity? TRENDS in Immunology 23(9), 445–449 (2002)
11. Moszkowski, B.: A temporal logic for multilevel reasoning about hardware. Computer 18(2), 10–19 (1985)
12. Oates, R., Greensmith, J., Aickelin, U., Garibaldi, J., Kendall, G.: The Application of a Dendritic Cell Algorithm to a Robotic Classifier. In: de Castro, L.N., Von Zuben, F.J., Knidel, H. (eds.) ICARIS 2007. LNCS, vol. 4628, pp. 204–215. Springer, Heidelberg (2007)
13. Oates, R., Kendall, G., Garibaldi, J.: Frequency Analysis for Dendritic Cell Population Tuning: Decimating the Dendritic Cell. Evolutionary Intelligence 1(2) (2008)
14. Olderog, E., Dierks, H.: Real-Time Systmes: Formal Specification and Automatic Verification. Cambridge University Press, Cambridge (2008)
15. UPPAAL. An integrated tool environment for modeling, simulation and verification of real-time systems (2009), `http://www.uppaal.com/`
16. Zhou, C., Hansen, M.R.: A calculus of durations. Information Processing Letters 40(5), 269–276 (1991)
17. Zhou, C., Hansen, M.R.: Duration Calculus: A Formal Approach to Real-Time Systems. Springer, Heidelberg (2004)

On AIRS and Clonal Selection for Machine Learning

Chris McEwan and Emma Hart

Napier University, Edinburgh, Scotland
{c.mcewan,e.hart}@napier.ac.uk

Abstract. AIRS is an immune-inspired supervised learning algorithm that has been shown to perform competitively on some common datasets. Previous analysis of the algorithm consists almost exclusively of empirical benchmarks and the reason for its success remains somewhat speculative. In this paper, we decouple the statistical and immunological aspects of AIRS and consider their merits individually. This perspective allows us to clarifying why AIRS performs as it does and identify deficiencies that leave AIRS lacking. A comparison with Radial Basis Functions suggests that each may have something to offer the other.

1 Introduction

The Artificial Immune Recognition System (AIRS) was proposed by Watkins [23,22], extending a lineage of immune-inspired work on unsupervised learning to the supervised domain. Initial results were favourable and these results have been reproduced several times by different authors [14]. To this day, AIRS remains one of the most widely studied and applied AIS in pattern classification. This popularity is further encouraged by a publicly available plug-in[1] for the Weka Data Mining environment [24].

Theoretical insight into *why* AIRS performs as it does remains scant. Several hypotheses have been tentatively offered in the literature [7,12], but did not reach any definite conclusions. These studies tend to lack the rigour typical of machine learning literature. It is from this perspective that we attempt to approach AIRS in this paper.

The paper unfolds as follows: In Sect. 2, after introducing AIRS, an experiment with a simplified derivative algorithm validates some concerns and allow us to work back towards the full AIRS algorithm, bringing its main functionality into focus. This then points to additional issues that we explore and verify experimentally, building a rather complete technical picture of AIRS as a learning algorithm. In Sect. 3, we propose that some of these issues can be rectified by exploiting aspects of a more classical approach, Radial Basis Functions. By comparison and experiment, we demonstrate that each may have something to offer the other. This leads to a more general comparison between clonal selection and classical iterative descent algorithms. We conclude in Sect. 4 by taking a broader view toward future work.

[1] http://www.artificial-immune-systems.org/algorithms.shtml

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 67–79, 2009.

2 AIRS

AIRS is an unweighted k-nearest neighbour classifier. The immunological inspiration contributes to how the algorithm is trained to develop a repertoire of "memory cells" which is based, loosely, on Burnet's Clonal Selection theory [9]. We outline this process in Algorithm (1): memory cells (prototypes) stimulated by antigen (data) proliferate and mutate; these stimulated cells and their progeny compete under selective pressure for continued stimulation, resulting in only the fittest being aggregated into the repertoire. It is this repertoire that is used for classification, in proxy of the full data set.

```
memory = initialiseRandomRepertoire()
for (x,y) in trainingData do
    best = memory.bestMatchingCell(x,y)
    pool = [best]
    while pool.avgStimulation() < threshold do
        for cell in pool do
            pool.add(cell.mutations(cell.stimulation(x)))
        end
        pool.cellsCompeteForResources()
    end
    fit = pool.fittestCell()
    if fit > best then
        memory.add(fit)
        if ||best − fit||_2 < ε then
            memory.remove(best)
        end
    end
end
```

Algorithm 1. Pseudo-code for the AIRS training procedure

Variations of this general strategy abound in the Pattern Recognition literature, thus, the immunological component is the key point of distinction of AIRS as an algorithm. Typically, clonal selection is applied in a (black-box) optimisation setting: each cell represents a solution; their "stimulation" reflecting an objective function value. This variant on evolutionary algorithms has some practical benefits that result from the immune-system's particular *hyper-mutation* process: there is no arbitrary parental cross-over in generating new solutions; and mutation in inverse proportion to stimulation promotes poor solutions to explore the space (few large mutations) and exploits better solutions which more conservatively approach their local optima (via many small mutations).

This population-based stochastic hill-climbing strategy has proven to be most effective in complex multimodel and multi-objective optimisation settings [6]. Certainly, finding the locations of prototypes in our data's feature space can be cast as an optimisation problem. This is vaguely implied in the AIRS literature, but a simple argument shows that this implication is misleading.

2.1 Clonal Selection in the Learning Context

Recall, AIRS takes a "one-shot" pass through the training set, responding to each datum individually. Each prototype μ_k receives stimulation as an inverse function of distance[2] from the datum x_i. This stimulation parameterises the quantity of mutants produced and the magnitude of mutation suffered.

We question the validity of applying black-box stochastic optimisation in a unimodal setting where stimulation and location have a monotonic relationship. Quite simply, the algorithm "knows" the direction and distance from its current optimum – this is used directly in calculating stimulation – and so random search appears to serve no valid purpose. As illustrated in Fig. (1) each best matching prototype μ_t has a surrounding region of potential mutations (solid circle) with an obvious optimal step μ_{t+1}. Over half of the potential mutations (shaded region) will be *a priori* less fit than the parent.

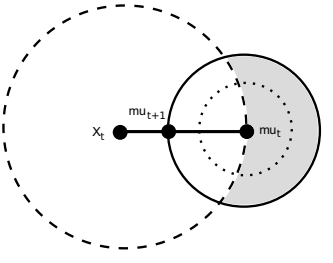


Fig. 1. A schematic representation of the stochastic search procedure for AIRS. There is a trivial (one generation) optimum μ_{t+1} easily derived from the same information used to calculate stimulation. Further, over half the potential mutations of a prototype μ_t will necessarily be less fit.

One might reason that the benefit of this stochasticity may be to overcome AIRS' necessarily myopic nature: rather than directly chasing immediate short-term optima, some random noise allows the algorithm to average out movements without averaging across the data (which should be inaccessible in batch form). While attractive, this justification is heavily contradicted by the implementation. The algorithm has an overly elitist selection criteria: only the best matching memory cell initiates a response, and only the best matching mutant becomes a candidate memory cell. Further, the generation of separate mutation pools per datum, from which only the best candidate can survive, does not favour retaining mutants that may still prove beneficial in hindsight.

A simple experiment clarifies. We completely remove the immunological component from AIRS, replacing it with a trivial, deterministic update which we

[2] Specifically $1 - ||\widehat{\mu_k - x_i}||_2$ where the hat represents a "normalised" Euclidean distance. We will have more to say on this later.

dub AIRS$^-$ (see Alg. 2). Here, we simply choose a single candidate memory cell exactly halfway between the datum and the best matching memory cell[3].

```
memory = initialiseRandomRepertoire()
for (x,y) in trainingData do
    best = memory.bestMatchingCell(x,y)
    fit = 0.5 * (best + x)
    memory.add(fit)
    if ||fit − best||_2 < ε then
        memory.remove(best)
    end
end
```

Algorithm 2. Pseudo-code for AIRS$^-$. The optimal (one step) candidate is chosen deterministically, rather than via AIRS' many rounds of mutation and resource competition.

The performance differences for several datasets are reported in Table 1. The figures validate our concern: *the clonal selection phase of AIRS has almost no positive effect on classifiers performance*. Not only is the stochastic search unnecessary, it can be detrimental. AIRS$^-$ performs significantly better on all high-dimensional datasets. Indeed, on the *newsgroup* dataset AIRS performs no better than random guessing. For comparison, on the same task 3-nearest neighbour achieves 75% accuracy, linear regression 80% and Multinomial Naive Bayes 97%. This suggests that the degrees of freedom in high-dimensional space seriously impede the stochastic search procedure; which is intuitive, but contrary to previous claims.

2.2 From AIRS$^-$ Back to AIRS

In deriving the deterministic update rule for AIRS$^-$ we simply performed the logical extreme of what AIRS was indirectly attempting by blind search. We can improve our understanding of AIRS if we pursue this idea some more. Recall, that for every datum the evolved candidate lies (somewhat) between the datum and the previously closest prototype. In AIRS$^-$ we used the update rule

$$\mu_{t+1} = \gamma(x_t + \mu_t) \tag{1}$$

where μ_{t+1} is the candidate, μ_t was the previous best matching prototype and $\gamma = 0.5$ was the distance to the boundary of the mutation region. Some trivial

[3] In an earlier experiment, rather than replace the evolutionary search phase we allowed the deterministically chosen candidate to compete with AIRS' mutants, but did not allow the candidate to mutate new solutions. In this regime, it is possible that the stochastic search could mutate past the optimal (one step) midpoint, getting even closer to the antigen. In fact, this almost never occurred – our deterministically selected midpoint was, almost without exception, selected as the fittest candidate for each training instance.

Table 1. Performance comparison of AIRS and our deterministic derivative. Experiments were performed in Weka using the default algorithm parameters, 10-fold stratified cross-validation and a paired T-test. Most datasets are standard UCI benchmark problems. *Elements* is a synthetic mixture of gaussians taken from [8] which is designed, for pedagogical reasons, to favour neither local nor global learning methods. *Newsgroups* is a two-class classification of determining `comp.graphics` from `alt.atheism` posts using a subset of the 20 Newsgroup dataset.

	dimension	AIRS	AIRS$^-$
elements	2	74.35 ± 7.29	71.95 ± 7.72
iris	4	94.67 ± 5.36	94.47 ± 6.34
balance	5	80.93 ± 4.11*	77.36 ± 4.83
diabetes	8	71.60 ± 4.40*	69.45 ± 4.98
breastcancer	9	96.28 ± 2.35	96.35 ± 2.19
heart-statlog	13	78.15 ± 8.63	77.11 ± 7.34
vehicle	18	62.05 ± 4.89*	57.06 ± 6.04
segment	19	88.21 ± 2.48*	83.79 ± 2.91
ionosphere	34	84.44 ± 5.18	89.66 ± 5.39*
sonar	60	67.03 ± 11.60	84.58 ± 7.86*
newsgroup	3783	51.35 ± 4.60	78.87 ± 14.05 *
* significant at p-value of 0.001			

manipulation allows us to express (1) as

$$\mu_{t+1} = \mu_t + \gamma(x_t - \mu_t) \tag{2}$$

which is also the formula for an exponentially weighted moving average. Rather than holding γ fixed, we can incorporate AIRS' mutation as a function of stimulation, by allowing γ to decrease as stimulation increases

$$\mu_{t+1} = (1 - \gamma)\mu_t + \gamma x_t \tag{3}$$

which is simply a linear interpolation between prototype and datum. The only significant difference between this and AIRS is that AIRS will take many indirect steps over several generations, before selecting the "best" found.

Now, Eq. (2) and (3) are *exactly* the update rule for MacQueen's 1967 online k-means algorithm [11]. But whereas K-means explicitly moves μ_t to μ_{t+1}, AIRS keeps one or both depending on their mutual pairwise distance. Also, k-means will monotonically decrease γ over time, ensuring convergence of centroid locations; in contrast, AIRS employs a datum-specific value of γ based on pairwise distance. We now address any contribution of these differences in AIRS.

2.3 Representational Power of the AIRS Repertoire

Given the previous analysis, we can see that the repertoire of memory cells in AIRS are a distorted snapshot of the trajectory of a moving average – distorted, because the direction and magnitude of movements are stochastic, undirected and unconstrained.

Based on this observation, we hypothesise that, though smaller in size, the AIRS repertoire does *not* compress or otherwise extract meaningful structure from the original dataset. We validate this claim by comparing the sum of squared distances between data and their closest memory cell, against that of k-means with the same number of centroids as AIRS memory cells (see Table 2). For non-trivial datasets, AIRS is far from the local optima found by k-means. Indeed, we can find the value $\hat{k}$ for k-means that produces the same performance as AIRS. It is apparent that a significantly larger amount of compression is possible than is achieved by AIRS.

Table 2. The within-cluster squared distances for AIRS and k-means using the same number of centroids as AIRS' memory cells. The value $\hat{k}$ is the number of k-means required to produce the same performance as AIRS. This tends to be dramatically lower than the number of AIRS memory cells, reinforcing that AIRS' repertoire, though smaller than the dataset, has not extracted meaningful structure.

	k (memory)	AIRS	k-means	$\hat{k}$
iris	47	1.10	0.768	20
balance	295	16.93	13.5	225
diabetes	407	22.81	8.028	125
breastcancer	209	55.22	28.0	100
heart-statlog	209	108.46	9.036	20
vehicle	336	92.50	23.284	25
segment	219	135.81	51.81	45
ionosphere	145	410.66	94.86	12
sonar	143	420.04	38.679	3

In Fig. (2) we illustrate this effect for the 2-dimensional *elements* dataset, on which AIRS performed reasonably. As Table (2) shows, the effect is less pronounced in low dimensions, but we are limited by what can be visualised. It is still clear from inspection that the density of the repertoire does not reflect the density of the data. Indeed, the repertoire appears to be uniformly spread within the same bounding region as the data. A similar result has been demonstrated by Timmis and Stibor for the algorithm *ai-Net* [19]. Although the details of both algorithms are quite different, AIRS also suffers from the same problem: when deciding if the candidate should replace the original prototype, inflicting a fixed threshold (based on mean pairwise distance) makes it is impossible for the algorithm to represent densities at a finer granularity than that threshold. Further, the mutual exclusion between prototype and candidate precludes any compromise by selecting, say, an intermediate representation such as an average between both.

In learning, in order to improve compression, generalisation and discrimination it is necessary to control the granularity of density representation: coarse in coherent, homogeneous regions; fine in ambiguous regions near the decision boundary. AIRS is limited in what it can achieve here. All prototypes share a

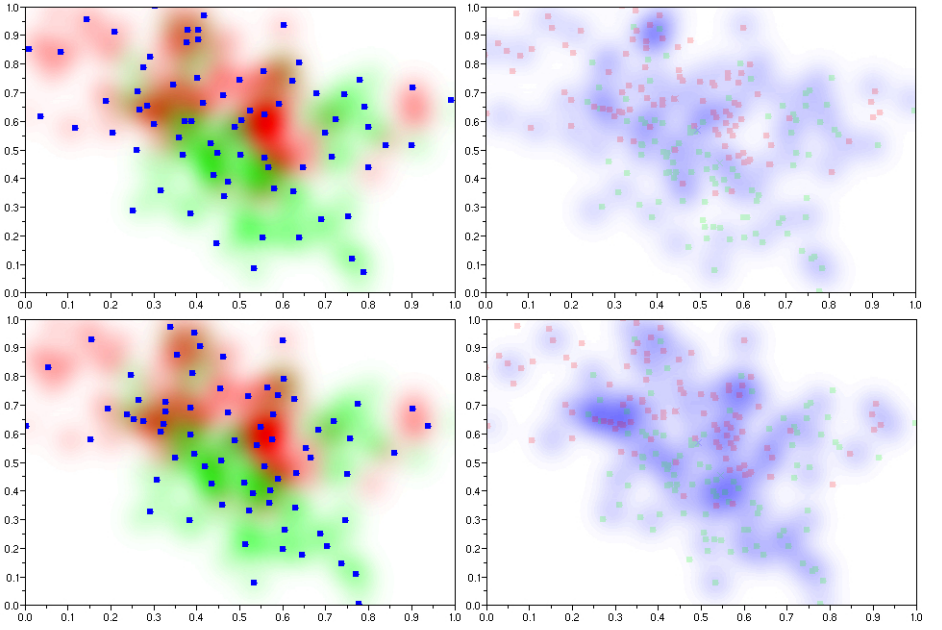


Fig. 2. AIRS memory repertoire for the *elements* dataset (top) compared to the same number of k-means (bottom). The left column illustrates the repertoire/means (blue) superimposed over the class density of the training data (red and green gradients with dark regions representing $p(+) \approx p(-)$). It is clear from inspection that the AIRS repertoire does not follow the density of the data. This is emphasised in the right column, where we illustrate the density of the repertoire/means with the dataset superimposed. The AIRS repertoire appears to have a weak uniform coverage compared to k-means. In higher dimensions, these effects would be much more pronounced, as demonstrated quantitatively in Table (2).

common pairwise distance constraint and violations are resolved simplistically, with no regard for the engineering goal (or relevant biological dynamics).

2.4 Discriminatory Power of the AIRS Repertoire

Following training, to classify data AIRS takes an unweighted majority vote amongst the prototypes. This produces very coarse decision boundaries and it is apparent that there is a lot of additional information that AIRS is ignoring. Some form of weighting (e.g. by prototype stimulation to test datum and prototype training fitness) would likely be beneficial. A previous investigation lends some support to this idea [12].

It is worth making entirely explicit that AIRS is essentially a *generative model*: it is an unsupervised learning process repeated in C_i separate compartments; one for each class. The classification decision can be easily interpreted as asking for the compartment that would be most likely to have generated the test instance.

However, unlike a generative model, AIRS makes no use of (anything equivalent to) prior probability in either determining the most representative class *or* the most appropriate prototype to select as the best. If the class distribution is skewed, the former will significantly influence AIRS' accuracy. The latter reduces AIRS' choice of "best" to simply closest, rather than a more general criteria of "fittest" developed during the training period.

AIRS is further limited because there is no feedback between data or prototypes of different class compartments. As such, the training process is blind to any ambiguity in the regions where complementary prototypes overlap: the most important regions for classification.

2.5 One-Shot or Not?

Prior to commencing Alg. (1) AIRS performs two relatively expensive initialisation procedures. Together, these procedures are $O(nm)$ and $O(nm^2)$, where m is the size of the dataset and n the dimensionality. We suspect this initialisation process is largely responsible for AIRS' often touted "out of the box" performance. However, such convenience is not without cost.

The $O(nm^2)$ step is the computation of an internal parameter – the mean pairwise affinity – which is used as a threshold distance to decide whether a candidate should replace its parent memory cell. In Sect. (2.3) we demonstrated the negative effects of using a uniform fixed threshold. The $O(nm)$ procedure is a "min-max" attribute normalisation. All data are rescaled and translated to lie in a unit bounding box, which simplifies logic by bounding affinity values and legal mutations. However, by computing these bounds at initialisation, AIRS cannot continually learn, as claimed, as such bounds do not remain valid – even for hold-out test data. Further, such normalisation largely presumes a Euclidean distance metric and does not generalise well. In short, this is arbitrary preprocessing, best left to the practitioner. Currently, the internals of AIRS are unnecessarily coupled with this particular initialisation procedure.

Though only polynomial in time, these costs further undermine any practical value in the claim of "one-shot" learning. Both historical and contemporary interest in online learning has been largely driven by its linear-time, fixed-space computational costs – e.g. learning from streaming or massive datasets. In these contexts, an initial linear scan or pairwise comparison is highly undesirable, if not impossible. The current design of AIRS assumes a batch/online compromise that suits neither situation: the computational cost of a batch algorithm and the learning restrictions of a one-shot algorithm.

3 Radial Basis Functions

Having uncovered some theoretical and practical issues with AIRS, we look to something more statistically solid on which to motivate and justify any novel immune-inspired deviations. Radial Basis Functions (RBF) [8,3] present a simple and elegant compromise between the trade-offs inherent in global (e.g. least

squares) and local (e.g. k-nearest neighbour) methods of learning. These trade-off are well documented and we will not labour over them here. An RBF classifies a data point $\hat{x}$ as

$$\begin{aligned}\hat{y} &= f(\hat{x}) \\ &= \sum_{i=1}^{k} \alpha_i \mathcal{K}(c_i, \hat{x}) \\ &= \sum_{i=1}^{k} \alpha_i \exp(-\beta_i ||\hat{x} - c_i||_2^2)\end{aligned}$$

where k is the number of c_i kernel centres (i.e. prototypes), $\mathcal{K}$ is a symmetric distance function parameterised by bandwidth β_i, and α_i are the weights of each prototype, to be found by training.

How the prototypes are chosen is quite arbitrary, though a common approach is to perform a k-means clustering of the data, prior to the supervised learning stage[4]. K-means converges on a local optima of minimising the sum of squared distances between prototypes and their assigned data-points

$$\underset{\mu_1 \ldots \mu_k}{\operatorname{argmin}} \sum_{i}^{k} \sum_{x_j \in C_i} ||x_j - \mu_i||_2^2$$

One can certainly question the validity of any optimisation criterion; the only point we wish make here is that there *is* a criterion. However, choosing the correct value for k is somewhat more troublesome. Further, the algorithm must be run several times as the quality of local optima depends on the initial (often randomised) placement of prototypes. Further still, this is typically a batch process, scaling poorly in the size of the dataset.

Regardless, assuming appropriate prototypes the RBF represents each datapoint as k features – the distances from each of the k prototypes. The method of least-squares is then employed in this reduced space to solve for $\alpha = \hat{X}^+ y$ where $\hat{X}^+$ is the pseudo-inverse of the transformed training data $\hat{X}_{ij} = \mathcal{K}(c_i, x_j)$. The elegance of this approach is two-fold: During training, the computational burden of a global least-squares solution is eased by reducing the dimensionality. During testing, performance is improved by avoiding lazy-learning. In both cases, the kernel function incorporates beneficial, non-linear locality.

3.1 A Comparison between RBF and AIRS

Though the details are somewhat different, there is an obvious high-level similarity in both approaches: *find the best positions for prototypes that can act as a proxy for the full training data and the full feature set.* We now highlight the key differences.

[4] More generally, one can fit a finite mixture model with the EM Algorithm [13]. RBF are essentially unnormalised, symmetric Gaussian mixtures.

- **Training:** The RBF has a well-defined optimisation criteria, although there is no well-defined manner to choose k. In contrast, AIRS (and brethren) aim to regulate the number of prototypes, but typically have no wider notion of optimally with which to drive evolution. AIRS implicitly partitions the repertoire into classes and fits prototypes to each class. Conversely, the choice of prototype placement for an RBF is *unsupervised*; supervisory information has its influence in the least squares solution for α.
- **Testing:** The RBF uses a combination of the optimal α, the chosen kernel $\mathcal{K}$ and bandwidth β to arrive at a classification decision. Conversely, AIRS simply chooses the partition with the majority of k matching prototypes.
- **Updating:** The cost of having optimal weights is that there is no efficient manner to update an RBF model, without re-computing and inverting $\hat{X}$. Because AIRS is a lazy-learner, it is, in principle, more suitable to update and adapt during execution.
- **Data Access:** A critical difference between both is that RBF uses the full dataset to fit prototypes, whereas AIRS treats data sequentially. One-shot learning may be a desirable feature to retain, but as we have demonstrated above, some significant changes would be required.

We propose that each method has something to offer the other. The RBF has a well defined optimisation criteria, a more elegant approach to handling distances, and a more sophisticated weighted decision process. AIRS can naturally perform multiclass classification, has the potential of deriving its own k per class and an inherent, if poorly utilised, capacity for adaptive updating.

Table 3. Classification accuracy comparison of AIRS and Radial Basis Functions. The RBF is handicapped to only two prototypes per class, compared to the AIRS repertoire size for the same datasets in Table (2).

	AIRS	RBF (2)
balance	80.93± 4.11	86.18± 3.76 *
breastcancer	96.40± 2.18	96.18± 2.17
diabetes	71.60± 4.40	74.06± 4.93 *
heart-statlog	78.15± 8.63	83.11± 6.50 *
ionosphere	85.53± 5.51	91.74± 4.62 *
iris	94.67± 5.36	96.00± 4.44 *
segment	88.21± 2.48 *	87.32± 2.15
sonar	67.03±11.60	72.62± 9.91 *
vehicle	62.05± 4.89	65.34± 4.32 *
elements	69.85±10.69	73.80± 10.28 *
***** significant at p-value of 0.05		

We illustrate the potential for contribution in Table (3) with a comparison between AIRS and RBF fit by k-means. This comparison is not entirely fair, as the RBF was fit in a batch setting. Although the RBF benefits from random access to all data, we wish to emphasise the virtues of a higher-level optimisation criteria; the capacity to generalise coherent and particularise ambiguous

regions using variable bandwidths; and a weighted decision process. As such, we handicap the RBF to only two centroids per class; it still outperforms AIRS.

Much of the RBF theory cleans up ad-hoc features of AIRS. Note that none of these changes compromise any "immunological metaphor". On the contrary, in some respects the metaphor is improved by introducing clone populations (weights), weighted responses, and adaptive recognition regions (bandwidths). Indeed, similar ideas have already been explored in the AIS literature, though in somewhat different contexts [1,2,20].

3.2 A Comparison between Clonal Selection and Iterative Descent

In the learning context, classical iterative descent algorithms, such as k-means and the EM Algorithm, largely embody the "immune principles" that drive AIRS – *clonal selection* (assign responsibility for data amongst prototypes) and *affinity maturation* (optimise prototypes based on assigned data). Thus, although clonal selection and affinity maturation may be a *necessary* element of immune-inspired learning algorithms (in that they approach essential functionality), they do not appear *sufficient* to determine novelty or value.

It would be remiss to not point out that the greedy nature of classical iterative descent algorithms is not lost on the statistical literature. There are many attempts to make these algorithms more adaptive and global in their optimisation ([13, Chapter 6] [18,17,4]). Evolutionary approaches have also contributed to this domain, though to our knowledge, the evolutionary search typically encodes *all* mixture parameters in a single genotype, and is then used to find a better initial configuration for iterative descent (see e.g. [10]). This is quite different from AIS learning algorithms, where each clone is, essentially, a mixture component and the emphasis tends to be on continual adaptation and internal dynamics of the repertoire.

By removing these internal dynamics, trivialising the fitness landscape and using pairwise distance as a proxy for an overarching objective function, AIRS leaves stochastic search with no competitive advantage over traditional iterative descent. These issues can all be addressed. However, clonal selection alone seems unlikely to induce convincing immune-like behaviour. Building upon clonal selection appears necessary if AIRS is to offer more than a global optimisation method for a prototype-based learning algorithm. Acknowledging the contributions of, and any similarities to, classical algorithms seems the best way to fruitfully direct novel immune-inspired research in this mature field.

4 Conclusion and Future Work

This paper reinforces warnings that one cannot work exclusively within the immunological metaphor [21]. In the present case, the metaphor has obscured contradictory design decisions and functional omissions with regard to the problem domain. Although these deficiencies may appear manifold, many are elementary and quite straightforward to address. The exception being stochastic search on

a uni-modal landscape, which is neither theoretically valid, computationally desirable or biologically plausible. We intend to address this in future work, by trading-off complexity in the fitness landscape against scaling independently from the size of the dataset.

When the strictly one-shot requirement on AIRS is relaxed, it begins to resemble many of the idiotypic-network style algorithms that preceded and inspired it – Neal's *meta-stable memory* [15], Timmis' *RAIN* [6], Von Zuben and de Castro's *CLONALG* and *ai-Net* [5,16]. Because AIRS is still essentially an unsupervised algorithm, run in class-specific partitions, any of these unsupervised AIS could equally be used to determine a repertoire of memory cells. But based on previous experience with some of these algorithms, we suspect that a good deal of what has been discussed with regard to RBF and iterative descent would largely translate to these settings. Rather, we would propose that there may be an opportunity for unifying this lineage of work, by acknowledging (and leveraging) existing research in machine learning and non-parametric statistics. Decoupling the statistical aspects of AIS from the immunological component aids clarity and correctness. Closure on the contribution of clonal selection would clear the way for more focused and sophisticated immunological contributions; that can be transparently motivated and communicated without metaphor.

References

1. Alonso, O., Gonzalez, F., Niño, F., Galeano, J.: A solution concept for artificial immune networks: A coevolutionary perspective. In: de Castro, L.N., Von Zuben, F.J., Knidel, H. (eds.) ICARIS 2007. LNCS, vol. 4628, pp. 35–46. Springer, Heidelberg (2007)
2. Andrews, P., Timmis, J.: Adaptable lymphocytes for artificial immune systems. In: 7th International Conference in Artificial Immune Systems, pp. 376–386. Springer, Heidelberg (2008)
3. Bishop, C.M.: Pattern Recognition and Machine Learning (Information Science and Statistics). Springer, Heidelberg (2006)
4. Cappe, O., Moulines, E.: Online em algorithm for latent data models. Submitted to the Journal of the Royal Statistical Society Series B (December 2008)
5. de Castro, L.N., Von Zuben, F.J.: The clonal selection algorithm with engineering applications. In: GECCO 2000, Workshop on Artificial Immune Systems and Their Applications (2000)
6. De Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, London (2002)
7. Goodman, D.E., Boggess, L., Watkins, A.: An investigation into the source of power for airs, an artificial immune classification system. In: Proceedings of the International Joint Conference on Neural Networks, 2003, vol. 3, pp. 1678–1683 (2003)
8. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer, Heidelberg (2001)
9. Janeway, C.A., Travers, P., Walport, M., Schlomchik, M.: Immunobiology. Garland (2001)

10. Jank, W.: The em algorithm, its randomized implementation and global optimization: Some challenges and opportunities for operations research. In: Perspectives in Operations Research, pp. 367–392. Springer, Heidelberg (2006)
11. Macqueen, J.: Some methods for classification and analysis of multivariate observations. In: Proc. Fifth Berkeley Symp. on Math. Statist. and Prob., vol. 1, pp. 281–297 (1967)
12. Marwah, G., Boggess, L.: Artificial immune systems for classification: Some issues. In: 1st International Conference in Artificial Immune Systems, vol. 1, pp. 149–153. Springer, Heidelberg (2002)
13. Mclachlan, G.J., Krishnan, T.: The EM Algorithm and Extensions, 1st edn. Wiley-Interscience, Hoboken (1996)
14. Meng, L., van der Putten, P., Wang, H.: A comprehensive benchmark of the artificial immune recognition system (AIRS). In: Li, X., Wang, S., Dong, Z.Y. (eds.) ADMA 2005. LNCS, vol. 3584, pp. 575–582. Springer, Heidelberg (2005)
15. Neal, M.: Meta-stable memory in an artificial immune network. In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 168–180. Springer, Heidelberg (2003)
16. de Nunes Castro, L., Von Zuben, F.J.: An evolutionary immune network for data clustering. In: Sixth Brazilian Symposium on Neural Networks, 2000. Proceedings, pp. 84–89 (2000)
17. Singer, Y., Warmuth, M.K.: Batch and on-line parameter estimation of gaussian mixtures based on the joint entropy. In: Proceedings of the 1998 conference on Advances in neural information processing systems II, pp. 578–584. MIT Press, Cambridge (1999)
18. Song, M., Wang, H.: Highly efficient incremental estimation of gaussian mixture models for online data stream clustering. In: Priddy, K.L. (ed.) Intelligent Computing: Theory and Applications III, vol. 5803, pp. 174–183. SPIE (2005)
19. Stibor, T., Timmis, J.: An investigation on the compression quality of ainet. In: IEEE Symposium on Foundations of Computational Intelligence, FOCI 2007, pp. 495–502 (2007)
20. Stibor, T.: Discriminating self from non-self with finite mixtures of multivariate bernoulli distributions. In: GECCO 2008: Proceedings of the 10th annual conference on Genetic and evolutionary computation, pp. 127–134. ACM, New York (2008)
21. Timmis, J., Hone, A., Stibor, T., Clark, E.: Theoretical advances in artificial immune systems. Theor. Comput. Sci. 403(1), 11–32 (2008)
22. Watkins, A., Timmis, J., Boggess, L.: Artificial immune recognition system (AIRS): An immune-inspired supervised learning algorithm. Genetic Programming and Evolvable Machines 5(3), 291–317 (2004)
23. Watkins, A.B.: Airs: A resource limited artificial immune classifier. Master's thesis, Mississippi State University, MS. USA (2001)
24. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, San Francisco (2005)

A Theoretical Analysis of Immune Inspired Somatic Contiguous Hypermutations for Function Optimization

Thomas Jansen[1,⋆] and Christine Zarges[2]

[1] University College Cork, Department of Computer Science, Cork, Ireland
t.jansen@cs.ucc.ie
[2] TU Dortmund, Fakultät für Informatik, LS 2, 44221 Dortmund, Germany
Christine.Zarges@tu-dortmund.de

Abstract. Artificial immune systems can be applied to a variety of very different tasks including classical function optimization. There are even artificial immune systems tailored specifically for this task. In spite of the successful application there is little knowledge and hardly any theoretical investigation about how and why they perform well. Here a rigorous analysis for a specific type of mutation operator introduced for function optimization called somatic contiguous hypermutation is presented. While there are serious limitations to the performance of this operator even for simple optimization tasks it is proven that for some types of optimization problems it performs much better than standard bit mutations most often used in evolutionary algorithms.

1 Introduction

Theoretical analysis of general randomized search heuristics (RSHs) is a modern and important area of research. General RSHs are a broad family of very different algorithms including randomized local search [1], simulated annealing [17], evolutionary algorithms (EAs) [10], artificial immune systems (AISs) [7,8], and many others. They provide a powerful and flexible way of tackling different problems when there is no time or expertise to develop a problem-specific solution. While being applicable in very different situations one of the most important tasks is function optimization. General RSHs are typically very easy to implement and apply but it turns out to be extremely challenging to prove in a rigorous way results about their performance and limitations. For EAs, there is a growing body of useful analytical tools and relevant results [12,18,19]. For AISs, currently there are hardly any theoretical investigations about their performance at all. One noteworthy exception is the work by Zarges [20].

Zarges [20] considers a mutation operator with inversely fitness proportional mutation rate often used in artificial immune systems and analyzes its performance when used in a very simple algorithmic framework. This way Zarges

⋆ This material is based in part upon work supported by the Science Foundation Ireland under Grant No. 07/SK/I1205.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 80–94, 2009.

proves this operator to be inefficient for a very simple optimization problem tackled with a kind of hill-climber. When used together with a population, however, its performance improves drastically [21]. In the same spirit we consider a mutation operator designed for the use of AISs as function optimizers, namely somatic contiguous hypermutations (CHM). Again, we embed this operator in a very simple algorithmic framework that allows us to concentrate on the assets and drawbacks of this specific mutation operator. This gives clear indications for the kind of optimization problems where artificial immune systems may turn out to be extremely efficient and to clearly outperform other RSHs like EAs.

Artificial immune systems are a special class of biologically inspired algorithms, which are based on the immune system of vertebrates and derive from various immunological theories, namely the clonal selection principle, negative selection, immune networks or the danger theory [7,8]. Besides the natural tasks of anomaly detection and classification, they are often applied to function optimization. In this context, mostly algorithms based on the clonal selection principle [3], a theory which describes the basic features of an adaptive immune response to invading pathogens (antigens), are used. During the last years, many clonal selection algorithms to tackle optimization problems have been developed, for example: CLONALG [9], OPT-IA [6], the B-Cell-Algorithm [16] and MISA [5]. These algorithms share a common approach in a broad sense. They work on a population of immune cells or antibodies that represent candidate solutions of the considered problem, i. e. the function to be optimized. The antibodies proliferate and undergo a hypermutation process called affinity maturation, implying mutations at high rate. The design of the mutation operator can be inspired by various types of immune cells found in the immune system.

The AIS for function optimization considered here was introduced by Kelsey and Timmis [16], known as B-Cell-Algorithm. They introduce a mutation operator inspired by the mutation mechanism found in B-cell receptors. Since we concentrate on the analysis of RSHs from a computational point of view this biological background used as a motivation is not important to us. Details of the operator including the precise probability distributions used can be found in Section 2. A convergence analysis of this algorithm is presented by Clark et al. [4] using a Markov chain model. Bull et al. [2] apply it on benchmarks generated as integer programming problems derived from Diophantine equations.

In the next section we give a precise formal description of CHMs and the algorithmic framework under consideration. We present results on well-known and instructive example functions thereafter (Section 3). This will help to get a clear understanding of assets and drawbacks of CHMs. We conclude and point out open problems as well as directions for future research that we consider important in Section 4.

2 Definitions and Analytical Framework

The mutation operator introduced by Kelsey and Timmis [16] performs CHMs on bit strings of length n in the following way. It chooses a position in the bit

string uniformly at random and determines a length of an interval to be mutated randomly. Then all bits beginning with the random position up to the end of the interval with random length are inverted with a given probability $r \in [0,1]$. The desired length of the interval is thereby chosen uniformly at random from $\{0,1,\ldots,n\}$ and thus equals $n/2$ in expectation. Note, that the bit string is not assumed to be cyclic and therefore the contiguous region does not wrap around. Since mutation stops at the end of the bit string the distribution of the actual length is not uniform and its expectation is less than $n/2$.

We observe that this operator is biased towards changing the value of bits farther to the right in the bit string. The more to the left a bit is positioned, the less likely it becomes that it is part of a randomly selected interval. Without additional knowledge about the roles of different bits such a bias is undesirable [13]. There are different ways such a bias could be removed and it is of interest to perform analyses for such modified mutation operators.

Here, we consider one specific variant. Instead of choosing one point and the length of the interval, we choose two positions p_1 and p_2 from the bit string uniformly at random and consider positions in-between, i. e. positions between $\min\{p_1,p_2\}$ and $\max\{p_1,p_2\}$, as within the interval. Then each bit $x[i]$ with $\min\{p_1,p_2\} \leq i \leq \max\{p_1,p_2\}$ is flipped with probability r independently for each such bit.

Note that this operator is also positionally biased. The probability to be mutated is larger for bits in the middle of the bit string. The important difference is the following. For mutations of single bits the probability is independent of the bit position. Since single bit mutations can be crucial this kind of being bias-free is important.

Here, we consider the extreme case $r = 1$. Note that this rules out mutations not flipping any bit as well as global convergence. Consider for example the function $f\colon \{0,1\}^n \to \mathbb{R}$ with

$$f(x) = \begin{cases} \sum_{i=1}^{n} x[i] & \text{if } x[1] = x[n], \\ -1 & \text{otherwise.} \end{cases}$$

The all one bit string 1^n is the unique global maximum. However, if $x[1]+x[n] = 0$ holds no somatic contiguous hypermutation with $r = 1$ can lead to this global optimum. It can only be reached via some x' with $x'[1] + x'[n] = 1$ implying a drastic decrease in function value. If such moves are not accepted the algorithm is unable to optimize this simple function. We are aware of this consequence due to $r = 1$. In the following, we restrict our attention to functions where these limited capabilities of CHMs with $r = 1$ are not an issue.

Note that p_1 as well as p_2 can be start and end position of the interval, respectively. As we have n^2 possible pairs of p_1 and p_2 and $n - i + 1$ possible start positions of an interval of length i, the probability to have an interval of length $2 \leq i \leq n$ equals $2(n - i + 1)/n^2$. We observe that for $i = 1$, the probability is $1/n$ since there is only one valid choice of p_1 and p_2 for each of the n possible positions. Altogether, we get the following probability distribution for

Algorithm 1. Analytical Framework

1: Choose $x \in \{0,1\}^n$ uniformly at random.
2: Create $y := \text{mutate}(x)$.
3: If $f(y) \geq f(x)$: Set $x := y$.
4: Continue at 2.

the length L of the interval considered by the mutation operator:

$$\text{Prob}(L = i) = \begin{cases} 1/n & \text{if } i = 1, \\ 2(n-i+1)/n^2 & \text{if } i \in \{2, \ldots, n\}, \\ 0 & \text{otherwise.} \end{cases}$$

Hence, we have $\text{E}(L) = (n^2 + 3n - 1)/(3n) = n/3 + 1 - o(1)$.

To analyze the performance of a mutation operator, it makes sense to embed it into an algorithmic framework. As we focus on the analysis of CHMs and omit other possible features of the artificial immune system, we use a minimal substrate as our experimentation platform. The following very simple algorithm maximizes some pseudo-Boolean function $f\colon \{0,1\} \to \mathbb{R}$ (Algorithm 1). It uses a population of size one and produces one new point via mutation. This point is accepted iff its function value is at least as large.

We intend to use CHMs and have a very simple artificial immune system. Using different mutation operators other algorithms can be obtained. In evolutionary computation standard bit mutations (SBMs) are most common [10]. There, each bit is flipped independently with mutation probability $1/n$. Plugging SBMs into our framework yields a very simple evolutionary algorithm, known as (1+1) EA [11]. Note that using other mutation operators algorithms like local search or tabu search may be obtained. Since the (1+1) EA is well investigated and well understood a multitude of results is available and may be used for comparisons [11,14,15,18,19].

As usual in the analysis of general RSHs, there is no stopping criterion in the algorithm and we investigate the first point of time when a global optimum of f is reached. This equals the number of iterations until a global optimum is found. We denote this as the optimization time of the algorithm. It corresponds to the number of function evaluations which is a common measure for the run time of RSHs. It is important to notice that the algorithm itself does not know that it has found an optimum.

In the following, let $T_{CHM,f}$ denote the optimization time of Algorithm 1 using CHMs. Let $\text{E}(T_{CHM,f})$ be its expected value. In the same way, let $T_{SBM,f}$ denote the optimization time of Algorithm 1 using SBMs and let $\text{E}(T_{SBM,f})$ be its expected value.

3 Analytical Results

We assess assets and drawbacks of CHMs in the following way. We analyze the expected optimization time when this operator is embedded in a minimal

algorithmic framework. We compare these results with the expected optimization times obtained when using SBMs known from EAs. For many objective functions $f\colon \{0,1\}^n \to \mathbb{R}$ it is at some point of time essential that the search operator is able to change exactly some bits, say b many. We call such a mutation a specific b bit mutation. In order to understand differences between CHMs and SBMs it helps to see the difference in the probabilities for such specific b bit mutations.

SBMs perform any such b bit mutation with probability $(1/n)^b \cdot (1-1/n)^{n-b} = \Theta\big(1/n^b\big)$ since b specific bits need to flip, each with probability $1/n$, and the other $n-b$ bits must not flip, each with probability $1-1/n$. Since the bits are flipped independently the result follows.

For CHMs things are entirely different. If the b bits that need to flip are not contiguous such a b bit mutation cannot be performed in a single mutation and thus has probability 0 to occur. This is due to our extreme choice $r = 1$. If, on the other hand, the b bits are contiguous then there are exactly two ways to perform this mutation: either we choose p_1 as the position of the first of these b bits and p_2 as the last or the other way round. Since each specific choice of the positions is done with probability $1/n$ and the positions are chosen independently, we have $2 \cdot (1/n) \cdot (1/n) = \Theta\big(1/n^2\big)$ as resulting probability for $b > 1$. For $b = 1$ we have probability $1/n^2 = \Theta\big(1/n^2\big)$ since there is only one valid choice of p_1 and p_2.

In the following we exclude functions where b bit mutations are necessary that cannot occur with CHMs. The fact that these mutations are impossible results from our extreme choice of $r = 1$, any choice $0 < r < 1$ yields a positive probability for such mutations. It would be inappropriate and misleading to choose an extreme framework and then demonstrate drawbacks that are uniquely due to this extreme choice. What we see for mutations that can be carried out is that their probability decreases exponentially with b for SBMs while it is completely independent of b for CHMs. This leads to the speculation that for functions where 1 bit mutations are sufficient CHMs may be outperformed by SBMs since such mutations occur with much smaller probability. On the other hand one may believe that for functions where b bit mutations with $b > 2$ are necessary CHMs may excel and that the advantage may be tremendously large for $b \gg 2$. We investigate these speculations in a rigorous way by considering appropriate example functions.

One easy consequence from the considerations above is a general lower bound on the expected optimization time for any objective function $f\colon \{0,1\}^n \to \mathbb{R}$ (except for degenerate cases). We state this simple result here since it helps us to get a better feeling for the upper bounds that we derive in the following.

Theorem 1. *Let $f\colon \{0,1\}^n \to \mathbb{R}$ be given such that*

$$\mathit{Prob}\left(x < \max\{f(y) \mid y \in \{0,1\}^n\}\right) = \Omega(1)$$

holds for $x \in \{0,1\}^n$ selected uniformly at random. $E(T_{CHM}, f) = \Omega\big(n^2\big)$

Proof. Due to the assumed property of the objective function f with probability $\Omega(1)$ at least one mutation is needed to reach a global optimum. Consider any run and the very last of the CHMs leading to the global optimum reached

in this run. Due to the properties of CHMs there are either one or two choices for p_1 and p_2 leading to this global optimum. Thus such a mutation occurs with probability at most $2/n^2$. Since this holds for any run and any global optimum reached in this run the lower bound $\Omega(n^2)$ follows. □

Probably the best-known example function in the context of evolutionary algorithms [11] is OneMax. It has also been studied in the context of artificial immune systems [20]. The function value simply equals the number of bits set to 1, $\textsc{OneMax}(x) = \sum_{i=1}^{n} x[i]$. Since we maximize the unique global optimum is the all one bit string 1^n. The expected optimization time using SBMs equals $\mathrm{E}(T_{\mathrm{SBM},\textsc{OneMax}}) = \Theta(n \log n)$ [11] and this can be achieved with 1 bit mutations, only. With CHMs it takes considerably longer to optimize this simple function.

Theorem 2. $E(T_{CHM,\textsc{OneMax}}) = \Theta(n^2 \log n)$

Proof. The upper bound is easy to prove using trivial fitness layers [11]. For any $x \in \{0,1\}^n$ with $\textsc{OneMax}(x) = n - i$ there are exactly i bits set to 0. If exactly one of these bits is flipped the function value is increased by exactly 1. Such a mutation occurs with probability $1/n^2$ for each of the i bits. Thus, the waiting time to increase the function value from $n - i$ to at least $n - i + 1$ is bounded above by n^2/i. Since function values cannot decrease due to the strict selection employed the expected optimization is bounded above by $\sum_{i=1}^{n} n^2/i = n^2 \sum_{i=1}^{n} 1/i = O(n^2 \log n)$.

For the lower bound we make use of simple drift arguments [14]. For any specific b bit mutation the probability is $\Theta(1/n^2)$. To make an advance by j we need to mutate $j + a$ 0-bits from 0 to 1 and a 1-bits from 1 to 0 (for any $a \in \mathbb{N}$). If such a mutation is to have a positive probability the bits need to be arranged in an appropriate way. Initially, the probability for this decreases exponentially with j and a since the bits are initialized uniformly at random. Later on, the probability to have 0-bits decreases. Thus, there is a constant $c > 1$ such that the probability for an increase by j is bounded above by $O(i/(n^2 \cdot c^j))$ if the current number of 0-bits equals i. We see that the drift is bounded above by $O\left(\sum_{j=1}^{n-i} j \cdot (i/(n^2 \cdot c^j))\right) = O(i/n^2)$ in this situation. This implies a lower bound of $\Omega(n^2 \log n)$ on the expected run time. □

We see that we lose a factor of $\Theta(n)$ by making use of CHMs. The proof of the lower bound, however, relies on the fact that the initial bit string is chosen uniformly at random. One may wonder what happens if the initial bit string happened to be the all zero bit string 0^n. With only 0-bits in the initial bit string there is a much bigger chance for larger increases in the number of 1-bits. However, this advantage is reduced over time as the 1-bits will be distributed randomly. It is unclear if this advantage that is big in the beginning where it is easy to make progress and decreased in the end where making progress becomes much harder anyway is sufficient to yield an asymptotically smaller expected

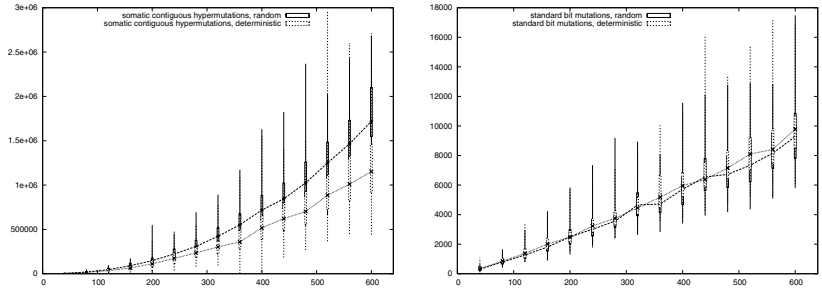


Fig. 1. Empirical data for OneMax comparing random with deterministic initialization in 0^n. On the left for CHMs, on the right for SBMs.

optimization time. To get an impression we provide results of experiments. We perform 100 independent runs of the algorithm using random initialization as well as deterministic initialization in 0^n. We plot the results using box-and-whisker plots providing the mean together with the minimum, maximum, upper and lower quartile of the 100 runs for $n \in \{40, 80, 120, \ldots, 600\}$ (Figure 1). For the sake of completeness we plot analogous data for the algorithm using SBMs. Note the different scaling that is due to the much smaller expected optimization time.

We observe that CHMs benefit from initialization in 0^n whereas for SBMs there is hardly any difference. If this difference is so large as to constitute an asymptotic difference in the expected optimization time is impossible to tell from this graph, of course.

The result on OneMax may lead to the belief that CHMs increase the expected optimization time by a factor of $\Theta(n)$ for objective functions where mutations of single bits are responsible for optimization. We demonstrate that things are not so simple by considering another well-known example function, namely LeadingOnes. The function LeadingOnes yields as function value the number of consecutive 1-bits counted from left to right, $\textsc{LeadingOnes}(x) = \sum_{i=1}^{n} \prod_{j=1}^{i} x[j]$. As for OneMax, the unique global optimum is the all one bit string 1^n. The expected optimization time using SBMs equals $\mathrm{E}\left(T_{\mathrm{SBM},\textsc{LeadingOnes}}\right) = \Theta\left(n^2\right)$ [11] and this can be achieved with 1 bit mutations, only. While the expected optimization time is larger when using CHMs it is much smaller than $\Theta\left(n^3\right)$.

Theorem 3. $E\left(T_{CHM,\textsc{LeadingOnes}}\right) = \Theta\left(n^2 \log n\right)$

Proof. Again, the upper bound is easy to prove using trivial fitness layers [11]. For any $x \in \{0,1\}^n \setminus \{1^n\}$ it suffices to mutate the leftmost 0-bit and an arbitrary number of bits to its right. If the position of the leftmost 0-bit is $n - i$, there are i bits to its right that may mutate. Such a mutation occurs with probability $2(i+1)/n^2$ for $i > 0$ and $(i+1)/n^2$ otherwise. Thus, the waiting time to increase the function value by at least 1 is bounded above by $n^2/(i+1)$. Since

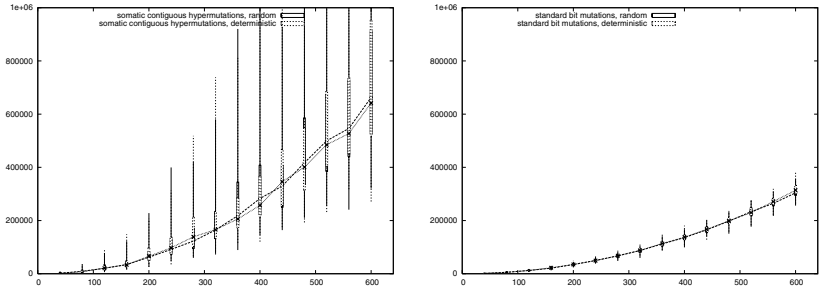


Fig. 2. Empirical data for LeadingOnes comparing random with deterministic initialization in 0^n. On the left for CHMs, on the right for SBMs.

function values cannot decrease due to the strict selection employed the expected optimization is bounded above by $\sum_{i=0}^{n-1} n^2/(i+1) = n^2 \sum_{i=1}^{n} 1/i = O\big(n^2 \log n\big)$.

For the lower bound we can observe that the bits to the right of the leftmost 0-bit are distributed uniformly at random. Thus, the probability to increase the function value by j is bounded above by $(2i)/\left(n^2 2^{j-1}\right)$. Again making use of drift arguments [14] we see that we obtain a lower bound of $\Omega\big(n^2 \log n\big)$ on the expected run time. □

For LeadingOnes, the use of CHMs implies a decrease in performance by a factor of $\Theta(\log n)$ in comparison to SBMs. As for OneMax, one may wonder what happens if the initial bit string happened to be the all zero bit string 0^n. While this constitutes an advantage in the beginning we observe that any mutation not affecting the first $i+1$ bits in a current bit string x with $\textsc{LeadingOnes}(x) = i$ will be accepted. This leads to a random distribution of the bits that are right of the leftmost 0-bit and decreases the initial advantage. We consider empirical results as we did for OneMax and display them in Figure 2.

We observe for LeadingOnes that both, CHMs as well as SBMs, are not sensitive at all with respect to the initialization method. In particular, in contradiction to our intuition, CHMs do not benefit from deterministic initialization in 0^n in a visible way. Obviously, the bits right to the leftmost 0-bit become randomly distributed so fast in comparison with the expected optimization time that no noticeable advantage is gained. This is different to the situation for OneMax. One difference between those two example functions is that the relevant bits for increasing the function value are gathered right to the leftmost 0-bit for LeadingOnes while they are randomly distributed among all bits for OneMax. This makes them more difficult to change for CHMs so that the initial advantage due to deterministic initialization in 0^n is longer preserved for OneMax leading to visible effects.

It is in some sense disappointing that initializing deterministically in 0^n helps so little. This becomes different for LeadingOnes if the bits that are right of the leftmost 0-bit do not become randomly distributed. We can enforce this by

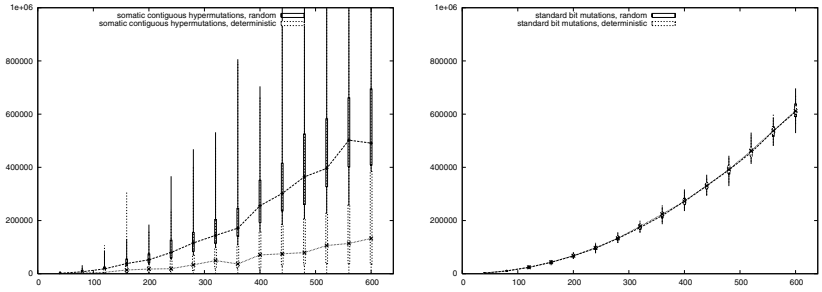


Fig. 3. Empirical data for $f = n \cdot \text{LeadingOnes} - \text{OneMax}$ comparing random with deterministic initialization in 0^n. On the left for CHMs, on the right for SBMs.

considering a different example function f, namely $f := n \cdot \text{LeadingOnes} - \text{OneMax}$.

Since the selection mechanism that we employ is insensitive to the absolute function values and reacts to the ordering of function values only we do not change anything by going from LeadingOnes to $n \cdot \text{LeadingOnes}$. Going from $n \cdot \text{LeadingOnes}$ to $f = n \cdot \text{LeadingOnes} - \text{OneMax}$ has the following effects. First, since the number of leading 1-bits comes with the factor n it is the number of leading 1-bits that dominates the function value. A bit string with a larger number of leading 1-bits has the larger function value. For bit strings with equal numbers of leading 1-bits, it is the number of 1-bits that decides. Fewer 1-bits imply larger function values. Thus, starting with 0^n, the current bit string will always be of the form $x = 1^i 0^{n-i}$ with $\text{LeadingOnes}(x) = i$ and $f(x) = n \cdot i$. This suffices to demonstrate a noticeable advantage for CHMs when started in 0^n. When initializing uniformly at random the expected optimization time is $\mathrm{E}\left(T_{\text{CHM},f}\right) = \Theta\left(n^2 \log n\right)$. The upper bound is proven analogously to the proof of Theorem 3. The lower bound follows from the lower bound for OneMax. It takes $\Omega\left(n^2 \log n\right)$ steps to reach the all zero bit string 0^n and thus no speed-up can be achieved significantly earlier.

Theorem 4. *When the algorithm is initialized deterministically in 0^n instead of initialization uniformly at random, $n^2/2 \leq E(T_{CHM,f}) \leq n^2$ holds.*

Proof. As we already discussed, the current bit string x is always of the form $x = 1^i 0^{n-i}$ with $\text{LeadingOnes}(x) = i$. This bit string is mutated into the unique global optimum 1^n if in the mutation either $a = i+1$ and $b = n$ or $a = n$ and $b = i+1$ hold. Thus, the probability for this event equals $2/n^2$ for $i \neq n-1$ and $1/n^2$ otherwise. This implies $n^2/2$ as lower bound on the expected waiting time for this event and n^2 as upper bound. □

Since we have asymptotically exact results for f and both kinds of initialization for CHMs, it is unnecessary to consider empirical results. For the sake of completeness we present empirical results as before (Figure 3).

When discussing probabilities of specific b bit mutations we observed that CHMs can be advantageous when feasible b bit mutations for $b > 2$ are needed and that this advantage grows exponentially with b. In the following, we consider a function designed for this purpose in a different context and algorithm.

We consider a function similar to LeadingOnes but replacing the role of single leading 1-bits by blocks of leading 1-bits of equal length b. Thus, for $b = 1$ we have LeadingOnes where single bit mutations are sufficient, for $b > 1$ we have a function where the function value can be increased by a mutation of b bits. Such an example function could be called LeadingOnesBlocks$_b$ or LOB_b, for short. Since we want to simplify analysis we would like to make sure that exactly b bit mutations are needed. This can be achieved by moving to $n \cdot \mathrm{LOB}_b - \mathrm{OneMax}$ just as we moved from LeadingOnes to f.

As we mentioned above, such a function was introduced in a different context. Since we want to avoid 'inventing' new example functions (where no results for comparisons are known) we stick to the definition of this known example function even though it is a little bit more involved than what is actually needed here. The function is called $\mathrm{CLOB}_{b,k}$ (short for ConcatenatedLeadingOnesBlocks$_{b,k}$) and is defined as k independent copies of $n \cdot \mathrm{LOB}_b$ where the complete function value is given by adding up the function values of the k copies. The function was originally introduced by Jansen and Wiegand [15] for the analysis of a simple cooperative coevolutionary algorithm, called the CC (1+1) EA. We proceed with a formal definition of this fitness function and a discussion of its main properties.

The function $\mathrm{CLOB}_{b,k} \colon \{0,1\}^n \to \mathbb{R}$ is defined for values $b, k \in \mathbb{N}$ with $n/k \in \mathbb{N}$ and $n/(bk) \in \mathbb{N}$. For any $x \in \{0,1\}^n$ the function value is given by

$$\mathrm{CLOB}_{b,k}(x) = n \cdot \left(\sum_{h=1}^{k} \sum_{i=1}^{n/(bk)} \prod_{j=1}^{i \cdot b} x\left[(h-1) \cdot (n/k) + j \right] \right) - \mathrm{OneMax}(x).$$

Obviously, it is defined as sum of k independent copies of the same function, operating on consecutive disjoint pieces of the bit string x, each of length n/k. To simplify notation a bit in the following we define $l := n/k$. Note that we have $l \in \mathbb{N}$. The function has the all one string 1^n as its unique global optimum.

Think of $x = x[1]x[2]\cdots x[n] \in \{0,1\}^n$ as devided into k pieces $x^{(1)}, x^{(2)}, \ldots, x^{(k)}$ with $x^{(i)} = x\,[(i-1)\cdot l + 1]\,x\,[(i-1)\cdot l + 2]\cdots x\,[i \cdot l] \in \{0,1\}^l$ for each $i \in \{1, 2, \ldots, k\}$. Each piece $x^{(i)}$ can be thought of as being divided into l/b consecutive disjoint blocks of length b each. In each piece the number of these blocks completely set to 1 is counted from left to right stopping with the first block different from 1^b. For each of these leading 1-blocks the function value is increased by n. We add this up for all k pieces and subtract the number of 1-bits.

Increasing the number of pieces k while keeping the length of the blocks b fixed decreases the length of each piece and decreases the number of blocks in each piece. Increasing b while keeping k fixed decreases the number of blocks in each piece without changing the length of the pieces.

It is known that the expected optimization time of the (1+1) EA on $\mathrm{CLOB}_{b,k}$ equals $\mathrm{E}\left(T_{(1+1)\ \mathrm{EA},\mathrm{CLOB}_{b,k}}\right) = \Theta\left(n^b\,(l/b + \ln k)\right)$ [15]. The cooperative

coevolutionary algorithm analyzed by Jansen and Wiegand, the CC (1+1) EA, achieves an expected optimization time of $\mathrm{E}\left(T_{\text{CC (1+1) EA,CLOB}_{b,k}}\right) = \Theta\left(k \cdot l^b \left(l/b + \ln k\right)\right)$ [15] beating the (1+1) EA by a factor of $\Theta\left(k^{b-1}\right)$. Note that this algorithm is provided with the information about the number k of pieces and their length l.

Theorem 5. *Let for $n \in \mathbb{N}$ the parameters $b, k \in \mathbb{N}$ be given with $n/(bk) \in \mathbb{N}$. $E\left(T_{CHM,\text{CLOB}_{b,k}}\right) = O\left(n^2 \log n\right)$*

Proof. We remember the k pieces $x^{(1)}, x^{(2)}, \ldots, x^{(k)}$ of length $l := n/k$ each that together form $x \in \{0,1\}^n$. We know that on average after $O\left(n^2 \log n\right)$ steps the optimum of OneMax is reached. We conclude that on average for each of the k pieces after that many steps a bit string of the form $1^{i \cdot b}0^{l-i \cdot b}$ (with possibly different values i for the different pieces) is reached. Then we are in a situation very similar to $f = n \cdot \text{LeadingOnes} - \text{OneMax}$. For each of the j pieces that are different from 1^l there is always a mutation creating 1^l. Each of these mutations occurs with probability at least $1/n^2$. We thus obtain $\sum_{j=1}^{k} n^2/j = O\left(n^2 \log k\right)$ as upper bound. Since $k \leq n$ holds, this proves the upper bound. □

We observe that when we initialize deterministically in 0^n we are in this special situation having $1^{i \cdot b}0^{l-i \cdot b}$ in each of the k pieces right at the start. Thus, the first phase where we wait for this to happen is empty. This reduces the upper bound that we are able to prove from $O\left(n^2 \log n\right)$ to $O\left(n^2 \log k\right)$.

Like for the other example functions we present empirical data (Figure 4). We choose (quite arbitrarily) $k \in \{5, 10\}$ and $b \in \{4, 8\}$, so that we consider in total four version of $\text{CLOB}_{b,k}$, namely $\text{CLOB}_{4,5}$, $\text{CLOB}_{8,5}$, $\text{CLOB}_{4,10}$, and $\text{CLOB}_{8,10}$. Note that for $\text{CLOB}_{8,10}$ we need $n/80 \in \mathbb{N}$ to hold so that we only present results for $n \in \{80, 160, \ldots 560\}$. Since the expected optimizations times when using SBMs are extremely large ($\Omega\left(n^4\right)$ for $b = 4$ and $\Omega\left(n^8\right)$ for $b = 8$) we do not present actual results for the (1+1) EA here. We remark that when performing such runs using SBMs and considering for example $\text{CLOB}_{4,5}$ the minimum observed optimization time in 100 runs for $n = 40$ when using SBMs $(3,896,445)$ is significantly larger than the maximum observed optimization time in 100 runs for $n = 600$ when using CHMs $(3,280,958)$. The maximum observed optimization time in 100 runs on $\text{CLOB}_{4,5}$ using CHMs for the same value $n = 40$ even equals only $8,328$. The comparison is pointless, performing the runs for the (1+1) EA for these values of b and k not even feasible.

Note that only for $f = n \cdot \text{LeadingOnes} - \text{OneMax}$ we can actually prove that the expected optimization time using CHMs decreases significantly when we initialize deterministically in the all zero bit string 0^n. For all other functions we do not have a lower bound for random initialization that is asymptotically larger than an upper bound for initializing deterministically in 0^n. We considered results of some runs and found the empirical data mostly inconclusive. We can improve the situation a bit by changing the form of presentation. Instead of displaying the actual observed mean optimization times we consider the quotients

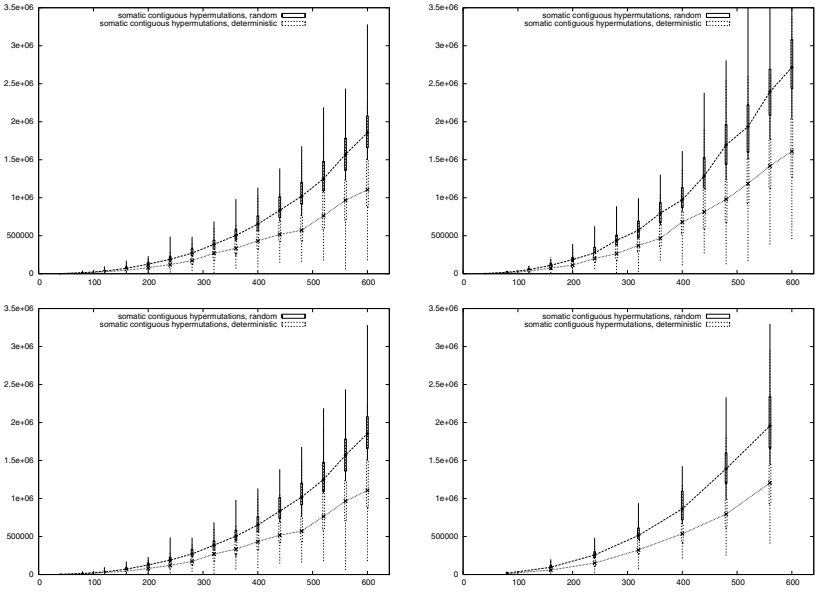


Fig. 4. Empirical data for $\mathrm{CLOB}_{b,k}$ comparing random with deterministic initialization in 0^n. On the top left for $\mathrm{CLOB}_{4,5}$, on the top right for $\mathrm{CLOB}_{8,5}$, on the bottom left for $\mathrm{CLOB}_{4,10}$, on the bottom right for $\mathrm{CLOB}_{8,10}$.

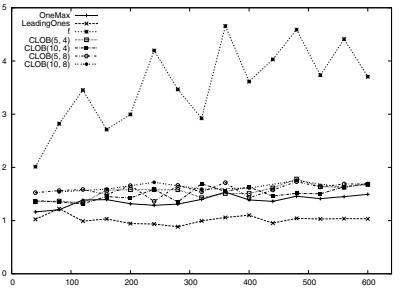


Fig. 5. Quotients of mean observed run times using CHMs together with random initialization and deterministic initialization in 0^n

of the mean optimization times for the two different forms of initialization. So, if $M_r(n)$ is the observed mean optimization time when using random initialization and $M_d(n)$ is the observed mean optimization time when initializing deterministically in 0^n (both for string length n) we plot $M_r(n)/M_d(n)$ as a function of n. If the expected optimization times are of the same order of growth this function should be constant (with random fluctuations), otherwise it should be growing. We consider the curves for all example functions and display them in Figure 5.

One may guess from the graphs in Figure 5 that indeed $f = n \cdot \text{LeadingOnes} - \text{OneMax}$ is the only function where the expected optimization time for CHMs is asymptotically reduced when going from random initialization to deterministic initialization in 0^n. We leave an actual proof as subject to future research.

4 Conclusions

CHMs are used in artificial immune systems applied to the task of optimization. We embedded this operator in a simple algorithmic framework and performed a rigorous performance analysis. This yielded proven results on the expected optimization time shedding light on specific properties and assets of this kind of mutation. We highlighted some of these aspects by presenting analyses for well-known example functions where the performance of CHMs can easily be compared with the performance of SBMs. Our analysis concentrates on an extreme version of CHMs and compensates for this choice by restricting the analysis to objective functions that are feasible for this operator. We observe that the probability for specific feasible b bit mutations ($b > 1$) is much smaller for CHMs than for SBMs. This indicates that CHMs may lose in comparison with SBMs on functions where mutations of single bits suffice. On one hand, we prove this to be the case for OneMax, but on the other hand we demonstrate that things are not that simple using LeadingOnes. We investigate the role of initialization and prove for one specific example that advantageous starting points can speed up optimization considerably for CHMs while having nearly no impact on SBMs. This, however, seems not to be true in general as empirical results on the other objective functions under consideration indicate. Finally, for a family of example functions that require mutations of many bits simultaneously for optimization we proved that CHMs can outperform SBMs drastically even by an exponential performance gap.

Many open questions deserve further attention. While it is intuitively clear that initializing in 0^n instead of uniformly at random is advantageous for the example functions considered we could actually only prove this for one function. Empirical results indicate that there may not even be an asymptotic decrease of the expected optimization time due to deterministic initialization. Finding out and rigorously proving what is the case would be interesting. We pointed out for OneMax and LeadingOnes that even when initializing deterministically in 0^n the 'irrelevant bits' become randomly distributed. Determining the random distribution is subject of future research. All our results are about 'artificial' example functions often considered when 'new' search heuristics are subject of rigorous analyses. These functions help to assess the assets and drawbacks of the heuristics under consideration and get a clearer and better founded understanding of their properties. These results are useful but only a first step. Results on classes of functions, typical situations, and combinatorial optimization problems need to be done. Finally, we pointed out that the kind of CHMs investigated here deviates from the operator introduced by Kelsey and Timmis [16] in order to reduce the effects due to positional bias. It would be interesting to see the expected optimization times of their operator for the functions investigated here.

References

1. Aarts, E., Lenstra, J.K. (eds.): Local Search in Combinatorial Optimization. Princeton University Press, Princeton (2003)
2. Bull, P., Knowles, A., Tedesco, G., Hone, A.: Diophantine benchmarks for the B-cell algorithm. In: Bersini, H., Caarneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 267–279. Springer, Heidelberg (2006)
3. Burnet, F.M.: The Clonal Selection Theory of Acquired Immunity. Cambridge University Press, Cambridge (1959)
4. Clark, E., Hone, A., Timmis, J.: A Markov chain model of the b-cell algorithm. In: Jacob, C., et al. (eds.) ICARIS 2005. LNCS, vol. 3627, pp. 318–330. Springer, Heidelberg (2005)
5. Cortés, N.C., Coello, C.A.C.: Multiobjective optimization using ideas from the clonal selection principle. In: Cantú-Paz, E., et al. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 158–170. Springer, Heidelberg (2003)
6. Cutello, V., Nicosia, G., Pavone, M.: Exploring the capability of immune algorithms: A characterization of hypermutation operators. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 263–276. Springer, Heidelberg (2004)
7. Dasgupta, D., Niño, L.F.: Immunological Computation: Theory and Applications. Auerbach (2008)
8. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, Heidelberg (2002)
9. de Castro, L.N., Zuben, F.J.V.: Learning and optimization using the clonal selection principle. IEEE Trans. on Evolutionary Computation 6(3), 239–251 (2002)
10. de Jong, K.A.: Evolutionary Computation: A Unified Approach. MIT Press, Cambridge (2006)
11. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science 276, 51–81 (2002)
12. Droste, S., Jansen, T., Wegener, I.: Upper and lower bounds for randomized search heuristics in black-box optimization. Theory of Computing Systems 39, 525–544 (2006)
13. Droste, S., Wiesmann, D.: On the design of problem-specific evolutionary algorithms. In: Ghosh, A., Tsutsui, S. (eds.) Advances in Evolutionary Computing, pp. 153–173. Springer, Heidelberg (2003)
14. He, J., Yao, X.: A study of drift analysis for estimating computation time of evolutionary algorithms. Natural Computing 3(1), 21–35 (2004)
15. Jansen, T., Wiegand, R.P.: The cooperative coevolutionary (1+1) EA. Evolutionary Computation 12(4), 405–434 (2004)
16. Kelsey, J., Timmis, J.: Immune inspired somatic contiguous hypermutations for function optimisation. In: Cantú-Paz, E., et al. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 207–218. Springer, Heidelberg (2003)
17. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (1983)
18. Oliveto, P.S., Witt, C.: Simplified drift analysis for proving lower bounds in evolutionary optimization. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 82–91. Springer, Heidelberg (2008)
19. Wegener, I.: Methods for the analysis of evolutionary algorithms on pseudo-boolean functions. In: Sarker, R., et al. (eds.) Evolutionary Optimization. Kluwer Academic Publishers, Dordrecht (2001)

20. Zarges, C.: Rigorous runtime analysis of inversely fitness proportional mutation rates. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 112–122. Springer, Heidelberg (2008)
21. Zarges, C.: On the utility of the population size for inversely fitness proportional mutation rates. In: Garibay, I., et al. (eds.) Foundations of Genetic Algorithms (FOGA), pp. 39–46. ACM Press, New York (2009)

Comparing Different Aging Operators

Thomas Jansen[1,⋆] and Christine Zarges[2]

[1] University College Cork, Department of Computer Science, Cork, Ireland
t.jansen@cs.ucc.ie
[2] TU Dortmund, Fakultät für Informatik, LS 2, 44221 Dortmund, Germany
Christine.Zarges@tu-dortmund.de

Abstract. Quite different search heuristics make use of the concept of assigning an age to search points and systematically remove search points that are too old from the search process. In evolutionary computation one defines some finite maximal lifespan and assigns age 0 to each new search point. In artificial immune systems static pure aging is used. There a finite maximal lifespan is defined but new search points inherit the age of their origin if they do not excel in function value. Both aging mechanisms are supposed to increase the capabilities of the respective search heuristics. A rigorous analysis for two typical difficult situations sheds light on similarities and differences. Considering the behavior on plateaus of constant function values and in local optima both methods are shown to have their strengths. A third aging operator is introduced that provably shares the advantages of both aging mechanisms.

1 Introduction

General randomized search heuristics (RSHs) are a class of algorithms used in practical settings where there is no time or expertise to develop problem-specific solutions. There are many different search heuristics like randomized local search [15] or tabu search [9]. Many such heuristics are inspired by natural processes (like simulated annealing [1]), biological systems (like artificial immune systems (AISs) [3,4]) or biological processes (like evolutionary algorithms (EAs) [6]). RHSs tend to be algorithmically simple while exhibiting complex random behavior that is notoriously difficult to analyze in a rigorous way. In practical applications the canonical and simple general variants often exhibit unsatisfying performance. It is therefore common practice to enhance their search capabilities by adding more complex mechanisms and modifying the search strategies. This way they are tailored towards the concrete search problem at hand. Clearly, this increased complexity complicates the task of a theoretical analysis even more. Since it is highly desirable to obtain a clear understanding of the working principles and properties of the different operators employed it makes sense to study their effects in isolation. Such analyses serve as building blocks of a theory of RSHs.

⋆ This material is based in part upon work supported by the Science Foundation Ireland under Grant No. 07/SK/I1205.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 95–108, 2009.

One task search heuristics like AISs and EAs are applied to is optimization. We consider the case where for the search space $\{0,1\}^n$ of bit strings of fixed length n an objective function $f\colon \{0,1\}^n \to \mathbb{R}$ is given and some $x^* \in \{0,1\}^n$ with $f(x^*) = \max\{f(x) \mid x \in \{0,1\}^n\}$ is sought. Since RSHs are most often implemented on standard binary computers the restriction to $\{0,1\}^n$ is not fundamental. The objective function f defines a search landscape that can exhibit different features that make the task of optimization difficult. Many kinds of difficult features are known [11], we concentrate on two common features.

Local optima are points in the search space where no neighbor has a strictly larger function value. They are a meaningful concept if the concrete notion of a neighborhood is connected to the search behavior of the RSHs. In the search space $\{0,1\}^n$ Hamming neighborhood often has this property. Almost all RSHs consider search points with a small Hamming distance to a current search point with much higher probability than search points with larger Hamming distances. This is almost necessarily the case since the number of points with Hamming distance d increases exponentially in d (for not too large values of d). It is a common experience that search heuristics get trapped in local optima.

A plateau is a set of neighboring points in the search space with equal function value. Again, we assume that Hamming distance is appropriate and consider two points to be neighbors if their Hamming distance equals 1. Plateaus turn out to be obstacles if they are not very small since they give no hint at all in what direction to search. Therefore, a kind of random walk on the plateau has to be performed that can be quite time consuming.

Different methods and tricks have been introduced to various search heuristics to overcome these and other difficulties. We concentrate on aging that has been introduced for two different kinds of heuristics, AISs [3,4] and EAs [6].

AISs are based on the immune systems of vertebrates and derive from various immunological theories, among those the clonal selection principle, negative selection, immune networks and danger theory. Even though anomaly detection and classification are the most natural applications for AISs they are also often applied to the problem of optimization. There is a variety of AISs, often built on the clonal selection principle, designed for this problem [2,5,14].

EAs mimic the process of natural evolution. They consider a set of points in the search space as a population that undergoes random variation using variation operators that are called crossover and mutation. They apply selection for reproduction to randomly select members of the current population to create offspring by means of these variation operators. Most often, the population size is kept constant by applying selection for replacement deciding on the members of the population in the next generation. Among others optimization is one of the most popular applications.

One important goal is to hinder the current set of search points from becoming too similar. Preserving some degree of diversity is thought to be helpful in many situations including avoiding getting stuck in local optima and exploring plateaus efficiently. One mechanism used to achieve this is aging.

In evolutionary computation, aging is used by assigning each new offspring age 0. The age is increased in each generation by 1. In selection for replacement the age is taken into account, search points exceeding a pre-defined maximal age τ are removed from the population. The extreme cases $\tau = 0$ and $\tau = \infty$ are known as comma selection and plus selection, respectively [16]. We call this kind of aging evolutionary aging here.

In AISs a different kind of aging is more common [2]. Again, each offspring is assigned an age that is increased by 1 in each round. As in EAs, search points exceeding a pre-defined maximal age τ are removed from the set of current search points. But the age of a new search point is only set to 0, if its function value is strictly larger than the function value of the search point it was derived from. Otherwise it inherits the age of this search point. This kind of aging is usually called static pure aging.

It is not obvious which kind of aging is to prefer in which situation. We shed light on this question by analyzing the influence of the performance in two paradigmatic situations, considering local optima and plateaus. We do this by embedding the two different aging operators in a minimal algorithmic framework that aims at being as simple as possible. We consider as testbed two example functions, one with a clearly structured local optimum and one with a small plateau. Using this setting we are able to prove where assets and drawbacks of the two aging operators can be found. Moreover, inspired by these findings we define a third aging operator that we call genotypic aging. This simple variant of static pure aging combines the specific strengths of both aging operators. We prove this by analyzing its performance in the same testbed.

Formal definitions of the algorithmic framework, the two different aging operators and the two example functions serving as testbed can be found in the next section. In Section 3 we derive results for both aging operators on an example function with a local optimum. In Section 4 we consider their performance on a plateau. We summarize our findings, define genotypic aging and analyze its performance on both example functions in Section 5. Finally, we conclude and point out possible directions of future research (Section 6).

2 Aging Operators, Algorithms and Analytical Testbed

We aim at analyzing the influence of aging operators on the performance of RSHs. This is most easily done if the aging operators can be studied as much in isolation as possible. We achieve this by implementing them in a minimal algorithmic framework (Algorithm 1). This algorithmic framework employs a collection of search points of size μ (that may be called a population in the context of EAs), lets each search point age at the beginning of each round, selects in each round (or generation) a single search point for variation, applies a standard mutation operator to it, decides on its age using the aging operator under consideration, removes search points due to their age, selects from the remaining points (at most $\mu+1$ if no search point is removed) the μ best according to their function values and fills up the collection of the search points by generating new

Algorithm 1. Algorithmic Framework

1. **Initialization**
   Set $P := \emptyset$. Repeat the following μ times.
   Select $x \in \{0,1\}^n$ uniformly at random. Set $x.\text{age} := 0$. Set $P := P \cup \{x\}$.
2. **Aging: Growing older**
   For all $x \in P$
   Set $x.\text{age} := x.\text{age} + 1$.
3. **Selection**
   Select $y \in P$ uniformly at random.
4. **Variation**
   Independently for each $i \in \{1, 2, \ldots, n\}$
   With probability $1/n$ set $y[i] := 1 - y[i]$.
5. **Aging: Age of new search point**
   SetAge(x, y) {*Sets the age of y depending on x.*}
6. **Aging: Removal due to age**
   Set $P := P \cup \{y\}$. For all $x \in P$
   If $x.\text{age} > \tau$ Then $P := P \setminus \{x\}$.
7. **Selection**
   If $|P| > \mu$ Then remove one $x^* \in P$ from P with $f(x^*) = \min\{f(x) \mid x \in \{0,1\}^n\}$.
8. **Aging: Filling-up population**
   While $|P| < \mu$ do
   Select $x \in \{0,1\}^n$ uniformly at random. Set $P := P \cup \{x\}$.
9. If Stopping Criterion not met continue at line 2.

search points uniformly at random if there are less than μ search points left. Note that the collection P is a multi-set. $P' := P \cup \{x\}$ results in a P' with $|P'| = |P| + 1$ even if already $x \in P$. Moreover, $P' := P \setminus \{y\}$ results in a P' with $|P'| \geq |P| - 1$ even if there are multiple copies of y in P. We have $|P'| = |P|$ if $y \notin P$ and $|P'| = |P| - 1$ otherwise.

One may argue that the algorithmic framework would even be simpler if we restricted ourself to the extreme case $\mu = 1$. While this is in fact not unusual (leading to the (1+1) EA [7] as an example) it is known that the presence of a population can have tremendous effects on the performance. For example, Zarges proved this for AISs and invers fitness proportional mutations [18,19]. Since both, AISs and EAs, almost always perform their search based on $\mu > 1$ search points we decide to stick with the more realistic framework.

We analyze two different aging operators, static pure aging and evolutionary aging. We give precise formal definitions of both operators as they become part of our algorithmic framework (Algorithm 1) in line 5.

Definition 1 (static pure aging)

SetAge(x, y) *If $f(y) > f(x)$ then set $y.age := 0$ else set $y.age := x.age$.*

Definition 2 (evolutionary aging)

SetAge(x, y) *Set $y.age := 0$.*

We are interested in the influence on the performance of the different aging operators (applied in line 5 of Algorithm 1). Performance of algorithms is most often measured by the computation time. Clearly, here this depends mostly on the stopping criterion. We adopt the common practice of avoiding this issue by considering the optimization time instead. Formally, we let the search heuristic run forever and analyze the first point of time when a global maximum of the objective function f is found. We measure this time by counting the number of times the main loop (lines 2–9 of Algorithm 1) is executed. As the optimization time T is a random variable we investigate its mean $\mathrm{E}(T)$ as well as sometimes information on its distribution like $\mathrm{Prob}(T < t)$ or $\mathrm{Prob}(T > t)$ for relevant points of time t.

As testbed for the analysis we make use of two example functions, one serving as an example for a fitness landscape containing a local optimum, the other serving as an example for a fitness landscape containing a plateau. The example function with a local optimum that we define follows the pattern of such example functions introduced by Jansen [12]. One example is a function called HSP_k that contains a broad and easy to find path to a local optimum and a narrow and hard to find path to the unique global optimum. The parameter k influences the likelihood of encountering the local optimum. Since this example function works only as desired for algorithms with very small μ we define a variant here with very similar properties that also works for larger values of μ.

Definition 3. *For* $n, k \in \mathbb{N}$ *with* $k = O(1)$ *the function* $\textsc{LocalOpt}_k \colon \{0,1\}^n \to \mathbb{R}$ *is defined for* $x \in \{0,1\}^n$ *by*

$$\textsc{LocalOpt}_k(x) = \begin{cases} n \cdot (i \cdot k + 1) & \text{if } x \in \left\{1^{i \cdot k} 0^{n - i \cdot k} \mid i \in \{1, 2, \ldots, \lfloor n/k \rfloor\}\right\}, \\ n \cdot (i + 1) & \text{if } x \in \left\{0^i 1^{n-i} \mid i \in \{1, 2, \ldots, \lfloor n/2 \rfloor\}\right\}, \\ n - \sum_{i=1}^{n} x[i] & \text{otherwise.} \end{cases}$$

In the vast majority of the search space the fitness value of $\textsc{LocalOpt}_k$ equals $n - \sum_{i=1}^{n} x[i]$ guiding a search heuristic towards the all zero bit string 0^n. There, two kind of paths begin. One path is entered by changing i of the right-most bits to 1 (with $i \in \{1, 2, \ldots, \lfloor n/2 \rfloor\}$). Increasing the number of 1-bits from right to left leads to the local optimum with function value $n \cdot (\lfloor n/2 \rfloor + 1)$. The other path is entered by changing exactly $i \cdot k$ of the left-most bits to 1 (with $i \in \{1, 2, \ldots, \lfloor n/k \rfloor\}$). The function value can be increased on this path in this manner until the global optimum $1^{\lfloor n/k \rfloor \cdot k} 0^{n - \lfloor n/k \rfloor \cdot k}$ with function value $n \cdot (\lfloor n/k \rfloor \cdot k + 1)$ is found. The larger k is, the longer it takes to reach this global optimum and the more difficult it becomes to find the beginning of this path. Thus, we can expect RSH to find the local optimum regularly for $k > 1$. For $k = 1$, the local and global path are entered with almost equal probability. Since the Hamming distance between the local optimum and the other path is $\Theta(n)$, any reasonable search heuristic should need an exponential number of steps to find the global optimum once the local optimum is found. For Algorithm 1 this is the case.

The other example function we consider (introduced by Jansen and Wegener [13]) contains a plateau of n points spanning a large Hamming distance of $n-1$ between the first and the last point on the plateau. We call it PLATEAU and define it as follows.

Definition 4 ([13]). *For $n \in \mathbb{N}$, the function* PLATEAU$\colon \{0,1\}^n \to \mathbb{R}$ *is defined for $x \in \{0,1\}^n$ by*

$$\textsc{Plateau}(x) = \begin{cases} n & \text{if } x \in \{1^i 0^{n-i} \mid i \in \{0,1,2,\ldots,n-1\}\}, \\ n+1 & \text{if } x = 1^n, \\ n - \sum_{i=1}^{n} x[i] & \text{otherwise.} \end{cases}$$

Like for LOCALOPT$_k$, in the vast majority of the search space the fitness values guide a search heuristic towards 0^n. There, the plateau of points $1^i 0^{n-i}$ begins. Since there are no hints in which direction better search points can be found and since all points not on the plateau have worse function value, usually RSHs will perform a random walk on the plateau until they happen to find the unique global optimum 1^n.

For LOCALOPT$_k$, results for the (1+1) EA (i. e., Algorithm 1 for $\mu = 1$ without aging) are known [12], for PLATEAU results for the (1+1) EA [13], the (μ+1) EA (i. e., Algorithm 1 without aging) [17], and some similar EAs without aging [8] are known. In the following, we derive results for the two aging operators (used within Algorithm 1) as well as for another aging operator that we introduce and define in Section 5.

3 Performance in Local Optima

We consider the example function LOCALOPT$_k$ with sufficiently large parameter k and the performance of the two aging operators. When evolutionary aging is used, we see what is expected for general RSHs. The local optimum is encountered with probability close to 1 implying a very large expected optimization time.

Theorem 1. *For $n, k \in \mathbb{N}$ with $k = O(1)$, any maximal age $\tau \in \mathbb{N}$, any population size $\mu \in \mathbb{N}$, and any number of steps $t \in \mathbb{N}$, Algorithm 1 with evolutionary aging on the function* LOCALOPT$_k$ *has optimization time at least t with probability $1 - O\big((\mu \log \mu)/n^{k-1}\big) + t/n^{(n/2)-k}\big)$. For $\mu = o\big(n^{k-1}/\log n\big)$ the expected optimization time is $\Omega(2^n)$.*

Proof. Note that for $\mu \log \mu = \Omega\big(n^{k-1}\big)$ and $t = \Omega\big(n^{(n/2)-k}\big)$ the statement becomes trivial. Thus, we assume $\mu \log \mu = o\big(n^{k-1}\big)$ in the following.

First we assume that $\tau = \infty$ holds. We discuss changes due to finite values for τ afterwards. Assume that the complete population is on the path to the local optimum, i. e., of the form $0^i 1^{n-i}$ for possibly different values of i with $i \geq 1$ for all of them. Then the global optimum can only be reached via a direct mutation

to the other path. Such a mutation has probability at most $1/n^{i+\max\{i,k\}}$ since the i right-most 1-bits need to be changed into 0-bits and we need to reach a point on the other path that has at least the same function value. We consider μn^2 generations. The probability not to make such a step in these generations is bounded below by μ/n^{k-1}. On the other hand, with probability $1 - e^{-\Omega(n)}$ in these steps the current best of the population is increased in function value. Then after on average $O(\mu \log \mu)$ generations the function value of each member of the population is increased to at least this function value since it suffices to make copies of the current best [17]. The probability to create a search point on the other path in this time is bounded above by $(\mu \log \mu)/n^{k+i}$. Then we are in the same situation as before with $i \geq 2$. Summing up the probabilities to reach the other path for $1 \leq i \leq n/2$ we obtain $O\big((\mu \log \mu)/n^{k-1}\big)$ as bound. Thus, with probability $1-O\big((\mu \log \mu)/n^{k-1}\big)$ the local optimum takes over the complete population. In this case a mutation of at least $(n/2)-k$ bits is necessary to reach the path. Such an event occurs in t steps with probability at most $O\big(t/n^{(n/2)-k}\big)$.

We still need to prove that the complete population gets on the path to the local optimum with sufficiently large probability. It is easy to see that each member of the population either reaches one of the two paths or 0^n within $O\big(\mu n^2\big)$ steps with probability $1 - e^{-\Omega(n/\log n)}$ [17]. Consider the set of search points with exactly i 1-bits. For each $i \in \{k, k+1, \ldots, n\}$ there is at most one point on the path leading to the global maximum. As long as no member of the population is on any of the two paths each point in the search space with exactly i 1-bits has equal probability to become member of the population. Thus, for each i the path to the global optimum is entered with probability at most $O\big(1/n^i\big)$. Summing up these probabilities for all $i \geq k$ we get the bound $O\big(1/n^k\big)$ on the probability to enter the path to the global optimum before either 0^n or the path to the local optimum is found. Consider some member of the population $x = 0^n$. Due to the strict selection employed we need only care about search points on one of the two paths. The probability to create some point on the path to the global optimum based on x given that a point on one of the two paths is created is $\Theta\big(1/n^{k-1}\big)$. Thus, with probability $1 - O\big(1/n^{k-1}\big)$ we get in the situation with the population on the path.

So far we ignored removes of individuals due to age. Now we consider the case of finite τ. If $\tau = o(\mu n)$ holds, with probability $1 - e^{-\Omega(n)}$ not even one of the two paths is reached [17]. Thus, we assume $\tau = \Omega(\mu n)$ in the following. Since evolutionary aging is employed, a new current member of the population starts with age 0. With probability at least $\Omega(1/\mu)$, the current best member of the population is copied. Thus, we manage to make a replica of the current best before it is removed due to its age with probability $1 - e^{-\Omega(n)}$. Then nothing changes from the line of reasoning above since any new individual that is created in line 8 of Algorithm 1 will be replaced by a copy of the current best member of the population or a search point with even larger function value with sufficiently large probability before reaching the path to the global optimum.

For sufficiently large values of τ we get $\Omega\big(n^{(n/2)-k}\big)$ as lower bound on the expected optimization time using the law of total probability. If τ is very small,

however, almost constantly new search points are created uniformly at random. This process finds the unique optimum on expectation in 2^n steps. This implies the lower bound on the expected optimization time. □

When using static pure aging instead of evolutionary aging at first sight not much is changed. If the maximal age τ is sufficient large the population will gather in the local optimum with probability close to 1 (for not too large μ). There static pure aging makes a difference. Since no new search points with larger function values can be created unless the global optimum is found we only create new search points that inherit their age. Creating a copy of a current best search point is much faster than finding the local optimum. So, no new search points enter the population. Thus, after some time, the complete population shares the very same age. This implies that at some point of time the complete population is replaced by new search points generated uniformly at random being equal to a restart. Since the path to the global optimum is found with not too small probability we expect to find the global optimum after not too many restarts.

Theorem 2. *For $n, k \in \mathbb{N}$ with $k = O(1)$, any number of search points $\mu \in \mathbb{N}$, and a maximal age $\tau \in \mathbb{N}$ with $\tau = \Omega\big(n^{k+1} + \mu n \log n\big)$, Algorithm 1 with static pure aging has expected optimization time $O\big(\tau n^{k-1}\big)$ on* $\textsc{LocalOpt}_k$.

Proof. Using insights from the proof of Theorem 1 and results on the (μ+1) EA due to Witt [17] the proof is relatively simple. Given that the local optimum is found the expected number of steps to do so is bounded by $O\big(n^2 + \mu n \log n\big)$ [17]. In the same way we see that given that the global optimum is found the expected number of steps to do so is bounded by $O\big(n^{k+1} + \mu n \log n\big)$. The term n^{k+1} stems from the fact that n mutations of exactly k specific bits are sufficient to find the global optimum. The expected waiting time for such a mutation is $\Theta\big(n^k\big)$. Note that the maximal age τ is sufficiently large to wait for this.

From the proof of Theorem 1 we know that we enter the path to the global optimum with probability $\Omega\big(1/n^{k-1}\big)$. Thus, on average after $O\big(n^{k-1}\big)$ 'restarts' of the algorithm this happens. We have such a 'restart' if all search points are removed due to their age simultaneously. This happens in the local optimum after each search point was created as a copy of the current best search point. The expected time for this to happen is $O(\tau + \mu \log \mu)$ since the time to have the population taken over by the youngest current best is $O(\mu \log \mu)$ [17] and after $O(\tau)$ generations all older current bests are removed due to their age. Moreover, the expected time to reach the local optimum is bounded by $O\big(n^2 + \mu n \log n\big)$. Together this yields $O\big(\tau + n^2 + \mu n \log n\big)$ as upper bound on the expected waiting time for a 'restart.' Multiplying it with the expected number of 'restarts' $O\big(n^{k-1}\big)$ we obtain $O\big(\tau n^{k-1} + n^{k+1} + \mu n^k \log n\big)$ as bound for this part. Since this dominates the upper bound for the expected time to reach the global optimum once the path leading to it is found we have this bound as upper bound on the expected optimization time. Remember that $\tau = \Omega\big(n^{k+1} + \mu n \log n\big)$ holds. The bound simplifies to $O\big(\tau n^{k-1}\big)$ since $\tau n^{k-1} = \Omega\big(n^{2k} + \mu n^k \log n\big)$ holds. □

4 Performance on Plateaus

We continue with analyzing the performance of the aging operators on PLATEAU (Definition 4). The proofs in this section use the method of family trees introduced by Witt [17]. A family tree $T_t(x_0)$ contains the descendants of a search point x_0 created by time $t \geq 0$ due to direct or indirect mutation. Thereby, nodes of the tree identify the search points generated and an edge (x, y) denotes that y was created by mutating x. Family trees can be used to analyze the progress of a population towards the optimum of the considered function. More precisely, an upper bound on the depth of a family tree can imply a lower bound on the optimization time as in this situation all points in the family tree are likely to be similar to the search points of the initial population P_0. Furthermore, lower bounds on the depth can imply upper bounds on the optimization time.

We first consider evolutionary aging and prove asymptotically the same bound on the optimization time $O(\mu n^3)$ as for the $(\mu{+}1)$ EA without strong assumptions on the maximal age τ. The proof follows the line of thought of Witt [17] for the $(\mu{+}1)$ EA. Therefore, we concentrate on modifications needed due to the use of aging. Analogously to Witt [17] we call search points on the plateau, i. e. points of the shape 1^i0^{n-i} for $0 \leq i \leq n$, *plateau points*. Moreover, we denote the first point in time where all search points are plateau points $T_{\text{PLATEAU}} = \min\{t \mid P_t \subseteq \{1^i0^{n-i} \mid 0 \leq i \leq n\}\}$. Considering a node $y \in T_t(x_0)$ and a path P from x_0 to y we say that y is *alive* if $y \in P_t$ and *dead* otherwise. A path P is *alive* if it contains at least one alive node. There is always at least one alive path in some family tree.

Theorem 3. *For $n, \mu \in \mathbb{N}$, max. age $\tau = \omega(\log n \cdot (n + \mu \log n))$, Algorithm 1 with evolutionary aging has expected optimization time $O(\mu n^3)$ on* PLATEAU.

Proof. The main idea is to rediscover a run of the (1+1) EA on PLATEAU on alive paths of a family tree [17]. The expected optimization time of the (1+1) EA $O(n^3)$ [7] implies a lower bound on the expected depth of such a family tree. We conclude that the expected runtime is bounded above by the sum of $\mathrm{E}(T_{\text{PLATEAU}})$ and the expected time until the family tree reaches depth $\Theta(n^3)$.

From the proof for the $(\mu{+}1)$ EA we know that $\mathrm{E}(T_{\text{PLATEAU}}) = O(\mu n + n \log n) = O(\mu n \log n)$ holds. As we use aging it remains to show that it is unlikely that any currently best search point is removed due to its age in this phase. Moreover, we have to take care of the probability to increase the currently best known function value in order to obtain the same bound on $\mathrm{E}(T_{\text{PLATEAU}})$. Afterwards we prove that the expected time until the family tree reaches some depth k can be bounded above by $O(\mu k)$ once all search points in the population are plateau points. This implies an upper bound on the optimization time of $O(\mu n^3)$. Search points outside the plateau are immediately deleted after time T_{PLATEAU}.

We first show that the maximal age τ is sufficiently large to carry over the upper bound on $\mathrm{E}(T_{\text{PLATEAU}})$. Assume that the population contains no plateau points. Then, in order not to remove the currently best search point due to its age, it suffices to select a best search point and produce a replica. The probability is at least $1/\mu$ for the first and $(1 - 1/n)^n \geq 1/(2e)$ for the second event. The

probability not to have an improvement in $4e\mu$ steps is at most $1/2$ due to Markov's inequality. Moreover, the probability not to have such an event in $\log n$ rounds of $4e\mu$ steps each can be bounded above by $(1/2)^{\log n} = 1/n$ and thus with $\tau = \omega(\mu \log n)$ a success probability of $1 - n^{-\omega(1)}$ follows.

The expected time for increasing the currently best function value is bounded by $O(n + \mu \log n)$ as the probability is i/μ to choose one of the i best search points and $1/n \cdot (1 - 1/n)^{n-1} \geq 1/(en)$ to increase the function value. As above we see that $\tau = \omega(\log n \cdot (n + \mu \log n))$ suffices to obtain a success probability of $1 - n^{-\omega(1)}$. Since we need at most n such improvements to reach the plateau, the probability not to reach the plateau is bounded by $1 - n \cdot n^{-\omega(1)} = 1 - n^{-\omega(1)}$.

Once a plateau point has been created, the number of those points is increased if we choose a plateau point and produce a replica. The expected time until the population only consists of copies of this point is $O(\mu \log \mu) = O(\mu \log n)$ [17]. We see that our choice of τ is large enough for moving all points on the plateau.

The upper bound on the expected time to reach depth k in a family tree remains to be shown. Let S_t be the set of alive search points in $T_t(x_0)$ that always have an alive descendant until time $T_{\text{PLATEAU}} + 4e\mu k$. Let L_t denote the maximum depth of $x \in S_t$ in $T_t(x_0)$. L_t increases if a search point x with depth $(x) = L_t$ is selected, a plateau point x' is created and x is deleted before x'. The probability for the first two events is $1/(2e\mu)$ as it suffices to create a replica of x. For the third event aging comes into play. The probability that x is deleted before x' is $1/2$ due to selection as both search points have the same function value. It is not possible that x is deleted before x' due to its age as we use evolutionary aging. It remains to show that x' is generated with high probability before x is deleted due to its age. The expected time for generating x' can be bounded above by $2e\mu$ and thus we can show that for $\tau = \omega(\log n \cdot (n + \mu \log n))$ the probability not to generate x' can be bounded by $1 - n^{-\omega(1)}$. This yields an expected number of $4e\mu k = O(\mu k)$ steps for reaching depth k. □

We have seen that using evolutionary aging does not change much compared to the corresponding algorithm without aging on PLATEAU given that the maximal age τ is sufficiently large. This is not the case when static pure aging is used as descendants of plateau points that are plateau points themself inherit the age of the parent. This may lead to the extinction of the whole population on the plateau if the maximal age τ is not sufficiently large. We formalize this in the next theorem.

Theorem 4. *For $n \in \mathbb{N}$ let $\alpha(n) = \omega(1)$ and $\alpha(n) = O\big((n/\ln n)^{4/3}\big)$. Then for $\mu, \tau \in \mathbb{N}$ with $\mu = \text{poly}(n)$, $\tau = \omega(\log n \cdot (n + \mu \log n))$ and $\tau = O\big(n^3/(\alpha(n) \ln n)\big)$, with probability $1 - n^{-\Omega\left(\sqrt{\alpha(n)}\right)}$, the optimization time of Algorithm 1 with static pure aging on* PLATEAU *is $n^{\Omega\left(\sqrt{\alpha(n)}\right)}$.*

Proof. In the proof of Theorem 3 we have seen that $\mathrm{E}\,(T_{\text{PLATEAU}}) = O(\mu n \log n)$ holds. As the lower bound on τ here matches that of Theorem 3, we can assume that all search points in the population are plateau points. Due to Witt [17] all populations until (and including) time T_{PLATEAU} only contain search points with

at most $3n/5$ ones with probability $1 - 2^{\Omega(\sqrt{n})}$, implying that all search points so far have linear Hamming distance to the optimum.

We now show that the population becomes extinct on the plateau with probability $1 - n^{-\Omega\left(\sqrt{\alpha(n)}\right)}$ after at most $\tau = O\big(n^3/(\alpha(n)\ln n)\big)$ steps. Assume that $x \in P_t$, $t > T_{\text{PLATEAU}}$, is selected. We denote a step as *relevant* iff it generates a plateau point $y \neq x$. Recall that y inherits the age of x in this case. We bound the number of relevant steps within overall τ steps and the step size of such a relevant step from above. Finally we show that this leads with probability converging to 1 super-polynomially fast to the extinction of the population before the optimum is reached and thus, to a super-polynomial lower bound on the optimization time.

We first consider only a single search point in the population. The probability for having a relevant step is at most $2/n$ as either the leftmost 0-bit or the rightmost 1-bit has to flip. Thus, in a phase of $O\big(n^3/(\alpha(n)\ln n)\big)$ steps there are at most $O\big(n^2/(\alpha(n)\ln n)\big)$ relevant steps with probability $1 - 2^{-\Omega\left(n^2/(\alpha(n)\ln n)\right)}$ using Chernoff bounds. Here, the upper bound on $\alpha(n)$ is needed.

A relevant step with step size b requires at least a b bit mutation which happens with probability at most $2/n^b$. The probability of not having a relevant step with step size at least $3 + \sqrt{\alpha(n)}$ in $\tau = O\big(n^3/(\alpha(n)\ln n)\big)$ steps is then bounded below by $1 - \tau \cdot O\Big(n^{-(3+\sqrt{\alpha(n)})}\Big) = 1 - n^{-\Omega\left(\sqrt{\alpha(n)}\right)}$.

We assume pessimistically that all relevant steps have step size $3 + \sqrt{\alpha(n)}$ and that the last such step reaches the optimum. The resulting random process corresponds to a fair random walk on at most $m := \Big((2n/5) - 3 - \sqrt{\alpha(n)}\Big)/(3 + \sqrt{\alpha(n)})$ states. We want to bound from above the probability to overcome the distance m, i. e. finding the optimum, in $s \leq n^2/(4\alpha(n)\ln n)$, steps. As we have a fair random walk, the probability for steps in either direction (say *left* and *right*) is $1/2$ and the expected number of steps for each direction in s steps equals $s/2$. We use Chernoff bounds to bound the probability sought in the following way:

$$\begin{aligned}\text{Prob}\left(\text{in } s \text{ steps distance } \geq m\right) &\leq \text{Prob}\left(\text{in } s \text{ steps } \leq \frac{s}{2} - \frac{m}{2} \text{ times left}\right)\\ &= \text{Prob}\left(\text{in } s \text{ steps } \leq \frac{s}{2}\cdot\left(1 - \frac{m}{s}\right) \text{ times left}\right) \leq e^{-\frac{s}{2}\cdot\frac{m^2}{s^2}\cdot\frac{1}{2}} = e^{-m^2/(4s)}\end{aligned}$$

A tedious but straightforward calculation yields the bound $n^{-\Omega\left(\sqrt{\alpha(n)}\right)}$ for $s = n^2/(4\alpha(n)\ln n)$ and our value of m. Using the simple union bound and that this probability is monotonically increasing in s, the probability to find the optimum within $\leq s$ steps is bounded by $s \cdot n^{-\Omega\left(\sqrt{\alpha(n)}\right)} = n^{-\Omega\left(\sqrt{\alpha(n)}\right)}$. Analogously, the probability that any search point in the population reaches the optimum in $O\big(\mu \cdot n^3/(\alpha(n)\log n)\big)$ steps is bounded by $\mu \cdot n^{-\Omega\left(\sqrt{\alpha(n)}\right)} = n^{-\Omega\left(\sqrt{\alpha(n)}\right)}$. □

We note that the plateau embedded in our example function only has size n. On bigger plateaus static pure aging has even bigger difficulties.

5 Combining the Assets of Aging

We have seen that static pure aging is able to efficiently optimize functions with local optima where evolutionary aging fails. This is due to the capability of performing restarts inherent to static pure aging. On the other hand we have seen that static pure aging fails on plateaus where evolutionary aging has no problems. This failure is caused by incompetency of static pure aging in recognizing that it is making some kind of progress even though the function values of the search points encountered do not improve. While recognizing stagnation by measuring function values helps to escape from local optima this very mechanism hinders static pure aging to be efficient on even rather small plateaus. When we consider both situations in direct comparison it is not difficult to spot a crucial difference. While the function values do not increase in both situations being stuck in a local optimum means that also no new search points with equal function values are discovered. When a random walk on the plateau is performed, there are constantly new points discovered even though they all have equal function value. We introduce an aging operator called genotypic aging that spots this difference.

Definition 5 (genotypic aging)

SetAge(x, y) *If* $f(y) \geq f(x)$ *and* $y \neq x$ *then* $y.age := 0$ *else* $y.age := x.age$.

In comparison to static pure aging we changed the condition '$f(y) > f(x)$' to '$(f(y) \geq f(x)) \wedge (y \neq x)$'. Since $f(y) > f(x)$ implies $y \neq x$, there is no change for this case. If $f(y) = f(x)$ holds we make a case distinction based on x and y. Those are called genotypes in the context of EAs motivating our choice of genotypic aging as name for this operator. Finding a new search point with equal function value is not sufficient progress to warrant setting its age to 0. This allows for random walks on plateaus beyond what the maximale age τ allows.

Theorem 5. *For* $n, k \in \mathbb{N}$ *with* $k = O(1)$*, any number of search points* $\mu \in \mathbb{N}$*, and a maximal age* $\tau \in \mathbb{N}$ *with* $\tau = \Omega\big(n^{k+1} + \mu n \log n\big)$*, Algorithm 1 with genotypic aging has expected optimization time* $O\big(\tau n^{k-1}\big)$ *on* $\textsc{LocalOpt}_k$.

Proof. We can mostly re-use ideas from the proof of Theorem 2 as the two aging mechanisms genotypic aging and static pure aging behave very similarly on $\textsc{LocalOpt}_k$. Nothing changes about the way the local optimum is found. This holds since the local optimum can be reached by a number of fitness improvements and the maximal age τ is sufficiently large to allow for each of them with a probability close to 1. Clearly, nothing changes about the probabilities to enter the two paths as these probabilities are determined by the variation operator, not by aging. The only thing we need to care about are if we still have 'restarts' when all search points are stuck in the local optimum. Note that the local optimum is a unique point $0^{\lfloor n/2 \rfloor}1^{n-\lfloor n/2 \rfloor}$ and that all points with equal or larger function value have Hamming distance $\Omega(n)$ to this point. Thus, with probability exponentially close to 1 no such search point will be encountered if $\tau = n^{o(n)}$ holds. For even larger τ the path to the global optimum may be discovered. Since this can only decrease the optimization time it does not hurt our upper bound. □

Theorem 6. *For $n, \mu \in \mathbb{N}$ and maximal age $\tau = \omega(\log n \cdot (n + \mu \log n))$ the expected optimization time of Algo. 1 with genotypic aging on* PLATEAU *is* $O(\mu n^3)$.

Proof. Again we can re-use the ideas from a previous proof (Theorem 3) as on PLATEAU the aging mechanisms genotypic aging and evolutionary aging behave very similar. For the upper bound on T_{PLATEAU} we have to consider the expected time for increasing the currently best function value. As in the proof of Theorem 3 we see that $\tau = \omega(\log n \cdot (n + \mu \log n))$ suffices to reach the plateau with the whole population with probability at least $1 - n^{-\omega(1)}$.

For the upper bound on the time to reach depth k in a family tree we have to be more careful as now replicas inherit the age of its parent. However, as the age of the replica and the parent are the same, the descendant can only be deleted before its parent due to selection which has still probability $1/2$. Therefore, we only need to care about the probability to generate a descendant x' of a plateau point x with maximal depth L_t and consider relevant steps where $x \neq x'$. The probability for a relevant step is at least $1/n \cdot (1-1/n)^{n-1} \geq 1/(en)$ as it suffices to flip exactly one bit and thus, the waiting time is bounded by $O(n)$. With $\tau = \omega(\log n \cdot (n + \mu \log n))$ the desired probability of $1 - n^{-\omega(1)}$ follows. □

6 Conclusions and Future Work

Aging is a powerful and flexible concept that has been introduced to different kinds of search heuristics. It adds to their complexity and is supposed to enhance their search capabilities. We investigated two aging operators, static pure aging used in AISs, and evolutionary aging known from EAs. They agree in having search points removed exceeding a maximal age τ and filling up gaps in the collection of search points with randomly created points. But they differ in the way they assign an age to new search points. While evolutionary aging always assigns age 0 in static pure aging this happens only for points superior to the point they are created from. Otherwise they inherit this search point's age.

While static pure aging can escape from local optima by recognizing stagnation and performing a kind of restart it fails on plateaus where it mistakes missing progress in function values for stagnation. On the other hand, evolutionary aging recognizes the random walk on plateaus but fails to escape local optima. We proved this by means of rigorous analyses on example functions.

Inspired by the failure of static pure aging on the plateau function we introduced genotypic aging. It differs in the behavior of static pure aging with respect to new search points with fitness equal to their origin. If they are equal to their origin they inherit its age, otherwise they are initialized with age 0. For such points static pure aging always sets the initial age to the age of the origin. This small modification is sufficient to yield efficient optimization in both scenarios.

There are other kinds of obstacles search heuristics can encounter. It is subject of future research to consider other situations. It is known that aging operators can be very sensitive to the maximal age τ [10]. A more in-depth analysis of the influence of this important parameter to different aging operators is subject of

future research. All three aging operators considered here implement aging in a very specific way. A more general framework needs to be considered.

References

1. Aarts, E., Korst, J., Michiels, W.: Simulated annealing. In: Search Methodologies, pp. 187–210. Springer, Heidelberg (2005)
2. Cutello, V., Nicosia, G., Pavone, M.: Exploring the capability of immune algorithms: A characterization of hypermutation operators. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 263–276. Springer, Heidelberg (2004)
3. Dasgupta, D., Niño, L.F.: Immunological Computation: Theory and Applications. Auerbach (2008)
4. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, Heidelberg (2002)
5. de Castro, L.N., Zuben, F.J.V.: Learning and optimization using the clonal selection principle. IEEE Trans. on Evolutionary Computation 6(3), 239–251 (2002)
6. De Jong, K.A.: Evolutionary Computation. MIT Press, Cambridge (2006)
7. Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science 276(1-2), 51–81 (2002)
8. Friedrich, T., Hebbinghaus, N., Neumann, F.: Comparison of simple diversity mechanisms on plateau functions. Theoretical Computer Science (2009) (to appear)
9. Glover, F., Laguna, M.: Tabu Search. Kluwer, Dordrecht (1997)
10. Horoba, C., Jansen, T., Zarges, C.: Maximal age in randomized search heuristics with aging. In: Genetic and Evolutionary Computation Conf. GECCO (to appear, 2009)
11. Jansen, T.: On classifications of fitness functions. In: Theoretical Aspects of Evolutionary Computing, pp. 371–386. Springer, Heidelberg (2001)
12. Jansen, T.: On the analysis of dynamic restart strategies for evolutionary algorithms. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 33–43. Springer, Heidelberg (2002)
13. Jansen, T., Wegener, I.: Evolutionary algorithms — how to cope with plateaus of constant fitness and when to reject strings of the same fitness. IEEE Trans. on Evolutionary Computation 5(6), 589–599 (2002)
14. Kelsey, J., Timmis, J.: Immune inspired somatic contiguous hypermutation for function optimisation. In: Cantú-Paz, E., et al. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 207–218. Springer, Heidelberg (2003)
15. Michiels, W., Aarts, E., Korst, J.: Theoretical Aspects of Local Search. Springer, Heidelberg (2007)
16. Schwefel, H.-P., Rudolph, G.: Contemporary evolution strategies. In: European Conf. on Artificial Life (ECAL), pp. 893–907. Springer, Heidelberg (1995)
17. Witt, C.: Runtime analysis of the (μ+1) EA on simple pseudo-Boolean functions. Evolutionary Computation 14(1), 65–86 (2006)
18. Zarges, C.: Rigorous runtime analysis of inversely fitness proportional mutation rates. In: Rudolph, G., Jansen, T., Lucas, S., Poloni, C., Beume, N. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 112–122. Springer, Heidelberg (2008)
19. Zarges, C.: On the utility of the population size for inversely fitness proportional mutation rate. In: Foundations of Genetic Algorithms (FOGA), pp. 39–46 (2009)

Efficient Algorithms for String-Based Negative Selection

Michael Elberfeld and Johannes Textor

Institut für Theoretische Informatik
Universität zu Lübeck, 23538 Lübeck, Germany
{elberfeld,textor}@tcs.uni-luebeck.de

Abstract. String-based negative selection is an immune-inspired classification scheme: Given a self-set S of strings, generate a set D of detectors that do not match any element of S. Then, use these detectors to partition a monitor set M into self and non-self elements. Implementations of this scheme are often impractical because they need exponential time in the size of S to construct D. Here, we consider r-chunk and r-contiguous detectors, two common implementations that suffer from this problem, and show that compressed representations of D are constructible in polynomial time for any given S and r. Since these representations can themselves be used to classify the elements in M, the worst-case running time of r-chunk and r-contiguous detector based negative selection is reduced from exponential to polynomial.

1 Introduction

The immune system protects us from many dangerous pathogens it has never seen, and it does that by essentially playing dice: T cells are generated randomly and in large numbers, in the hope that every pathogen that infects the host is detected by at least some of these cells. However, the host must ensure that no cells are generated that would turn against itself – many severe diseases are caused by such autoimmune reactions. Hence, newborn T cells undergo the process of *negative selection*. In a special organ, the thymus, they are shown *self* proteins, which belong to the host. If a T cell detects any self protein, it is destroyed.

When Forrest et al. [1,2] analyzed the immune system as a source of inspiration for computer security, they found that the problem faced by the immune system is similar to one that today's computer systems face: It is difficult to defend a system against a previously unknown danger, such as an exploit of a new security hole. The only reliable knowledge we have is the normal behavior of the system – the equivalent of self. The idea of the negative selection classification scheme is to mimic the T cells in the immune system: Generate a set of detectors that do not match anything in self, then use these detectors to monitor the system for unusual behavior. A more precise definition of the negative selection algorithm is shown in Figure 1.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 109–121, 2009.

Algorithm NEGATIVE-SELECTION.
Input: A self-set $S \subseteq \mathcal{U}$, a monitoring set $M \subseteq \mathcal{U}$.
Output: For each element $m \in M$, either self or non-self.
1 $D \leftarrow$ Set of detectors that do not match any $s \in S$.
2 **for each** $m \in M$ **do**
3 **if** m matches any detector $d \in D$ **then**
4 **output** "m is non-self"
5 **else**
6 **output** "m is self"

Fig. 1. The generic negative selection algorithm

The generic scheme of negative selection is independent of the kind of abstraction used for self and non-self elements, which we call the *universe* $\mathcal{U}$. For instance, negative selection algorithms have been implemented for both continuous and real-valued universes. In this paper, we only consider the universe $\mathcal{U} = \Sigma^L = \{0,1\}^L$, of binary strings of length L (larger alphabets can be encoded as binary). All existing negative selection algorithms based on the string universe suffer from a worst-case exponential size of D in the total size of the input [3]. This severely limits the practical applicability of negative selection [4]. We show here, however, that this is not an intrinsic problem of string-based negative selection, at least not when r-chunk or r-contiguous detectors are used.

The structure of this paper is as follows: In the next section, we define r-chunk and r-contiguous detectors and show that it is infeasible to enumerate all of them. The subsequent two sections show that an exhaustive enumeration of all detectors is not necessary: For both r-chunk and r-contiguous detectors, we can generate compressed representations of the entire sets that are themselves usable as detectors. In the last section, we summarize our findings and discuss the implications for related work.

2 String-Based Detectors and Detector Sets

Throughout the paper, we use the following notation conventions: Let $s \in \Sigma^L$ be a string. Then $|s| = L$ is the *length* of s and $s[i, \ldots, j]$ is the substring of s with length $j - i + 1$ that starts at position i. Let $S \subseteq \Sigma^L$ be a set of strings. Then $\overline{S} = \Sigma^L \setminus S$ is its *complement*.

Now we are ready to define what r-chunk and r-contiguous detectors are.

Definition 1 (Chunk detectors). *An r-chunk detector (d, i) is a tuple of a string $d \in \Sigma^r$ and an integer $i \in \{1, \ldots, L - r + 1\}$. It matches another string $s \in \Sigma^L$ if $s[i, \ldots, i + r - 1] = d$.*

Definition 2 (Contiguous detectors). *An r-contiguous detector is a string $d \in \Sigma^L$. It matches another string $s \in \Sigma^L$ if there is an $i \in \{1, \ldots, L - r + 1\}$ with $d[i, \ldots, i + r - 1] = s[i, \ldots, i + r - 1]$.*

Definition 3 (Detector sets). *Given a self-set $S \subseteq \Sigma^L$ and an integer $r \in \{1, \ldots, L\}$, let* $\textsc{ChunkD}(S, r)$ *be the set of r-chunk detectors that do not match any string in S, and let* $\textsc{ContD}(S, r)$ *be the set of r-contiguous detectors that do not match any string in S.*

Figure 2 shows an example of a self-set together with its r-chunk detectors and r-contiguous detectors. This self-set is used several times as an example in the present paper.

Self-set S with $L = 5$ and $\lvert S\rvert = 3$	The set of 3-chunk detectors, $\textsc{ChunkD}(S, 3)$.	The set of 3-contiguous detectors, $\textsc{ContD}(S, 3)$.
01011 01010 01101	(000,1) (000,2) (000,3) (001,1) (001,2) (001,3) (100,1) (010,2) (100,3) (101,1) (011,2) (110,3) (110,1) (100,2) (111,3) (111,1) (111,2)	00000 10100 00100 10110 00110 10111 00111 11000 10000 11001 10001 11110 11111

Fig. 2. An example self-set $S \subseteq \Sigma^5$ together with the detector sets $\textsc{ChunkD}(S, 3)$ and $\textsc{ContD}(S, 3)$

All previous attempts to build efficient algorithms for generating all r-chunk and r-contiguous detectors have resulted in a worst-case complexity that is exponential in the input size. For example, there is one algorithm for determining the size of $\textsc{ContD}(S, r)$ that runs linearly in $|S|$, but exponentially in r [5].

In fact, it is impossible to build an algorithm that generates *all* detectors and is guaranteed to run in less than exponential time. Consider the worst-case size of $\textsc{ChunkD}(S, r)$ and $\textsc{ContD}(S, r)$ in terms of the size of the input, that is, $|S| \cdot L$ (the space needed for storing r is negligible). Fix $r = L$, and let S contain only one arbitrary string s. That is, the input size is L. Now, the size of both $\textsc{ChunkD}(S, r)$ and $\textsc{ContD}(S, r)$ is $2^L - 1$: any string $d \in \Sigma^L \setminus \{s\}$ is both an r-contiguous detector, and an r-chunk detector at position 1. Hence, any algorithm that completely enumerates $\textsc{ChunkD}(S, r)$ or $\textsc{ContD}(S, r)$ must take exponential time in the worst case.

In principle, the infeasibility of generating all detectors might be due to some intrinsic hardness of the problem – maybe, as argued by Timmis et al. [6], it could be equivalent to an $\mathcal{NP}$-hard problem. However, we show in this paper that this is not the case. Instead, the problem is somewhat ill-posed. As an illustrating analogy, consider the task of enumerating all strings $s \in \Sigma^L$. This takes time 2^L, but it is certainly not very hard to do – the set of all strings of length L has a trivial internal structure, which could be represented simply by the number L. We will show in the next two sections that efficient representations for both

ChunkD(S, r) and ContD(S, r) can be constructed. The detection power of these representations is equivalent to that of the entire set, but the time and space required to construct them are guaranteed to be polynomial in the size of the input.

3 Negative Selection with Chunk Detectors

Our algorithm for r-chunk detection uses *patterns* that describe sets of several detectors at once. In the context of negative selection, the usage of patterns to describe complement sets was first explored by Esponda [7]. Our contribution in this section lies in applying this idea to r-chunk detection.

Definition 4 (Prefix patterns). *A prefix pattern π is a string over the alphabet $\Sigma \cup \{\diamond\}$ that has the form $\pi = vw$, where v is a string over Σ and w is a string consisting only of $\diamond$. A string s is described by $\pi = vw$ if v is a prefix of s and $|\pi| = |s|$. The set of all strings described by π is denoted by $P(\pi)$.*

For example, the pattern $01\diamond\diamond$ describes all binary strings of length 4 that start with 01. Obviously every set of strings of length L can be described by a set of prefix patterns, since every string is itself a prefix pattern. What we are interested in is a *minimal* representation of a set of strings in terms of prefix patterns. The following lemma, which is adopted from Esponda [7] with a slightly modified proof, shows that such a representation can be constructed efficiently.

Lemma 1 (Minimal prefix patterns for the complement of S). *Given a set $S \subseteq \Sigma^r$, denote by* MinPrefix$(\overline{S})$ *the smallest set of prefix patterns that describe exactly the strings in $\overline{S}$.* MinPrefix$(\overline{S})$ *is uniquely defined, has at most $O(|S|r)$ elements and can be constructed in time $O(|S|r^2)$.*

Proof. The set MinPrefix$(\overline{S})$ is constructed by the algorithm Construct-Minimal-Prefix-Patterns (Figure 3), which iterates over the set of all prefixes of all strings in S (we ignore the trivial case that S is empty). For each prefix, the last letter is flipped. If the resulting string p_i is not a prefix of any $s \in S$, then no prefix of p_i other than p_i itself has this property. Therefore, the patterns in the resulting set D describe pairwise disjoint sets of strings, and there is no pair (π_1, π_2) of two different patterns in D that could be replaced by a single pattern π_0 such that $P(\pi_1) \cup P(\pi_2) \subseteq P(\pi_0) \subseteq \overline{S}$. Hence, there is no pattern set smaller than D that describes the same strings.

To see that the entire set $\overline{S}$ is covered by the set D, fix an arbitrary string $\hat{s} \in \overline{S}$. Since $\hat{s}$ is not a prefix of any $s \in S$, there must be a shortest prefix $\hat{p}$ of $\hat{s}$ with the same property. Then the algorithm inserts $\hat{p} \diamond \ldots \diamond$, which describes $\hat{s}$, into the set D in line 6. Since there are $|S|r$ prefixes of the strings in S, the algorithm outputs at most $|S|r$ patterns. Its time complexity depends on the efficiency of the procedure used to check if the p_i match any string in S. Using a suffix tree data structure (which can be constructed in time $O(|S|r)$ beforehand), this can be achieved in time $O(r)$ [8]. Hence, the resulting time complexity is $O(|S|r^2)$. □

For example, if $S = \{0101, 0011\}$, then algorithm CONSTRUCT-MINIMAL-PREFIX-PATTERNS yields $\{1\diamond\diamond\diamond, 011\diamond, 0100, 000\diamond, 0010\}$. This is indeed the minimal set of prefix patterns describing the complement of $\overline{S}$: No pattern can be removed from the set since some string in $\overline{S}$ would no longer be described, and no two patterns can be merged because the resulting pattern would describe strings that are not in $\overline{S}$.

Algorithm CONSTRUCT-MINIMAL-PREFIX-PATTERNS.
Input: A nonempty set $S \subseteq \Sigma^r$.
Output: The set MINPREFIX($\overline{S}$).

```
1  D ← ∅
2  for each s ∈ S do
3      for i = 1 to r do
4          p_i ← s[1, ..., i − 1]s[i]‾
5          if p_i is not a prefix of any s ∈ S then
6              D ← D ∪ {p_i ◇...◇}   (r − i ◇'s)
7  output D
```

Fig. 3. Algorithm for constructing a minimal set of prefix patterns describing the complement of S

Now we are ready to apply the idea of minimal prefix patterns to r-chunk detection. This is done by iterating over all positions $i \in \{1, \ldots, L - r + 1\}$, and considering only the substrings of the strings in S of length r at these positions. Let us denote this set by $S[i, \ldots, i + r - 1]$, then the set of all r-chunk detectors for position i is precisely the complement $\overline{S[i, \ldots, i + r - 1]}$. By generalizing the concept of r-chunk detectors to patterns, it is straightforward to construct an efficient representation of the set of all r-chunk detectors and to use it for r-chunk detection.

Definition 5 (Chunk patterns). *An r-chunk pattern is a tuple (π, i) of a prefix pattern π and an integer i. A string $s \in \Sigma^L$ is matched by (π, i) if there exists a string $d \in \Sigma^r$ described by π, such that the r-chunk detector (d, i) matches s.*

Algorithm EFFICIENT-CHUNK-NEGATIVE-SELECTION in Figure 5 provides an efficient approach for negative selection with r-chunk detectors. In lines 1 to 4 a set of r-chunk patterns D that describe exactly the r-chunk detectors of a self-set S is generated by using minimal prefix patterns. For the self-set from Figure 2, these r-chunk patterns are shown in Figure 4.

In lines 5 to 9 the set D is used to classify the elements from M. The algorithm correctly outputs for each element of M either self or non-self, since the set of r-chunk patterns is completely equivalent to the set of r-chunk detectors: If any given string is matched by an r-chunk detector, it is also matched by the pattern describing this detector; and if there is no r-chunk detector that matches a given string, then none of the equivalent r-chunk patterns match it.

D_1	D_2	D_3
(1⋄⋄,1)	(100,2)	(00⋄,3)
(00⋄,1)	(111,2)	(11⋄,3)
	(0⋄⋄,2)	(100,3)

Fig. 4. This figure shows the set of 3-chunk patterns $D = D_1 \cup D_2 \cup D_3$ that the algorithm Efficient-Chunk-Negative-Selection constructs for the self-set from Figure 2. While the set of all r-chunk detectors can grow exponentially, the maximum size of the equivalent set of r-chunk patterns is polynomially bounded.

By Lemma 1, constructing each D_i takes time $O(|S|r^2)$, and thus the time to construct D is $O(|S|r^2(L-r+1))$. Checking if a string matches a pattern takes exactly the same time as matching it against another string. Hence, we have reduced both time complexity of detector generation and the time complexity of detection itself from exponential to polynomial.

Algorithm Efficient-Chunk-Negative-Selection.
Input: A self-set $S \subseteq \Sigma^L$, an integer $r \in \{1, \ldots, L\}$ and a monitor set $M \subseteq \Sigma^L$.
Output: For each $m \in M$, either self or non-self for r-chunk detection.

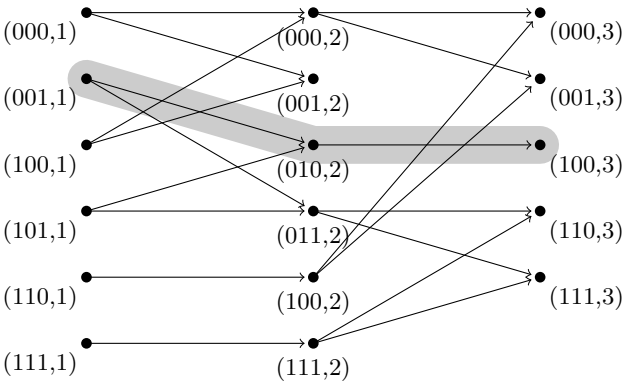
```
1  for i = 1 to L − r + 1 do
2      S_i ← {s[i, ..., i + r − 1] | s ∈ S}
3      D_i ← {(π, i) | π ∈ MinPrefix(S̄_i)}
4  D = ⋃_i D_i
5  for each m ∈ M do
6      if m matches any pattern p ∈ D then
7          output "m is non-self"
8      else
9          output "m is self"
```

Fig. 5. Our efficient algorithm for negative selection with r-chunk detectors

In the next section we present an efficient algorithm for r-contiguous detection using a slightly more complicated data structure that represents sets of contiguous detectors.

4 Negative Selection with Contiguous Detectors

In the previous section we used r-chunk patterns to describe sets of r-chunk detectors. In this section we introduce the *r-pattern graph* that is used to represent the r-contiguous detectors of a self-set $S \subseteq \Sigma^L$. The idea behind this data structure is that every r-contiguous detector can be constructed from a set of "overlapping" r-chunk detectors. As shown in Figure 6, we can construct a graph by linking every r-chunk detector to its overlapping neighbours. Now every r-contiguous detector corresponds to a path through this graph of length $L - r + 1$.



Derivation of the contiguous detector from the emphasized path:

3-chunk detectors	000
from the path:	010
	100
3-contiguous detector:	00100

Fig. 6. For the self-set from Figure 2 and $r = 3$, this figure shows how one can construct r-contiguous detectors from overlapping r-chunk detectors. The r-chunk detectors are arranged in levels and there is a directed edge from a detector d in level i to a detector d' in level $i + 1$ if $d[2, \ldots, r] = d'[1, \ldots, r - 1]$. Every path from the leftmost to the rightmost level of the graph corresponds to an r-contiguous detector.

Of course, the simple procedure depicted in Figure 6 again suffers from an exponential number of nodes in the graph. Again, we can use patterns to reduce the number of nodes to polynomial. In the rest of this section, we prove that this compression procedure does not break the correctness of the construction. Also, it is not immediately obvious how the r-pattern graph can be used for self-nonself classification. This question will be addressed at the end of this section.

Our compressed data structure, the r-pattern graph, is organized in $L - r + 1$ levels from left to right and the vertices are labeled by r-chunk patterns. Edges are only drawn between vertices of consecutive levels. A path from the leftmost to the rightmost level represents a set of r-contiguous detectors and two different paths describe different sets of r-contiguous detectors.

Definition 6 (Pattern graphs). *Let $S \subseteq \Sigma^L$ be a self-set. The r-pattern graph for S is a directed graph with $L - r + 1$ levels and constructed as follows:*

1. *Let $D = D_1 \cup \ldots \cup D_{L-r+1}$ be the set of r-chunk patterns for S, as generated by the algorithm in Figure 5. For every level i, construct $|D_i|$ vertices and label them bijectively with the patterns from D_i.*

2. *For every level i from $1, \ldots, L - r$, insert an edge from a vertex with label (π, i) to a vertex with label $(\pi', i + 1)$ if $P(\pi[2, \ldots, r]) \supseteq P(\pi'[1, \ldots, r - 1])$.*
3. *Repeatedly do the following:*
   (a) *Delete vertices from level 1 without outgoing edges.*
   (b) *Delete vertices from levels $2, \ldots, L - r$ without outgoing or incoming edges.*
   (c) *Delete vertices from level $L - r + 1$ without incoming edges.*

The construction procedure for an r-pattern graph is directly given by its definition. Given the set D beforehand, which is constructed by the algorithm in Figure 5, the worst-case runtime for lines 1 and 2 is $O(r|D|^2)$ and $O(|D|^3)$ for line 3. Figure 7 shows an example of an r-pattern graph.

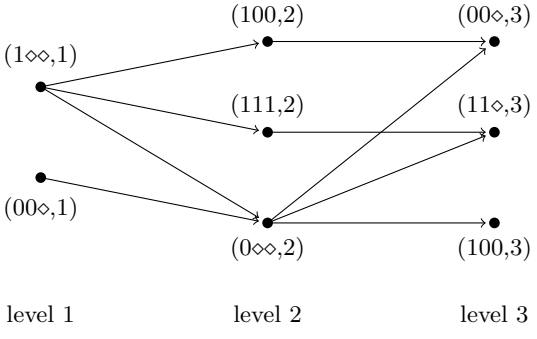
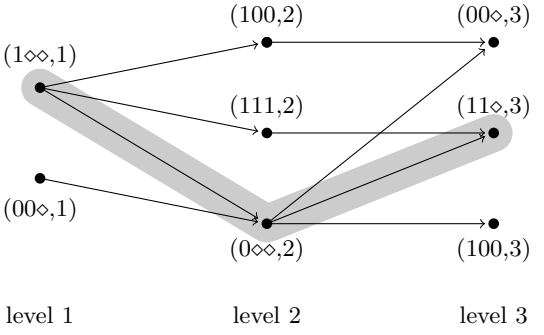


Fig. 7. The 3-pattern graph for the self-set S from Figure 2, which is a compressed version of the graph shown in Figure 6

Definition 7 (Pattern paths). *Let $S \in \Sigma^L$ be a self-set and G its r-pattern graph. A path from a vertex in level 1 to a vertex in level $L - r + 1$ is called a pattern path. It describes a string $d \in \Sigma^L$ if for every r-chunk pattern (π_i, i) on the path, the pattern π_i describes $d[i, \ldots, i + r - 1]$.*

The role of pattern paths for r-contiguous detectors is similar to the role of r-chunk patterns for r-chunk detectors: (1) A pattern path describes r-contiguous detectors. These detectors can be constructed by merging the r-chunk patterns of the path according to their position and then filling up the $\diamond$ entries arbitrarily, as shown in Figure 8. Since for every position $i \in \{1, \ldots, L - r + 1\}$ there exists an r-chunk pattern that matches the constructed string, it does not match any string in S and is, therefore, an r-contiguous detector. (2) Two pattern paths that differ in at least one vertex describe disjoint sets of r-contiguous detectors. (3) The pattern paths cover all r-contiguous detectors, as stated in the following lemma:

Lemma 2 (Pattern paths describe contiguous detectors). *Let $S \subseteq \Sigma^L$ be a self-set and G its r-pattern graph. The set of strings described by the pattern paths of G is exactly* $\textsc{ContD}(S, r)$.



Merge of prefix patterns from the path:

Prefix patterns from the path:	1⋄⋄
	0⋄⋄
	11⋄
Merged path patterns:	1011⋄
Described detectors:	10110
	10111

Fig. 8. A pattern path in the 3-pattern graph from Figure 7 that describes two 3-contiguous detectors

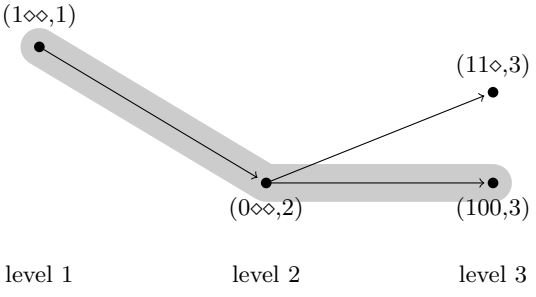
Proof. We have discussed in the previous paragraph that every pattern path corresponds to ab r-contiguous detector. It remains to show that there is a pattern path for each $d \in \textsc{ContD}(S, r)$. Let $d \in \textsc{ContD}(S, r)$ be an r-contiguous detector. For every position $i \in \{1, \ldots, L - r + 1\}$, the tuple $(d[i, \ldots, i + r - 1, i)$ is an r-chunk detector and, hence, described by some r-chunk pattern $(\pi_i, i) \in D_i$. We show that G contains the path $(\pi_1, 1), \ldots, (\pi_{L-r+1}, L - r + 1)$, which describes s by definition. Note, that it suffices to show that this holds after step 2 of the construction of the r-pattern graph in Definition 6. Consider the two patterns (π_i, i) and $(\pi_{i+1}, i + 1)$ with $\pi_i = v_i w_i$ where v_i is a string over $\{0, 1\}$ and w_i consists only of $\diamond$ (similarly $\pi_{i+1} = v_{i+1} w_{i+1}$ for appropriate v_{i+1} and w_{i+1}). If $|v_{i+1}| \geq |v_i| - 1$, then after step 2 there is an edge from (π_i, i) to $(\pi_{i+1}, i + 1)$. The case $|v_{i+1}| < |v_i| - 1$ cannot occur for the following reason: By the construction of the r-chunk patterns in the algorithm in Figure 5, v_i is not a substring of any string from S at position i, but its prefix of length $|v_i| - 1$ is a substring of a string $s \in S$ at position i. Therefore, v_{i+1} is a substring at position $i + 1$ in s, which contradicts the construction of the r-chunk patterns. □

Similar to the fact that pattern paths describe r-contiguous detectors, subpaths of them describe substrings of r-contiguous detectors. Thus, if we want to know whether a given string m matches an r-contiguous detector, it suffices to look

at subpaths that represent substrings of length r. We use the following graph to look at possible paths for a string m and a position i:

Definition 8 (Restricted pattern graph). *Let $S \subseteq \Sigma^L$ be a self-set, G its r-pattern graph, $m \in \{0,1\}^L$, and $i \in \{1,\dots,L-r+1\}$. The (m,i)-restricted r-pattern graph for S is the induced graph of G that contains exactly the vertices (π, j) with $j \in \{i,\dots,i+r-1\}$ where $\pi[1,\dots,r+i-j]$ describes $m[j,\dots,i+r-1]$.*

Given the r-pattern graph beforehand, a string m and an index i, the (m,i)-restricted graph can be constructed as in the definition, which needs time at most $O(|D|r)$. Figure 9 shows an example for restricted pattern graphs.



Path describes a substring of length 3 of a detector:

Merged path patterns:	10100
Monitor string m:	10101
Matching detector:	10110

Fig. 9. The $(10101, 1)$-restricted 3-pattern graph for the self-set from Figure 2 with a path from level 1 to 3

Lemma 3 (Contiguous detection with pattern graphs). *Let G' be the (m,i)-restricted r-pattern graph for a self-set $S \subseteq \{0,1\}^L$ and $m \in \{0,1\}^L$. The string m matches a detector $d \in \textsc{ContD}(S,r)$ at position i if, and only if, there is a path from the leftmost level (level i) to the rightmost level (level $\min\{i+r-1, L-r+1\}$) in G'.*

Proof. Let m match a detector $d \in \textsc{ContD}(S,r)$ at position i. By Lemma 2 we know that there exists a pattern path $(\pi_1,1),\dots,(\pi_{L-r+1},L-r+1)$ in the unrestricted r-pattern graph that describes d. Since $m[i,\dots,i+r-1] = d[i,\dots,i+r-1]$, we also know that for every $j \in \{i,\dots,i+r-1\}$, the prefix of length $r+i-j$ of π_j describes the substring of length $r+i-j$ at position j in m. Thus, there is a path from the leftmost to the rightmost level in G'.

For the other direction, consider a path from the leftmost to the rightmost level in the restricted graph. It is part of a pattern path in the unrestricted r-pattern graph. We choose an r-contiguous detector d from the set of detectors

Algorithm Efficient-Contiguous-Negative-Selection.
Input: A self-set $S \subseteq \Sigma^L$, an integer $r \in \{1, \ldots, L\}$ and a monitor set $M \subseteq \Sigma^L$.
Output: For each $m \in M$, either self or non-self for r-contiguous detection.

```
1   construct the set D of r-patterns
2   construct the r-pattern graph G from D
3   for each m ∈ M do
4       for i = 1 to L − r + 1 do
5           construct the (m, i)-restricted r-pattern graph G′
6           if there is a path from the leftmost to the rightmost level in G′ then
7               output m is "non-self"
8       if m was not detected as "non-self" then
9               output m is "self"
```

Fig. 10. Our efficient algorithm for negative selection with r-contiguous detectors

described by the pattern path, such that the free entries between position i and $i + r - 1$ (the positions with only $\diamond$ entries in all r-chunk patterns) are filled like in m. Then d matches m at position i since the construction of the restricted graph assures that, whenever the considered pattern path forces a 0 (or 1) entry in a position $j \in \{i, \ldots, i + r - 1\}$, m already has a 0 (or 1) entry at that position. □

Given an (m, i)-restricted r-pattern graph, the property of the last lemma can be tested as follows: Assign to each vertex of level i the weight 1. Then iterate through all of the remaining levels and set the weight of every vertex to the maximal weight of a predecessor vertex plus one. There is a path from the leftmost to the rightmost level if the weight of a vertex in the rightmost level equal the number of levels in the restricted graph. This procedure takes time at most $O(|D|^2 r)$.

Bringing it all together, the algorithm in Figure 10 correctly detects for each element of a monitor set M, whether it is self or non-self. The algorithm uses time at most $O(|S|Lr^2)$ for the construction of the set D in line 1 and $O(|S|Lr)$ is an upper bound for the number of patterns in D. In line 2 the running time for the construction of the r-pattern graph is at most $O(|D|^2 r + |D|^3)$. For the detection of a single element m from the monitor set, the algorithm iterates over all possible positions $i \in \{1, \ldots, L - r + 1\}$. For each position, it constructs the (m, i)-restricted pattern graph and decides whether there is a path from the leftmost to the rightmost level. Thus, the time for the detection of a single element is at most $O(|D|^2 Lr)$.

5 Discussion

We have seen that negative selection based on r-chunk and r-contiguous detector is inefficient if *all* detectors are generated. However, the complete detector sets are highly redundant. Our results show that equivalent, but significantly smaller

representations of these sets can be constructed. It is therefore unnecessary to enumerate all detectors explicitly. In the case of r-chunk detectors, our construction is a straightforward continuation of Esponda's results [7] and should not be much harder to implement than the original procedure. On the other hand, the construction for r-contiguous detectors is still a little more involved. We would like to point out here that our main focus here was to show that r-contiguous detection is *principally* feasible in polynomial time. For concrete implementations, the algorithm developed in the previous section can still be streamlined.

It is important to put our work in context with the work carried out by Stibor [3,6,9], who demonstrated that the problem of deciding if any r-contiguous detector is generable for a given self-set can be expressed as an instance of the k-CNF satisfiability problem (SAT), which is $\mathcal{NP}$-complete. This result is not a contradiction to our findings. Formally, Stibor showed that the generability problem for r-contiguous detectors – let us denote it GenP – is polynomial-time reducible to SAT, and this implies GenP $\in \mathcal{NP}$. However, it does not imply that GenP is $\mathcal{NP}$-complete – to prove that, one needs additionally a reduction in the opposite direction. In fact, our results imply GenP $\in \mathcal{P}$: Given S and r, construct the r-pattern graph G. Detectors are generable if, and only if, there is a pattern path in the graph. This test takes polynomial time. Hence, it is very unlikely that detector generability is complexity-wise equivalent to boolean satisfiability, since this would now imply $\mathcal{P} = \mathcal{NP}$.

In summary, our work shows that string-based negative selection is a computationally feasible approach. It remains to be seen if this finding leads to a substantial improvement to the real-world performance of negative selection, or even the desired "killer application" for AIS. However, since the negative selection algorithm has been a source of inspiration for many in the past, maybe there is now a little more hope.

References

1. Forrest, S., Perelson, A.S., Allen, L., Cherukuri, R.: Self-nonself discrimination in a computer. In: Proceedings of the IEEE Symposium on Research in Security and Privacy, pp. 202–212 (1994)
2. Forrest, S., Hofmeyr, S.A., Somayaji, A.: Computer immunology. Communications of the ACM 40, 88–96 (1997)
3. Stibor, T.: Foundations of r-contiguous matching in negative selection for anomaly detection. Natural Computing (2008)
4. Stibor, T.: On the Appropriateness of Negative Selection for Anomaly Detection and Network Intrusion Detection. Ph.D thesis, Darmstadt University of Technology (2006)
5. D'Haeseleer, P.: An immunological approach to change detection: theoretical results. In: Proceedings of the 9th IEEE Computer Security Foundations Workshop, pp. 18–26 (1996)
6. Timmis, J., Hone, A., Stibor, T., Clark, E.: Theoretical advances in artificial immune systems. Theoretical Computer Science 403, 11–32 (2008)

7. Esponda, C.: Negative Representations of Information. Ph.D thesis, University of New Mexico (2005)
8. Ukkonen, E.: On-line construction of suffix-trees. Algorithmica 14(3), 249–260 (1995)
9. Stibor, T.: Phase transition and the computational complexity of generating r-contiguous detectors. In: de Castro, L.N., Von Zuben, F.J., Knidel, H. (eds.) ICARIS 2007. LNCS, vol. 4628, pp. 142–155. Springer, Heidelberg (2007)

T Cell Receptor Signalling Inspired Kernel Density Estimation and Anomaly Detection

Nick D.L. Owens[1], Andy Greensted[1], Jon Timmis[1,2], and Andy Tyrrell[1]

[1] Department of Electronics, University of York, York, UK
[2] Department of Computer Science, University of York, York, UK
{ndlo100,ajg112,jt517,amt}@ohm.york.ac.uk

Abstract. The T cell is able to perform fine-grained anomaly detection via its T Cell Receptor and intracellular signalling networks. We abstract from models of T Cell signalling to develop a new Artificial Immune System concepts involving the internal components of the TCR. We show that the concepts of receptor signalling have a natural interpretation as Parzen Window Kernel Density Estimation applied to anomaly detection. We then demonstrate how the dynamic nature of the receptors allows anomaly detection when probability distributions vary in time.

1 Introduction

Recently the Artificial Immune Systems (AIS) community has called for a greater consideration of biology when designing new AIS [8], [9]. This greater consideration is intended to come in the form of rigorous modelling of the immune system with the objective of capturing its more complex and appealing properties. Simultaneously there have been calls for more theoretical results relating to AIS, [8] and [10]. In this paper we present new AIS concepts that generalises T Cell signalling and fulfils both criteria of modelling and theoretical results. The methods have been constructed from modelling of signalling processes related to the T Cell Receptor [3], [4]. We discover that an interpretation of receptor signalling has a direct mapping to statistical anomaly detection [11] evaluated through kernel density estimation [6]. The paper is organised as follows: section 2 gives biological background; section 3 presents features of the biology important to AIS; section 4 analyses the one receptor case; section 5 reviews the basic concepts of kernel density estimation and Bayesian anomaly classification; section 6 presents a mapping of our algorithm onto the statistical techniques; section 7 incorporates the dynamic behaviour of the single receptor with the mapping given in section 6; finally section 8 presents some conclusions.

2 Biological Background

The T lymphocyte (T cell) must perform fine grain discrimination of peptide bound Major Histocompatibility Complex (pMHC) molecules on antigen-presenting cells (APCs) through its T cell Receptor (TCR) [1]. The T cell must

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 122–135, 2009.

discriminate between abundant self-pMHC, $99.9 - 99.99\%$ of all pMHC on an APC, and non-self-pMHC which comprises the other $0.01 - 0.1\%$ of the total pMHC expressed [1]. The bind between a TCR and pMHC is approximately 3-7 orders of magnitude weaker than the bind between antibody and antigen [1]. The lock-and-key paradigm common to certain immunological theories and present in the shape-space representation of many AIS is not necessarily applicable here. Explanations for the T Cell's discrimination abilities can be found in the complex and *tunable* information processing pathways emanating from the TCR [1]. In this paper, we have taken T cell signalling models given in [2], [5], [3] and [4] to develop our algorithm. We provide a brief overview of the concepts in the model [4]. The complexities of the TCR-pMHC bind are well abstracted by considering the average lifetime t of the bind and so the TCR and pMHC can be thought to dissociate at rate $1/t$. Given an input of the bind lifetime and the state of the surrounding intracellular molecules the TCR may induce an activation signal in the T cell. This is achieved via the following processes:[1]

- *Kinetic Proofreading.* This entails energy consuming steps that must be overcome before the TCR can generate an activation signal. These steps are reversed upon TCR pMHC dissociation. The result is step-like behaviour, TCR-pMHC binds that dissociate before kinetic proofreading steps complete never generate an activation signal, TCR-pMHC binds that exist long enough for kinetic proofreading completion generate identical activation signal.
- *Negative Feedback.* The progression of kinetic proofreading steps generates a signal that inhibits and reverses the kinetic proofreading steps. This signal is not instantaneous, it takes time to build, but it has the potential arrest any kinetic proofreading process regardless of TCR-pMHC bind strength.
- *Negative Feedback Destruction.* If a TCR completes kinetic proofreading it will generate an activation signal. This signal undergoes amplification and is able to protect all TCRs from the negative feedback.
- *Tuning.* The co-receptor (CD8 in the cytotoxic case considered in [2], [4]) density can be understood as a tuning parameter. It has been shown [4] that small increases in co-receptor density increase the probability of activation, however large increases in co-receptor density will desensitise a cell and decrease the probability of activation.

A key concept is signal spreading: first, if a TCR-pMHC dissociates the pMHC may reassociate to a nearby TCR; second, the negative signal generated by a single TCR will spread and dampen surrounding TCRs; third, the negative feedback destruction signal will spread and protect surrounding TCRs from the negative feedback. The models in [2], [3], [4] had well-mixed (thermodynamic equilibrium) assumptions, in this paper we remove the this assumption and use space to dictate the coupling between receptors.

[1] With the exception of certain co-receptor behaviour, all these processes occur internally, within the T cell.

3 The Generalised Receptor

The modelling work in [3] and [4] has demonstrated the qualitatively important features of a receptor. We move away from biological descriptions and present the features of the receptor within a computational context. The receptor has more features of interest (at least they have a complex role within the biological system) than are used in this paper. We refrain from specifying all semantics of the definitions of a receptor. We feel it prudent to inform the community of all features, even if we do not make use of the entire set. In subsequent sections we will make concrete instantiations of subsets of receptor features and at which point we will be clear on semantics.

Definition 1. *A receptor r is a tuple $(\ell, \mathcal{B}, \beta, p, \mathcal{C})$, with:*

- $\ell \in (0, \infty)$*, the length of the receptor.*
- $\mathcal{B} = \{b_0, b_1, \ldots, b_{b-1}\}$*, the ordered set of b barriers, with $b_i \in (0, \ell)$ and $b_i < b_j \Leftrightarrow i < j$*
- $\beta \in \mathcal{B}$*, the base negative feedback barrier.*
- $p \in [0, \ell]$*, the receptor position.*
- $\mathcal{C}$*, the set out receptor outputs. Generally $\mathcal{C} = \{0, 1\}$ is a binary output denoting whether $p = \ell$.*

Definition 2. *The environment $\mathcal{E}$ is a tuple $(\mathcal{U}, \mathcal{R}, \mathcal{L}, \mathbf{n}, \mathbf{q}, \mathbf{d})$, with:*

- $\mathcal{U} = \mathbb{R}^m$*, set of m-dimensional real inputs.*
- $\mathcal{R} = \{r_0, r_1, \ldots, r_{s-1}\}$ *the set of s receptors.*
- $\mathcal{L} : \mathcal{R} \rightarrow \mathbb{R}^m$*, The location function that distributes the receptors within input space.*
- $\mathbf{n}$ *the negative feedback signal, specified in bold type to signify that space can be included in the negative feedback signal. Generally we assume the negative feedback at a particular location to be $n \in \mathbb{R}^+$.*
- $\mathbf{q}$ *the negative feedback protection signal, specified in bold type to signify that space can be included in the protection definition. Generally we assume that the protection at a particular location to be binary $q \in \{0, 1\}$.*
- $\mathbf{d}$ *the tuning parameter.*

A diagrammatic representation of the receptor is given in figure 1. The environment $\mathcal{E}$ has s receptors, $\mathcal{L}$ provides the distribution of receptors. In this paper we will uniformly distribute receptors in input space $\mathcal{U}$. If one so chooses it would be possible to position the receptors within a shape space formalism. The following provides a high level description of a generalised receptor:

1. A receptor r receives input $u \in \mathcal{U}$. The receptor's position p is advanced from an initial 0 toward ℓ by a function of u and the tuning parameter $\mathbf{d}$. A greater value of $\mathbf{d}$ will result in greater progression of p.
2. After r receives u, p may be lowered to the greatest barrier smaller than the current value of p. Appropriate stimulations of the receptor will increment p through the barriers $b_1, \ldots, b_{b-1}$.

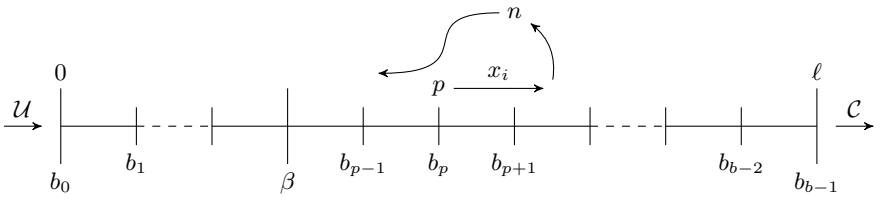


Fig. 1. The Receptor. The receptor receives an input u which allows the receptor position p to progress toward ℓ. The receptor will generate negative feedback if $p > \beta$. Should $p = \ell$ then the receptor will successfully signal with $\mathcal{C} = 1$.

3. If $p \geq \beta$ the receptor will linearly generate negative feedback. The positive tuning density **d** is able to increase the generation rate of negative feedback. The negative feedback n at a receptor will reverse the progression of p, pushing the receptor position toward zero.
4. The receptor position p and the negative feedback n will decay in time.
5. If $p = \ell$, a classification signal $\mathcal{C} = 1$ is generated. A protection signal **q** may also be generated which inhibits the negative feedback **n**.

Summarising, a receptor receives an input which advances the receptor's position toward the receptor end. If the position is greater than the base negative feedback barrier the receptor will generate negative feedback. This negative feedback will actively reverse the progression of the receptor position p.

4 The Single Receptor

We analyse a possible choice for the behaviour of a single receptor. This should aid understanding of the concepts given in the previous section and is of importance in the dynamic case in section 7. We describe the evolution of the receptor p_t and the negative feedback n_t at time t as follows:

$$\begin{aligned} p_{t+1} &= p_t + \Delta(u_t - an_t - bp_t), \quad p_t \geq 0, \\ n_{t+1} &= n_t + \Delta(gH(p_t - \beta) - dn_t), \quad n_t \geq 0, \end{aligned} \qquad H(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases} \tag{1}$$

u_t is the input at time t; a is the negative feedback efficacy; b is the receptor position decay rate; g is the negative feedback generation rate; β the base negative feedback barrier; d the negative feedback decay rate; Δ controls the granularity of the discrete steps; $H(x)$ is the Heaviside step function. We assume that $u_t, a, b, d, g, \beta > 0$. We calculate some relevant properties of these recurrences, we look for conditions on u_t for $p_t \geq \ell$. To alleviate difficulties presented by $H(p_t - \beta)$ we condition on being above the step, $p_t \geq \beta \ \forall t$. For convenience we write $\phi_x \equiv (1 - \Delta x)$. Setting initial position $p_0 = \beta$ and initial negative feedback n_0, results in the following solutions:

$$n_t = (n_0 - g/d)\phi_d^t + g/d \tag{2}$$

$$p_t = \beta\phi_b^t + \Delta\sum_{k=0}^{t-1} u_{n-k}\phi_b^k + \eta_t \tag{3}$$

$$\eta_t = -a\left[\frac{(n_0 - g/d)\phi_d\left(\phi_d^t - \phi_b^t\right)}{b-d} + \frac{g(1-\phi_b^t)}{db}\right] \tag{4}$$

η_t describes the influence of the negative feedback on p_t. The solutions have the requirement $\Delta b, \Delta d < 1$ so that $\phi_b, \phi_d < 1$. With these conditions the maximum negative feedback is g/d. If we assume constant $u_t = u$ then the solution for p_t becomes:

$$p_t = (\beta - u/b)\phi_b^t + u/b + \eta_t \tag{5}$$

With no negative feedback the maximum value for p_t is u/b. A further case of interest is a periodic $u_{r,t}$:

$$u_{r,t} = \begin{cases} u & \text{if } t \bmod r = 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

This will result in $u_{r,t} = u$ once every r time steps. The solution for p_t is now:

$$p_t = \beta\phi_b^t + \Delta u\phi_b^{t_{[r]}}\left[\frac{\phi_b^t - 1}{\phi_b^r - 1}\right] + \eta_t \tag{7}$$

$t_{[r]} \equiv t \bmod r$. We return to the constant $u_t = u$ case, and note that as $t \to \infty$ one expects similar mean behaviour for constant $u_t = u/r$ and periodic $u_{r,t}$. We derive conditions to ensure the assumption $p_t \geq \beta$ and answer the related question, what are the constraints on u such that $p_t < \ell$:

$$b\left[\beta - \frac{\eta_t}{1-\phi_b^t}\right] \leq u < \frac{b\left(\ell - \beta\phi_b^t - \eta_t\right)}{1-\phi_b^t} \tag{8}$$

$$\text{As } t \to \infty: \qquad b\beta + \tfrac{ag}{d} \leq u < b\ell + \tfrac{ag}{d} \tag{9}$$

The lefthand inequality states the condition on u such that the assumption $p_t \geq \beta$ holds. The righthand inequality states the condition on u for the negative feedback and receptor position decay to contain $p_t < \ell$. Relaxing the assumption that $p_t \geq \beta$ and starting from $p_0 = 0$ and $n_0 = 0$ and for constant input u, if $b\beta \leq u < ag/d$ then the negative feedback will rise to $p_t = \beta$ at equilibrium. The level of negative feedback at equilibrium will be:

$$n_t = (u - b\beta)/a \tag{10}$$

We now calculate one-step breaking conditions such that $p_t < \ell$ and $p_{t+1} \geq \ell$ with $u_{t-1} = u$, $u_t = u^\star$ and $u < u^\star$. That is, we assume p is at equilibrium with input u and then at time t the input is increased to $u^\star$, and $p_{t+1} \geq \ell$ if:

$$\begin{cases} u^\star - u \geq (\ell - \beta)\Delta^{-1} & \text{if } p_t = \beta \\ (u^\star - u)\Delta b + u \geq \Delta b\ell + agd^{-1} & \text{if } \beta < p_t < \ell \end{cases} \tag{11}$$

5 Kernel Density Estimation and Statistical Anomaly Detection

We provide an overview of pertinent concepts of kernel density estimation and statistical anomaly detection. For a superior presentation of these ideas we direct the reader to [6], [11], [12]. We have an interest in autonomous systems that are equipped with a set of sensors. We would like to address the following:

1. Given a sequence of sensor readings, classify a subsequent new sensor reading as normal or anomalous. We name this anomaly *type 1*.
2. The classification of anomaly *type 1* will be with respect to a hypothesis on the environment. We desire some plasticity in the representation of the environment, such that the classification of a point in input space may change in time[2]. Our anomaly detection system should be robust to certain changes in the environment, but will detect an anomaly *type 2* if the environment changes and it is not sufficiently recognisable from previous states.

We formalise these two problems in terms of statistical anomaly detection on probability distributions over sensor values. The second problem requires changes in distribution over time, or the inclusion of time as an extra dimension in a multi-variate distribution. We hold on these concerns until section 5.2 and address the simpler first problem.

Given a collection of n $d-$dimensional observations $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n} \in \mathbf{x}$ which we assume to be i.i.d. and $\mathbf{x} = \mathbb{R}^d$, we discuss anomaly detection in terms of estimation of probability density function $p(\mathbf{x})$. One method of estimation which is attractive due to its non-parametric nature is kernel density estimation (also known as Parzen Window Estimation) [6]. This provides the following estimation $\hat{p}(\mathbf{x})$:

$$\hat{p}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x_i}}{h}\right), \qquad K(\mathbf{x}) \geq 0, \qquad \int_{-\infty}^{\infty} K(\mathbf{x}) d\mathbf{x} = 1 \tag{12}$$

$K(\cdot)$ is a kernel function with width h. With the properties that $K(\cdot)$ is always positive and integrates to one, then $\hat{p}(\mathbf{x})$ is also a probability density function. Further, if $h = h(n)$ is a function of the number of samples n then $\hat{p}(\mathbf{x})$ will converge to $p(\mathbf{x})$ if $\lim_{n\to\infty} h(n) = 0$ and $\lim_{n\to\infty} nh^d(n) = \infty$.

Common choices for the kernel $K(\cdot)$ are the standard multivariate normal density function:

$$K_n(\mathbf{x}) = (2\pi)^{-d/2} \exp\left(-\mathbf{x^T x}/2\right) \tag{13}$$

and the bounded multivariate Epanechnikov kernel, which may be computationally simpler to evaluate:

$$K_e(\mathbf{x}) = \begin{cases} (2c_d)^{-1}(d+2)(1 - \mathbf{x^T x}) & \text{if } \mathbf{x^T x} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

[2] An example is the expected ambient light level between night and day.

$c_d = \pi^{\frac{n}{2}}/(\Gamma(\frac{n}{2}+1))$ is the volume of the unit $d-$dimensional sphere. Often, more important than choice of kernel function is choice of kernel width h [6]. This choice has a tradeoff between the bias and variance of $\hat{p}(\mathbf{x})$ (between a systematic error and random error), a small value of h may eliminate the bias but it will result in a large variance if the sample size is not large enough. A common method of choosing h is leave-one-out likelihood cross-validation [6]. This involves choosing h to maximise a score $C_v(h)$, which should result in a density estimate $\hat{p}(x)$ that is close to the true density $p(x)$ in terms of the Kullback-Leibler information divergence $I(p,\hat{p})$. $C_v(h)$ and $I(p,\hat{p})$ are defined:

$$C_v(h) = \sum_{i=1}^{n} \log \left[\frac{1}{(n-1)h^d} \sum_{j \neq i}^{n} K\left(\frac{\mathbf{x_i} - \mathbf{x_j}}{h} \right) \right], \quad I(p,\hat{p}) = \int p(\mathbf{x}) \log \left[\frac{p(\mathbf{x})}{\hat{p}(\mathbf{x})} \right] d\mathbf{x} \tag{15}$$

5.1 Anomaly Classification Using Density Estimation

Using methods given in [11] and [7] the problem of *type 1* anomaly classification can be formulated as follows: given training data $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n} \in \mathbf{x}$ we must classify a new data point $\mathbf{v}$ to be in class $\mathcal{C}_1$ if the $\mathbf{v}$ is thought to come from the same distribution as the $\mathbf{x_i}$ or to be in class $\mathcal{C}_2$ if $\mathbf{v}$ is thought to be anomalous. By assumption all the training data are classified class $\mathcal{C}_1$. The new data point $\mathbf{v}$ may belong to either class with prior probabilities $P(\mathcal{C}_1)$ and $P(\mathcal{C}_2)$ and $P(\mathcal{C}_1) + P(\mathcal{C}_2) = 1$. To minimise the probability of misclassification $\mathbf{v}$ is assigned to the class with the largest posterior probability, so we assign $\mathbf{v}$ to $\mathcal{C}_1$ if $P(\mathcal{C}_1|\mathbf{v}) > P(\mathcal{C}_2|\mathbf{v})$. Bayes theorem states:

$$P(\mathcal{C}_i|\mathbf{v}) = \frac{p(\mathbf{v}|\mathcal{C}_i)P(\mathcal{C}_i)}{p(\mathbf{v})} \tag{16}$$

We assign $\mathbf{v}$ to class $\mathcal{C}_1$ when:

$$p(\mathbf{v}|\mathcal{C}_1)P(\mathcal{C}_1) > p(\mathbf{v}|\mathcal{C}_2)P(\mathcal{C}_2) \tag{17}$$

The quantities involving class $\mathcal{C}_1$ may be modelled using kernel density estimation on the training data. As nothing is known about the distribution of anomalous data the simplest assumption is to assume a uniform distribution across a large region of input space. With this assumption equation 17 is equivalent to applying a threshold on the estimated probability density of the training data, as shown in figure 2. If the distribution for the anomalous data is assumed to be uniform with probability α, then the classification decision for a new data point $\mathbf{v}$ is as follows:

$$\text{Classification}(\mathbf{v}) = \begin{cases} \text{Normal} & \text{if } \alpha \leq \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{v}-\mathbf{x_i}}{h}\right) \\ \text{Anomaly} & \text{otherwise} \end{cases} \tag{18}$$

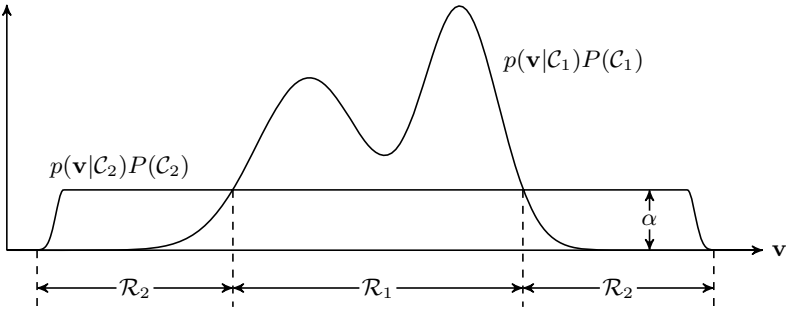


Fig. 2. Bayesian formalism for determining whether a new datapoint is anomalous, adapted from [7]. The input space for $\mathbf{v}$ is divided into regions $\mathcal{R}_1$ and $\mathcal{R}_2$ such that $\mathbf{v}$ is classified as an anomaly if it falls in region $\mathcal{R}_2$. The threshold α defines the anomalous distribution.

5.2 Probability Distributions Varying in Time

We turn to the second problem *type 2* anomaly detection. Assume that the sample $\mathbf{x_1}, \mathbf{x_2}, \ldots \in \mathbf{x}$ occurs in time, that is sample point $\mathbf{x_i}$ occurs at time t_i. Further, assume that no two samples occur at identical times and $t_i < t_j \Leftrightarrow i < j$, and for simplicity the interval between sample points, $t_{i+1} - t_i$, is constant. Define a sliding the window of length w to be $\omega_i = \{\mathbf{x_i}\} \cup \omega_{i-1} \setminus \{\mathbf{x_{i-w}}\}$. We are interested in comparing the estimates $\hat{p}(\mathbf{x}|\omega_i)$ with $p(\mathbf{x}|\omega_j)$, we may do this via the Kullback-Leibler divergence given in equation 15, that is $I(\hat{p}(\mathbf{x}|\omega_i), \hat{p}(\mathbf{x}|\omega_j))$. Then we define a threshold on the rate of change of information divergence, γ:

$$\frac{I(\hat{p}(\mathbf{x}|\omega_i), \hat{p}(\mathbf{x}|\omega_j))}{|i-j|} < \gamma \qquad i \neq j \tag{19}$$

6 Mapping the Generalised Receptor onto Kernel Density Estimation

Appropriate choice of features of the generalised receptor allow us to exactly reconstruct the kernel density estimation Bayesian anomaly detection technique given in the previous section. The receptor position and negative feedback are now defined at every point in $\mathbb{R}^d$, we label the receptor position at $\mathbf{x} \in \mathbb{R}^d$ as $r_p(\mathbf{x})$ and the negative feedback at the same point as $r_n(\mathbf{x})$. Using the concept of the signal spreading discussed in section 2, we model the spreading of stimulation to receptors via a kernel. Then the stimulation from a sample point $\mathbf{x_s}$ is given by a stimulation kernel K_S:

$$S(\mathbf{x}, \mathbf{x_s}) = \frac{1}{nh^d} K_S \left(\frac{\mathbf{x} - \mathbf{x_s}}{h} \right) \tag{20}$$

We could also use a kernel K_N to describe the spread of negative feedback, however for the current purposes this is unnecessary[3]. We replicating the static classification problem and so do not include any of the dynamic behaviour of the single receptor case (section 4). We allow a training phase, where the receptor positions $r_p(\mathbf{x})$ and the negative feedback $r_n(\mathbf{x})$ are established, and then a test phase in which we test new data points to see if $r_p(\mathbf{x}) \geq \ell$, i.e. we have receptor activation. We will show that this condition is equivalent to the classification condition in equation 18.

Training Phase. Set the base negative feedback barrier β as the classification threshold α for the anomalous distribution depicted in figure 2. The total receptor stimulation $S(\mathbf{x})$ and negative feedback $r_n(\mathbf{x})$ are:

$$S(\mathbf{x}) = \sum_{i=1}^{n} S(\mathbf{x}, \mathbf{x_i}) \qquad r_n(\mathbf{x}) = \begin{cases} S(\mathbf{x}) - \beta & \text{if } S(\mathbf{x}) \geq \beta \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

This agrees with the properties of the single receptor outlined in section 4: equation 10 states the negative feedback is proportional to the difference between the input (stimulation) and the negative feedback barrier.

Test Phase. Set the receptor position $r_p(\mathbf{x}) = 0$ everywhere and the receptor length ℓ equal to the maximum height of the stimulation kernel K_S. In the case of the standard 1-dimensional normal kernel $\ell = (\sqrt{2\pi})^{-1}$. Then for a test point $\mathbf{v}$ set the new receptor positions $r_p(\mathbf{x})$:

$$r_p(\mathbf{x}) = K_S((\mathbf{x} - \mathbf{v})/h) - r_n(\mathbf{x}) \tag{22}$$

Then classify $\mathbf{v}$:

$$\text{Classification}(\mathbf{v}) = \begin{cases} \text{Normal} & \text{if anywhere } r_p(\mathbf{x}) < \ell \\ \text{Anomaly} & \text{if anywhere } r_p(\mathbf{x}) \geq \ell \end{cases} \tag{23}$$

This classification is equivalent to the classification given in section 5, equation 18.

Proof. Since $r_n(\mathbf{x}) \geq 0$ and as $\ell = \max\{K_S\}$ then $r_p(\mathbf{x}) \leq \ell$. Then $r_p(\mathbf{x}) = \ell$ if $r_n(\mathbf{x}) = 0$. The condition for $r_n(\mathbf{x}) = 0$ is $S(\mathbf{x}) < \beta$ and as $\beta = \alpha$ the condition is $\alpha > S(\mathbf{x}) = \sum_{i=1}^{n} S(\mathbf{x}, \mathbf{x_i}) = \sum_{i=1}^{n} (nh^d)^{-1} K_S((\mathbf{x} - \mathbf{x_i})/h)$. Which is the condition of equation 18. □

This result is not surprising, in essence we have implemented the classical anomaly detection technique within the framework of the generalised receptor. It is the existence of the base negative feedback barrier β that allows the successful mapping.

[3] If a K_N were used the total negative feedback would be the stimulation convolved with the K_N. If K_S and K_N are normal, then $K_S * K_N$ is also normal.

7 Dynamic Anomaly Detection

We combine the static concepts introduced in the previous section, with the dynamic behaviour of the one receptor discussed in section 4. We consider a stream of points $\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3} \ldots \in \mathbf{x}$ at times $t_1, t_2, t_3 \ldots \in \mathbb{R}^+$ as discussed in section 5.2. We continue to specify a receptor position $r_p(\mathbf{x})$ and negative feedback $r_n(\mathbf{x})$ at all points in $\mathbb{R}^d$, but now their behaviour is defined by the recurrences of equation 1. We focus on anomaly *type 2* detection, that is we detect changes in an underlying distribution given a time window ω_t. As a proof of concept, we demonstrate that kernel density estimation can be performed well using receptors with just an appropriately chosen decay rate. We examine the following definition for receptor position at location $\mathbf{x}$ at time t with an input $\mathbf{x_t}$:

$$r_{pd}(\mathbf{x}, t+1) = r_{pd}(\mathbf{x}, t)\phi_b + \Delta S(\mathbf{x}, \mathbf{x_t}) \tag{24}$$

We are interested in anomalies with respect to a time window length w, so the influence of an input size u should decay to be less than ϵ within w time steps, therefore:

$$b \geq \Delta^{-1}(1 - (\epsilon/u)^{\frac{1}{w}}) \tag{25}$$

We compare the density estimation of $r_{pd}(\mathbf{x}, t)$ with a standard kernel density estimation $\hat{p}_k(\mathbf{x}|\omega_i)$. For simplicity we choose the 1-dimensional standard normal distribution as the underlying distribution $p(x)$. We compare the closeness between $p(x)$ and $p_k(x|\omega_t)$ and $r_{pd}(x, t)$ via the Kullback-Leibler divergence (equation 15). The kernel is set as the standard normal distribution; $w = 500$; cross-validation shows $h = 1.9$ to give robust performance over many samples of size 500. The results for two values of ϵ are given in figure 3, they show that $r_{pd}(x, t)$ has comparable performance $\hat{p}_k(x|\omega_t)$, and further $r_{pd}(x, t)$ seems to be less susceptible to noise.[4] We now include negative feedback and examine the ability of receptors to detect a *type 2* anomaly. The receptor positions and negative feedback are now defined:

$$\begin{aligned} r_p(\mathbf{x}, t+1) &= r_p(\mathbf{x}, t)\phi_b + \Delta(S(\mathbf{x}, \mathbf{x_t}) - a r_n(\mathbf{x}, t)) \\ r_n(\mathbf{x}, t+1) &= r_n(\mathbf{x}, t)\phi_d + \Delta g H(r_p(\mathbf{x}, t) - \beta) \end{aligned} \tag{26}$$

A *type 2* anomaly is detected if $r_p(\mathbf{x}, t) \geq \ell$. We perform numerical tests with two simple test distributions, $\rho_1(x, t)$ and $\rho_2(x, t)$, that vary in time:

$$\rho_1(x, t) = N(x - \theta_t) \tag{27}$$

$$\rho_2(x, t) = \tfrac{1}{2}(1 - \sigma_t)(N(x + 3) + N(x - 3)) + \sigma_t N(x) \tag{28}$$

$N(x)$ is the standard normal distribution, θ_t and σ_t are varied to produce changes in the underlying distributions:

[4] It should be noted that $\epsilon = 1.55 \times 10^{-3}$ is large enough that a sample greater than 500 is contributing to the $r_{pd}(x, t)$. A fair comparison with standard kernel density should take this into account. Since our intent is a proof of concept we do not calculate this discrepancy.

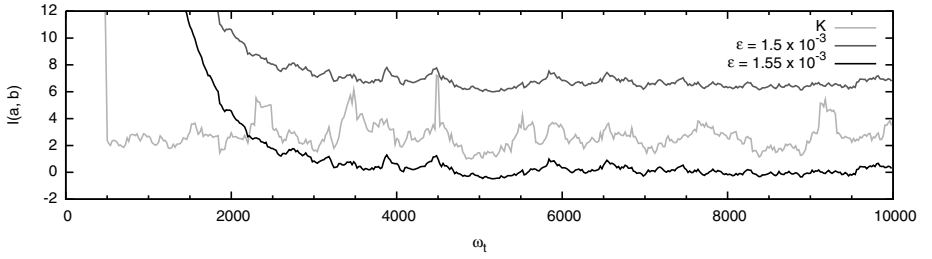


Fig. 3. The Kullback-Leibler divergence between $p(x)$ and $p_k(x|\omega_t)$ (K), and two estimates using receptor positions $r_{pd}(x,t)$ for $\epsilon = 1.5 \times 10^{-3}$ and $\epsilon = 1.55 \times 10^{-3}$. The close I is to zero the better the estimate.

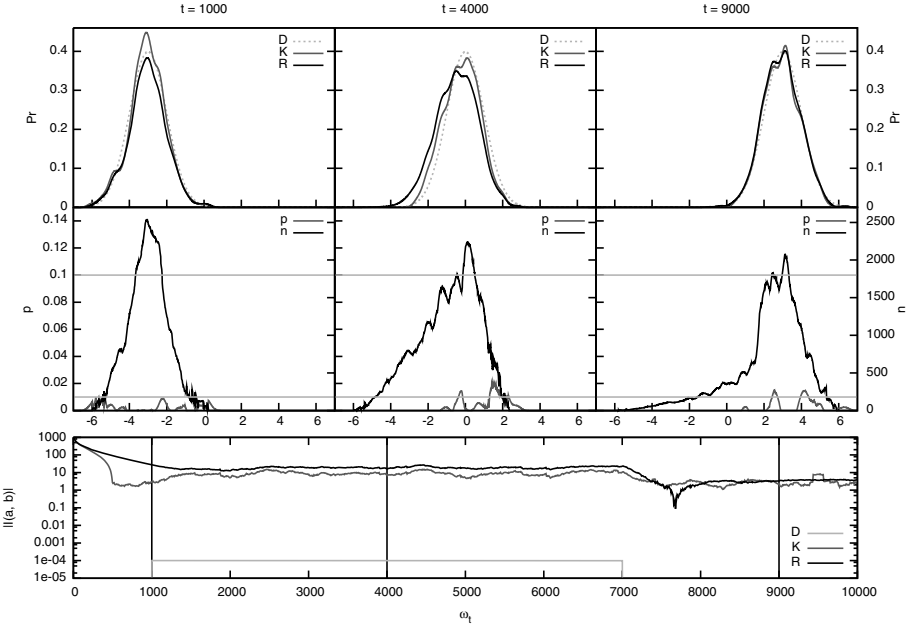


Fig. 4. Results for $\rho_1(x,t)$ and slow rate $\nu_s = 0.001$. Three times are shown: before the distribution change ($t = 1000$); during the distribution change ($t = 4000$); after the distribution change ($t = 9000$). The top row gives $\rho_1(x,t)$ (D); $\hat{p}_k(x|\omega_t)$ (K); $r_{pd}(x,t)$ (R). The middle row gives $r_p(x,t)$ and $r_n(x,t)$, $\beta = 0.01$ and $\ell = 0.1$ lines are marked. Note that $r_p(x,t) < \ell$. The bottom row plots the magnitude of the Kullback-Leibler divergence, D: $|I(\rho_1(x,t), \rho_1(x,t-1))|$, which is the change in underlying distribution in one time step; K: $|I(\rho_1(x,t), \hat{p}_k(x|\omega_i))|$; R: $|I(\rho_1(x,t), r_{pd}(x,t))|$. $t = 1000, 4000, 9000$ lines are marked.

$$\theta_t = \begin{cases} -3 & 0 \le t < \tau \\ -3 + \nu(t-\tau) & \tau \le t \le \frac{(6+\nu\tau)}{\nu} \\ 3 & \frac{(6+\nu\tau)}{\nu} < t \end{cases} \quad \sigma_t = \begin{cases} 0 & 0 \le t < \tau \\ \nu(t-\tau) & \tau \le t \le \frac{(1+\nu\tau)}{\nu} \\ 1 & \frac{(1+\nu\tau)}{\nu} < t \end{cases} \quad (29)$$

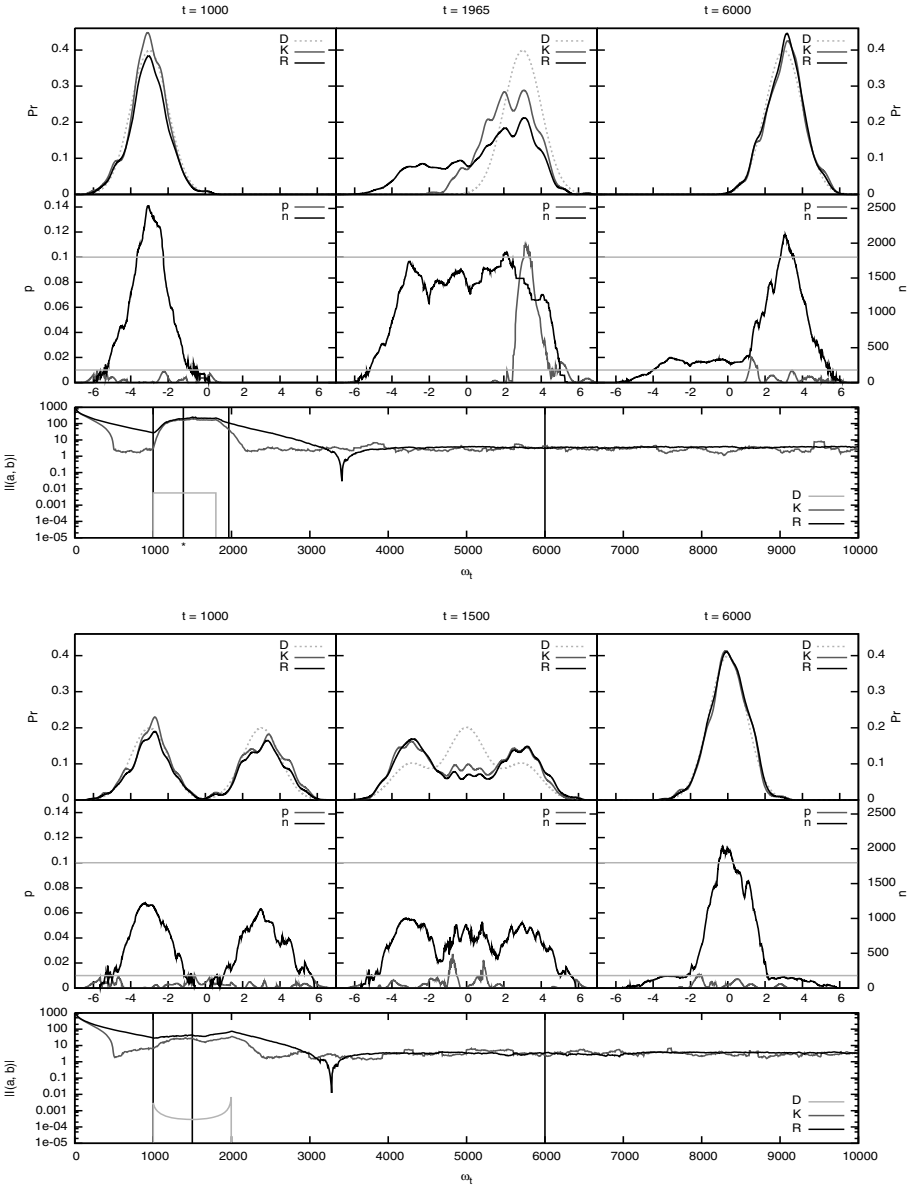


Fig. 5. *Top:* As figure 4, except for $\rho_1(x,t)$ and slow rate $\nu_f = 0.0075$, and for time points $t = 1000, 1965, 6000$. Note that $t = 1965$ plots show detection of a *type 2* anomaly, with $r_p(x,t) \geq \ell$, this anomaly was first detected at $t = 1385$ which is marked with a line and a "$*$". *Bottom:* As figure 4, except for $\rho_2(x,t)$ and slow rate $\nu_s = 0.001$ and for $t = 1000, 1500, 6000$. Here $r_p(x,t) < \ell$ and so the distribution changes slowly enough that no anomaly is detected.

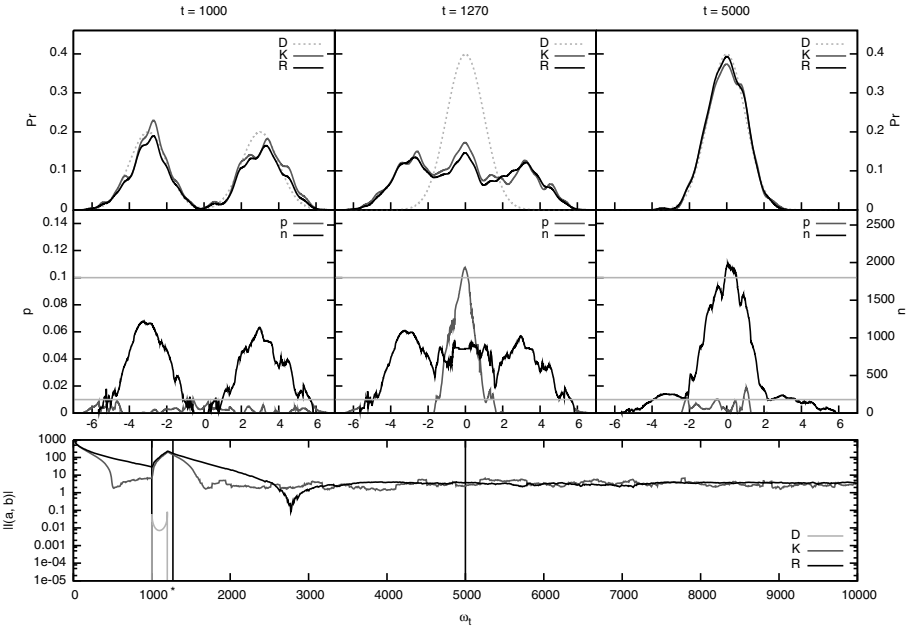


Fig. 6. As figure 4, except for $\rho_2(x,t)$ and slow rate $\nu_f = 0.005$, and $t = 1000, 1270, 5000$. A *type 2* anomaly is shown at $t = 1270$, with $r_p(x,t) \geq \ell$. This anomaly was first detected at $t = 1259$ which is marked with a line and a "$*$".

The mean of $\rho_1(x)$ is increased at time $t \geq \tau$ at rate ν from -3 to 3. $\rho_2(x)$ is a bidmodal distribution which changes to a single mode distribution at rate ν at time $t \geq \tau$. We use the properties of a single receptor calculated in section 4 to choose parameters to ensure appropriate behaviour. The standard normal distribution is set as the kernel and the parameters as follows: $w = 500$; $h = 1.9$; $\epsilon = 1.5{\times}10^{-3}$; $\beta = 0.01$; $\ell = 0.1$ $b = 1.95{\times}10^{-3}$; $d = 3.91{\times}10^{-4}$; $a = 4.67{\times}10^{-7}$; $g = 4$; $\Delta = 1$; $\tau = 1000$. For each test distribution we examine a fast rate ν_f and a slow rate ν_s. The results are given: $\rho_1(x,t)$, $\nu_s = 0.001$ in figure 4; $\rho_1(x,t)$, $\nu_f = 0.005$ in figure 5; $\rho_2(x,t)$, $\nu_s = 0.001$ in figure 5; $\rho_2(x,t)$, $\nu_f = 0.0075$ in figure 6. The results demonstrate that we are able to track with appropriate rates of change of the underlying distribution. However if the distribution changes too quickly we can detect a *type 2* anomaly.

8 Conclusions

We have presented some biological background of TCR signalling and highlighted features of potential to AIS. Taking a subset of these features we have designed a static anomaly detection method which we have shown equivalent to a conventional statistical anomaly *type 1* detection technique. This static system has

been extended with dynamic recurrence equations. We have demonstrated the dynamic system is able to perform kernel density estimation and detection of *type 2* anomalies. Some of the ideas presented here are either introductory or proof of concept, much further development is required. Particularly the dynamic case requires greater analysis to exactly connect $r_p(\mathbf{x}, t) \geq \ell$ to γ (equation 19) and changes in underlying distribution. In experiments, some not reported here, we are able to pick parameters to achieve *type 1* or *type 2* detection, but found difficulty in satisfying both simultaneously. A route forward is via the recurrence equations, greater analysis and perhaps a different choice of equations may aid in combining *type 1* and *type 2* detection. Once the anomaly detection system is more concrete we will investigate implementation issues and derive time and space complexities. We note that returning to the biological models with insights gained for kernel density estimation may prove of interest.

Acknowledgements. This work is sponsored by EPSRC Grant: EP/E005187/1.

References

1. Germain, R.N., Stefanov, I.: The dynamics of T cell receptor signaling: complex orchestration and the key roles of tempo and cooperation. A. Rev. Imm., 17 (1999)
2. Altan-Bonnet, G., Germain, R.N.: Modeling T cell antigen discrimination based on feedback control of digital ERK responses. PLoS Biol. 3, 356 (2005)
3. Owens, N.D.L., Timmis, J., Greensted, A., Tyrrell, A.: Modelling the Tunability of Early T cell Signalling Events. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 12–23. Springer, Heidelberg (2008)
4. Owens, N.D.L., Timmis, J., Greensted, A., Tyrrell, A.: Elucidation of T Cell Signalling Models. Submitted to Journal of Theoretical Biology (2009)
5. Feinerman, O., Veiga, J., Dorfman, J.R., Germain, R.N., Altan-Bonnet, G.: Variability and Robustness in T cell Activation from Regulated Heterogeneity in Protein Levels. Science 321 (2008)
6. Silverman, B.W.: Density Estimation for Statistics and Data Analysis. Chapman & Hall, Boca Raton (1986)
7. Bishop, C.M.: Novelty Detection and neural network validation. IEE Proceedings of Vision, Image and Signal Processing 141, 4 (1994)
8. Timmis, J., Andrews, P., Owens, N., Clark, E.: An Interdisciplinary Perspective on Artificial Immune Systems. Evolutionary Intelligence 1(1), 5–26 (2008)
9. Stepney, S., Smith, R.E., Timmis, J., Tyrrell, A.M., Neal, M.J., Hone, A.N.W.: Conceptual Frameworks for Artificial Immune Systems. Int. J. Unconventional Computing 1(3), 315–338 (2005)
10. Stibor, T.: An Empirical Study of Self/Non-Self Discrimination in Binary Data with a Kernel Estimator. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 352–363. Springer, Heidelberg (2008)
11. Duda, O.R., Hart, P.E., Stork, D.G.: Pattern Classification. Wiley & Sons, Chichester (2001)
12. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford Univ. Press, Oxford (1995)

An Immuno-engineering Approach for Anomaly Detection in Swarm Robotics

HuiKeng Lau[1,3], Iain Bate[1], and Jon Timmis[1,2]

[1] Department of Computer Science, University of York
[2] Department of Electronics, University of York,
Heslington, YO10 5DD, UK
[3] School of Engineering and IT, Universiti Malaysia Sabah, Malaysia

Abstract. In this paper, we present the first stage of our research strategy to develop an immune-inspired solution for detecting anomalies in a foraging swarm robotic system with an immuno-engineering approach. Within immuno-engineering, the initial stage of our research involves the understanding of problem domain, namely anomaly detection, in a foraging swarm robotic system deployed in dynamic environments. We present a systematically derived set of activities for this stage derived with Goal Structuring Notation and results of experiments carried out to establish the time-varying behaviour and how anomalies manifest themselves. Our future work will then be used to select and tailor an appropriate AIS algorithm to provide an effective and efficient means of anomaly detection.

Keywords: Immuno-engineering, swarm robotics, foraging.

1 Introduction

Research in swarm robotics has seen an increase in recent years due to its huge potential in various applications from medical, industrial, civilian and military to deep water and space exploration. A large amount of research funding has been awarded, for example a recent project called the SYMBRION project [10] supported by the European Commission. In a swarm robotic system (SRS), the achievement of a collective task is through the coordination of a group of simple homogeneous swarm that interact among themselves and with the environment [11]. Research into swarm robotics emerges from the inspiration of system-level functioning of social insects (ants, wasps, termites) that demonstrates the characteristics of robustness, flexibility, and scalability [2]. Insect communities are able to survive even after losing large numbers of insects (workers) or when faced with environment changes. In addition, they are very flexible in that they are able to solve foraging, prey retrieval and chain formation problems with the same base self-organised mechanism [2]. The increase or reduction in the number of individuals seem not to have severe impact on the operation of the community [2]. And thus, our intention is to attain these same desirable properties for the system-level operation of a SRS [12]. To distinguish swarm robotics from other terms being used to describes different approaches used in multi-robot systems,

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 136–150, 2009.

Sahin *et al* [12] outlines four distinguishing characteristics of swarm robotics that are considered representative for the purposes of our research. These characteristics, quoted verbatim, are:

- **Coordination of swarm:** Coordination mechanisms should be scalable to various swarm sizes.
- **Homogeneous robot:** Robots should be identical, at least at the level of interactions. Heterogeneous multi-robot systems fall outside of the swarm robotics approach.
- **Simplicity:** Robot should be simple in capabilities in relative to the task (not necessary hardware or software complexity).
- **Local interaction:** Individuals should have local interaction to ensure distributed coordination and scalability.

Swarm robotics is essentially an instance of distributed autonomous systems where a collection of interacting entities work together to accomplish certain goals without centralised control [13]. Each entity interacts with its environment and other entities through local communication but acts with some degree of autonomy [9]. For any system, given a reasonably well understood operational environment it is bound to experience undesirable behaviour or anomalies. These undesirable behaviours can be caused by random errors in hardware components, design errors, or deliberate sabotage [17]. To ensure normal operation of a system, detecting abnormal behaviour or malicious activities is very important. In current swarm robotics research, anomalies are normally handled through redundancy of robots that results in fault tolerance and no detection is carried out [17]. Such an approach assumes that anomalies occur at a rate that will not prevent the completion of a collective task and that individual robots with anomalies do not have an undesirable effect on the behaviour of the overall swarm. In this work we address situations where these assumptions may not hold. In such cases, detection of anomaly may help in deciding whether to continue with current task or switch to another task. Most work in SRS usually considered normal behaviour of the whole swarm. However as shown by [3] and [6], the issue of anomaly detection is an area that is starting to receive more attention from researchers.

There are significant challenges in detecting anomalies in swarm robotics in particular those deployed in dynamic environments. Due to the nature of SRSs in dynamic environments, anomaly detection systems (ADS) for such systems must fulfil the requirements of accuracy, responsiveness, resource usage and robustness [8]. With a lack of support from existing ADS to the above mentioned requirements, we decided to look at biological distributed autonomous systems that operates in analogous environment for inspiration. This is also in part influenced by research into artificial immune systems (AIS) that has showed that the immune system exhibits the properties we wish to endow the ADS in swarm robotics with. In approaching this problem, a systematic and principled approach of developing AIS called immuno-engineering [16] is adopted. One lesson from the thesis of Andrews [1] is that a firm understanding of the problem domain is needed before developing an immune inspired solution, therefore we focus our

initial efforts at a detailed understanding of the domain first: our findings are reported in this paper. The novel contribution of this paper is considered to be showing how the initial stage of immuno-engineering in *understanding the problem domain* [4] can be systematically derived through empirical experimental means, i.e., the nature of anomalies, and the time-varying nature of the environments. Specifically our work addresses the problem of *Modelling of Information Processing* described in [16].

Section 2 introduces immuno-engineering and how an instantiation of immuno-engineering is carried out in this research with research plan derived using Goal Structuring Notation (GSN) [7]. Section 3 and section 4 report the set of experiments carried out to establish the time-varying behaviour (TVB) and anomalies in a foraging SRS. Section 5 is the conclusion.

2 Immuno-engineering

The concept of immuno-engineering was proposed as an approach to develop biologically grounded and theoretically understood AIS through an interdisciplinary collaboration to capture the richness offered by immune system as opposed to weak analogy of the immune process they are based [16]. Immuno-engineering is defined in [16] as

> *The abstraction of immuno-ecological and immuno-informatics principles, and their adaptation and application to engineered artefacts (comprising hardware and software), so as to provide these artefacts with properties analogous to those provided to organisms by their natural immune systems.*

Immuno-engineering involves the adoption of conceptual framework [15] in AIS algorithm development and the problem-oriented perspective [4] in developing engineered AIS solutions. Thus an understanding of a problem domain is crucial in determining the type of AIS solution to develop. With a problem-oriented perspective in mind, the first stage of this research involves the investigation on the TVB and anomalies in a foraging SRS and how these behaviour exhibit themselves. In this research, GSN [7] is adopted to derive the set of activities required to answer the research questions. The advantage and motivation is that problem derivation with GSN allows the research problem to be successively broken down into smaller goals or objectives to the level where they are supported by the defined set of activities or solutions. This allows us to systematically derive research questions to be answered until we reach a level of detail that clearly derives a set of experiments, and associated experimental conditions, that need to be performed. Worth mentioning here that GSN can be used for any problem and not only in the context of AIS. Figure 2 shows the set of activities derived with GSN to establish the TVB and anomalies in a foraging SRS.

2.1 Overview of GSN

GSN was originally derived for use in the production of safety cases as part of the certification of systems. During the establishment of the safety argument's

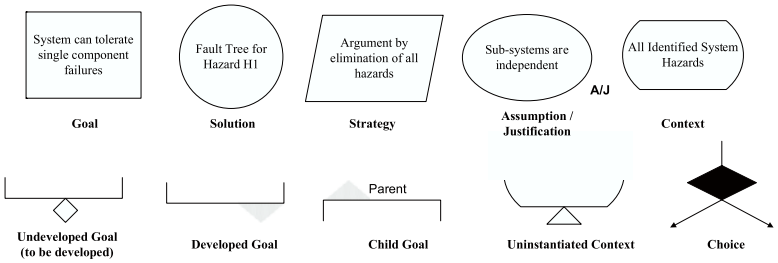


Fig. 1. Principal Elements of the Goal Structuring Notation

claims (often referred to as goals), context, assumptions and justifications are captured which has a number of uses including managing change. The GSN [7] - a graphical argumentation notation - explicitly represents the individual elements of any safety argument (requirements, claims, evidence and context) and, perhaps more significantly, the relationships that exist between these elements (i.e. how individual requirements are supported by specific claims, how claims are supported by evidence and the assumed context that is defined for the argument). The principal symbols of the notation are shown in Figure 1 (with example instances of each concept).

The principal purpose of a goal structure is to show how goals (claims about the system) are successively broken down into sub-goals until a point is reached where claims can be supported by direct reference to available evidence (solutions). The solutions, and their parent claims, map onto the experimental questions to be answered and hence the test cases needed. As part of this decomposition, using the GSN it is also possible to make clear the argument strategies adopted (e.g. adopting a quantitative or qualitative approach), the rationale for the approach (i.e. assumptions, justifications) and the context in which goals are stated (e.g. the system scope or the assumed operational role). In our work these *non-spinal elements* help define the experimental conditions relevant. It is noted in our work not all symbols are used.

2.2 Overview of Research Strategy

The first stage of developing an immune-inspired ADS for a foraging SRS with immuno-engineering involves the identification and understanding of the TVB and anomalies in such a system. Figure 2 shows the set of activities derived for this stage with GSN where Gx refers to the research goal, Cx refers to context and Snx refer to an activity or solution. In order to establish G1, the experimental strategy is to set-up simulation environment (G2), simulating scenarios in a static environment (G3), dynamic environments without anomaly (G4), and dynamic environments with anomaly (G7). G4 and G7 are both further decomposed into defining sufficient test cases (G5) and identifying key features (G6). The static environment serves as the baseline for comparison to observe the deviations when TVB and anomalies are introduced into the system. As part of

breaking down the goals is establishing context. For instance, goal G1 has context concerning what TVB is (C1) and overall task which is foraging (C2). The three TVB to establish dynamic environments are:

- **Varying Food Growth Rate** (V_{FGR})**:** The growth of food in the arena changes at different time interval.
- **Varying Food Distribution** (V_{FD})**:** The concentration of food redistributed in the environment is biased towards certain region.
- **Varying Presence of Obstacles** (V_{PO})**:** The presence of obstacles at different time interval affect the time required to reach the food, avoid obstacles, and carrying the food back to the base.

In terms of anomalies, random errors due to hardware malfunction are investigated focusing on robot wheels, food grippers and the communication device, labelled as C3 in Figure 2. We only considered hardware malfunctions in this stage to focus on one type of failure before proceed with more complicated failures such as those influenced by control software and environmental factors. The failures in both robot wheels and grippers have direct and immediate impact on the foraging task. They also span both subtle transient failures and much easier failures - permanent or significant in size. Whilst it is not claimed these are complete, they do represent a reasonable comprehensive coverage. Thus, the scale of failure is divided into the following:

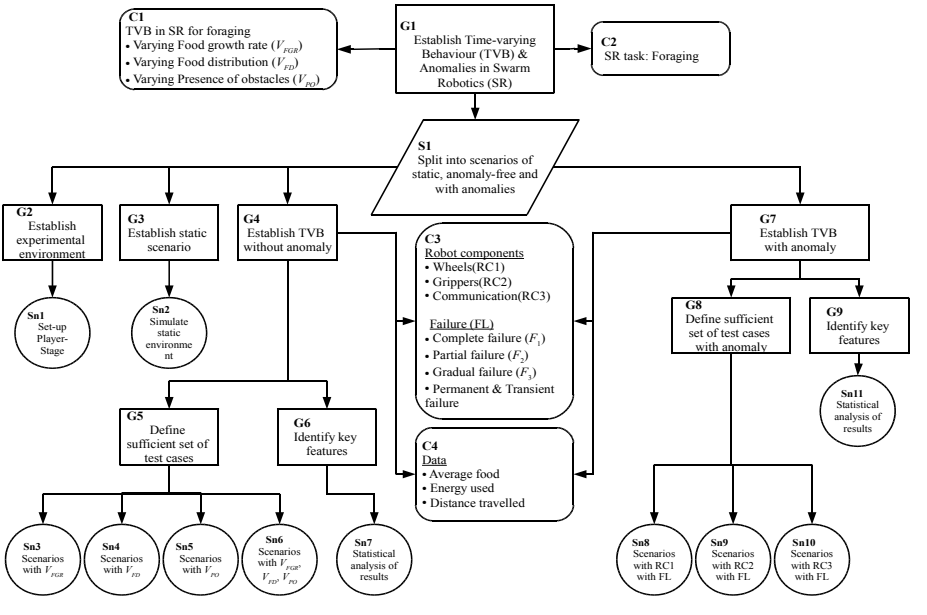


Fig. 2. Derived set of activities with GSN to investigate the TVB and anomalies in a foraging SRS

- **Complete failure (F_1):** Robot component stop responding completely. For instance in robot wheels, this failure may causes the wheels to halt to a complete stop or the robot turning with a fix angle causing it to move in a circle indefinitely.
- **Partial failure (F_2):** Robot component functioning less efficiently. In the case of robot wheels, the moving speed might be set to certain value x m/s.
- **Gradual failure (F_3):** Robot component fail slowly. For example, gradual failure in robot wheels involve gradual reduction in robot speed by y m/s for each simulation update.
- **Transient failure (F_4):** This type of failure involves alternate on and off occurrences of faulty robot component. For transient failure in robot wheels, the wheels might stop responding for z seconds and then start functioning normally before stop responding again. The component fault can be a complete, partial or gradual.

Data identified to be beneficial in detecting anomalies in foraging include the average unit of food collected (μ_{food}), distance travelled and energy used (C4). For all experiments, simulations for the swarm robotic system are implemented on the Player-Stage [14] - a software tool consists of the Player for the definition of robots and the Stage that simulates the population of robots in two-dimensional environment.

3 Time-Varying Behaviour Experiment

This section describes the simulations carried out to simulate the TVB in a foraging SRS without anomaly (G5). For clarification, following parameters are used in all experiments carried out in this paper.

- **Size of arena:** 10 m x 10 m
- **Normal robot moving speed:** 0.15 m/s
- **Initial food in arena:** 100 unit
- **Maximum food in arena:** 200 unit
- **Normal food growth rate (FGR):** 0.1 unit/simulation update
- **Each simulation update:** 200 microseconds

As mentioned in section 2.2, the simulation of foraging swarm robots in a static environment (Sn2) serves as the baseline for comparison to dynamic environments. Figure 3 shows the μ_{food} by each robot in such an environment. It is noted that individual robots behaviour cannot easily be distinguished, however as it is the differences in individual behaviour versus the swarm that is important then this is not considered to be an issue. In this paper, the μ_{food} is calculated over a time window of size four with each time slot of size 250 simulation seconds. These values are chosen from experimental results showed that they are most suitable for the problem of interest. In Figure 3, a difference up to 2 units of food (11.70%) is observed between the robots but the largest difference, which occurs around time 48, only lasts one or two clock cycles. This gives the potential to achieve more reliable detection by observing differences over a period of

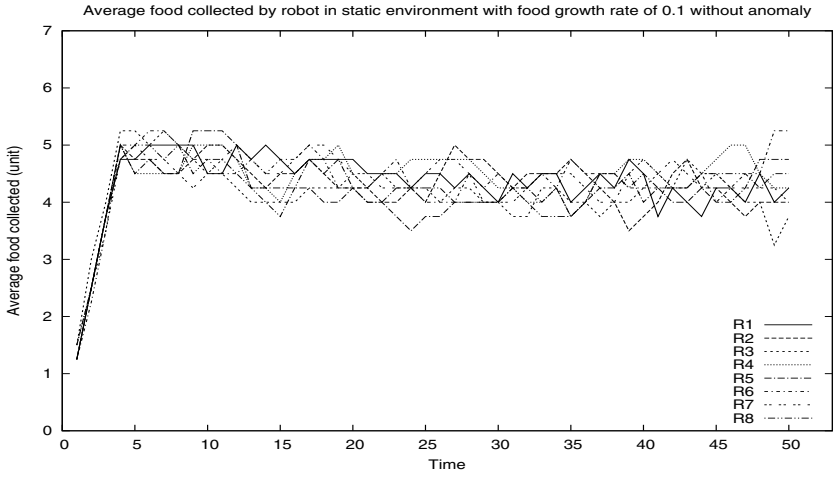


Fig. 3. μ_{food} in a static environment without anomaly

time. However, the overall pattern remained consistently similar throughout the experiment in all robots. This observation confirmed our initial belief that the μ_{food} by all robots is consistent within a local proximity and this information may proved to be useful in helping detecting anomalies.

When the same robots are placed in environments with V_{FGR} (Sn3), V_{FD} (Sn4), V_{PO}(Sn5), or combination of all three behaviour (V_{MIX}) (Sn6), the observed collection of food changes significantly as shown in Figure 4. In this simulation, the FGR (simulated with normal distribution random number generator provided in GNU Scientific Library (GSL)) is varied between 0.1 and 0.025. FGR of 0.1 means that for each simulation update, the probability of adding one unit of food into the arena is 0.1. Lower value of FGR means lower probability of adding new food into the arena. With FGR set to lower value, μ_{food} by each robot decreases as the concentration of food in the arena decreases.

The observations in Figure 4 as well as other simulations with V_{FD}, V_{PO} and V_{MIX} signify the huge influence of these TVB over the collective task. Although the pattern of food collected is very similar among the robots in all scenarios, deviation from the local mean (calculated over all robots in the arena) from 2.82% to 33.29% is observed. Again the larger difference were observed to be short lived. Noise of such magnitude make the development of an ADS with high detection rates and low false alarms a very challenging task. Table 1 is a detailed information regarding the maximum and minimum standard deviation from local mean calculated for all five simulated scenarios. It can be seen that the deviations among robots with scenarios of single TVB (V_{FGR}, V_{FD}, V_{PO}) are very similar. When the environment becomes more dynamic (V_{MIX}), it becomes more unstable and thus higher deviation in food collection by robots is observed.

Table 2 shows the distance travelled and energy used by each robot under both static and dynamic environments. It can be seen that depending on the type

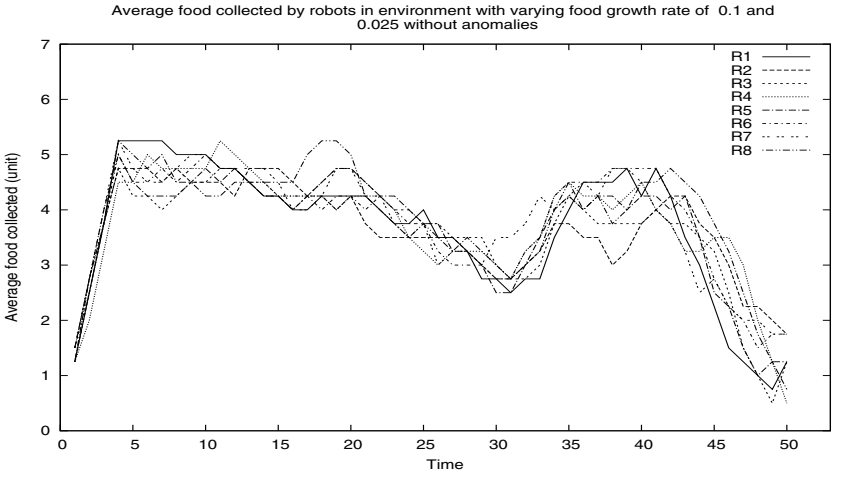


Fig. 4. μ_{food} by robots in a dynamic environment with V_{FGR} without anomaly. In this experiment, FGR is varied between 0.1 and 0.025. At time t=[10,20) and t=[30,40), the FGR is set to 0.1 while at time t=[20,30) and t=[40,50) it is set to 0.025.

Table 1. Minimum and maximum percentage of deviation, σ, between the mean for all eight robots in the arena and their individual values

Scenario	Static	V_{FGR}	V_{FD}	V_{PO}	V_{MIX}
σ_{min} (%)	3.35	4.88	6.22	3.54	6.95
σ_{max} (%)	11.70	25.41	25.57	20.06	33.29

of TVB simulated, robots might move either with greater distance or shorter distance compared to the one in a static environment. Simulations so far have showed that V_{FGR} and V_{FD} resulted in a greater distance travelled by all robots. With V_{PO}, one explanation as to why less distance was travelled by each robot might be that time was wasted to avoid the obstacles instead of wandering the arena to search and collect food. This is similar to the case with V_{MIX}. In terms of energy usage, results in Table 2 indicate that robots are consistently consuming less energy for all four scenarios in a dynamic environment compared to a static environment. This is largely due to the fact that the number of food collected by each robot is less in dynamic environments and carrying of food consumes more energy than wandering around the arena.

The TVB experiment showed the nature of dynamic environments and the effects of the TVB on the foraging task. It can be concluded that dynamic environments pose significant challenges and influence greatly the ability of an ADS in detecting anomalies.

Table 2. Distance travelled and energy usage by robots in a static and dynamic environments. For energy usage, object avoidance and moving around in the arena without food in the grippers cost 1 unit of energy. Moving with food in the grippers consumes 1.5 unit of energy while robot in resting state only consumes 0.1 unit of energy.

Metric	Distance (meter)					Energy (unit)				
Robot	Static	V_{FGR}	V_{FD}	V_{PO}	V_{MIX}	Static	V_{FGR}	V_{FD}	V_{PO}	V_{MIX}
R1	1565.36	1590.38	1610.56	1517.33	1557.97	15131.7	14662.7	14423.8	14923.5	14557.9
R2	1561.66	1589.48	1586.09	1512.09	1548.55	15107.0	14780.8	14295.2	14969.2	14575.2
R3	1569.17	1595.31	1584.71	1521.10	1531.81	15103.6	14818.2	14342.4	15060.9	14369.1
R4	1566.37	1585.18	1600.88	1523.31	1553.73	15209.3	14787.9	14352.0	15005.2	14444.7
R5	1566.62	1592.64	1602.72	1520.98	1507.24	15113.7	14760.6	14426.3	14929.3	14456.6
R6	1570.64	1610.90	1592.42	1528.34	1544.99	15086.8	14791.9	14372.3	15031.7	14441.6
R7	1568.66	1585.28	1589.67	1488.28	1565.98	15167.2	14748.8	14358.2	14997.3	14498.0
R8	1562.83	1597.19	1593.70	1522.35	1540.28	15143.0	14793.4	14329.6	14965.3	14402.6

4 Anomaly Experiment

To establish anomalies in a foraging SRS in dynamic environments (G7), four scales of malfunctions to robot components are investigated (C2). In all simulations conducted for G7, the anomaly (F_1, F_2 and F_3) is injected into robot R5 at time t=25 while F_4 is injected at time t=[15,25) and t=[40,50). Due to space limitation, we only present the results of simulating failure in robot wheels (Sn8) in this paper. Table 3 shows the details about the type of anomaly, parameter and time when the anomalies are injected. These anomalies are simulated for all four dynamic environments.

Table 3. Test cases for TVB with anomaly

Anomaly	Parameter	Time
Permanent failure, F_1	Speed=0.15 m/s, turn angle = left 10°	t=[25,50)
Partial failure, F_2	Speed={0.05, 0.1, 0.11}m/s	t=[25,50)
Gradual failure, F_3	Speed reduction per simulation update ={0.00001, 0.00005, 0.0001}m/s	t=[25,50)
Transient failure, F_4 (with F_1, F_2 or F_3)	Same as F_1, F_2 and F_3	t=[15,25) and t=[40,50)

When F_1 is injected into a robot, the effect is immediate as illustrated in Figure 5 for robot R5. Significant drop in μ_{food} for R5 can be observed and the value reduced to zero at time t=29. This type of failure should be easier to detect and is included as a baseline for comparison with other type of failures. Similar results are expected when such failure is introduced in other scenarios with V_{FD}, V_{PO} and V_{MIX}.

For partial failure (F_2), the severity of partial failures has significant influence on the possibility of detecting the anomaly. If the partial failure is more critical, significant deviations can be observed over a period of time. However

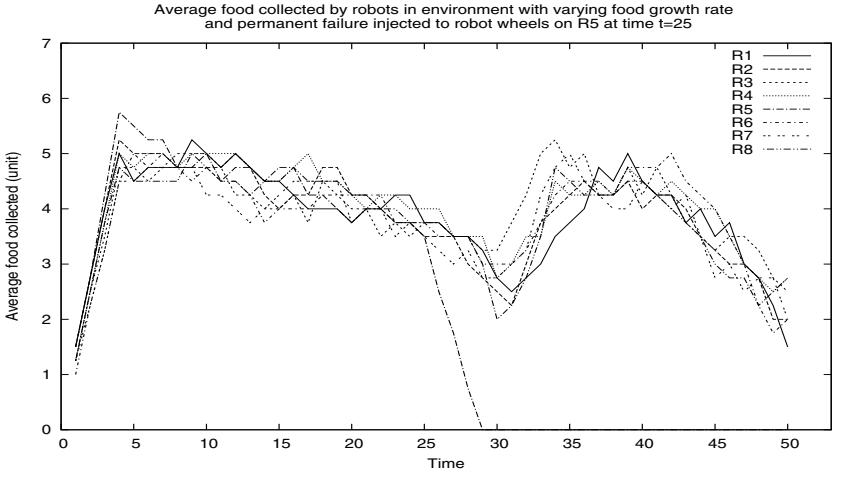


Fig. 5. μ_{food} by robots in a dynamic environment with V_{FGR} and F_1 is injected to robot wheels on R5 at time t=25. F_1 in this scenario is setting the turning angle of R5 to left turn 10° causing the robot to move in a circle.

when the partial failure is more subtle, such as setting the value of F_2 to 0.11 m/s (normal is 0.15 m/s), the differences are not that obvious and can be very difficult to be spotted as illustrated in Figure 6. One might be tempted to further experiment with F_2 value set to be as close as possible to the normal speed such as 0.13 or 0.14 m/s. However, such an action is questionable since such subtle

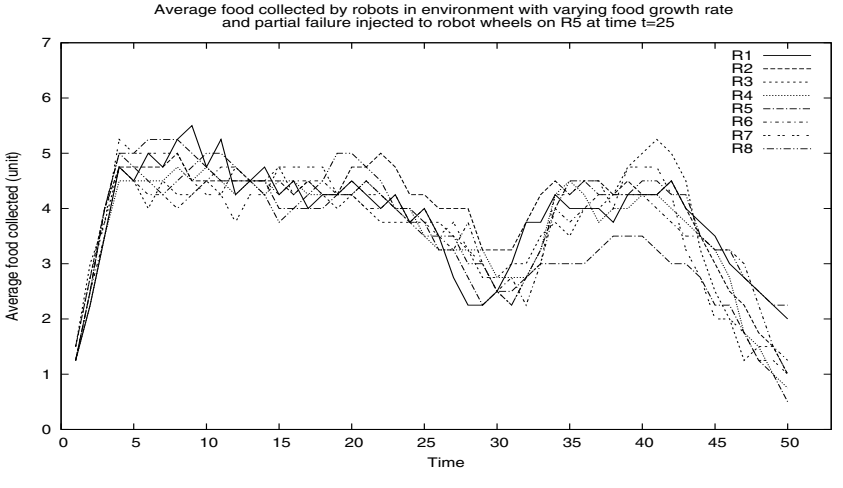


Fig. 6. μ_{food} by robots in a dynamic environment with V_{FGR} and F_3 is injected to robot wheels of R5 by setting the moving speed to 0.11 m/s at time t=25

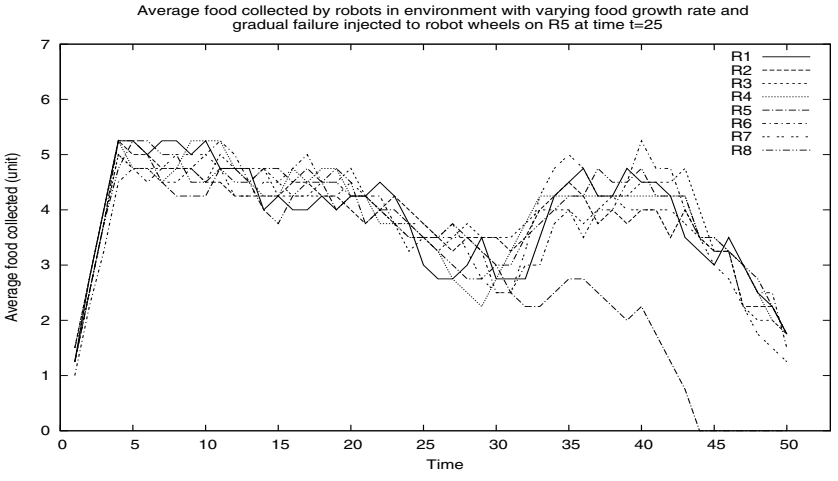


Fig. 7. μ_{food} by robots in a dynamic environment with V_{FGR} and gradual failure F_2 of 0.00001 m/s is injected to robot wheels of R5 at time t=25

difference might not be suitable to be classified as failure especially in real world applications where navigation surfaces are not exactly the same for all robots.

F_3 represents failures in the robot components that occur gradually instead of immediate failure as in F_1 and F_2. Such anomalies have a greater impact on the responsiveness of an ADS since it takes some time before any differences can be detected. Similar to F_2, this type of failure will prove to be very difficult to detect if the gradual failure progress very slowly over a long period of time. Figure 7 shows such an example where significant deviation is only apparent after time $t \geq 32$. Therefore, in detecting such anomalous behaviour over a period of time (long term) is more beneficial instead of looking at values at each time instance. This is also the reason why we chose to use average value instead of exact quantity of food collected for each time period. As discussed in the previous section this would also help remove exceptional, but short-lived, measures within the normal behaviour. Such characteristic is similar to biological immune system that operates on multiple timescales to protect the body.

Transient failure (F_4) is the type of error that occurs periodically instead of permanently as in F_1 to F_3. Such failures can happen to robot swarms in an environment with inconsistent environmental conditions. It maybe a case where a few sections on a long path between the base and the food source are covered with rough and uneven surfaces or there exist electromagnetic interference to the sensors. In such situation, ADS developed needs to be able to learn such behaviour and able to provide faster responses on second encounter. Figure 8 shows an example of F_4 simulated in an environment with V_{FGR}. Two occurrences of anomaly can be seen from the graph. Both of these have slightly different failure patterns even though they are injected with the same F_2 value. This observation

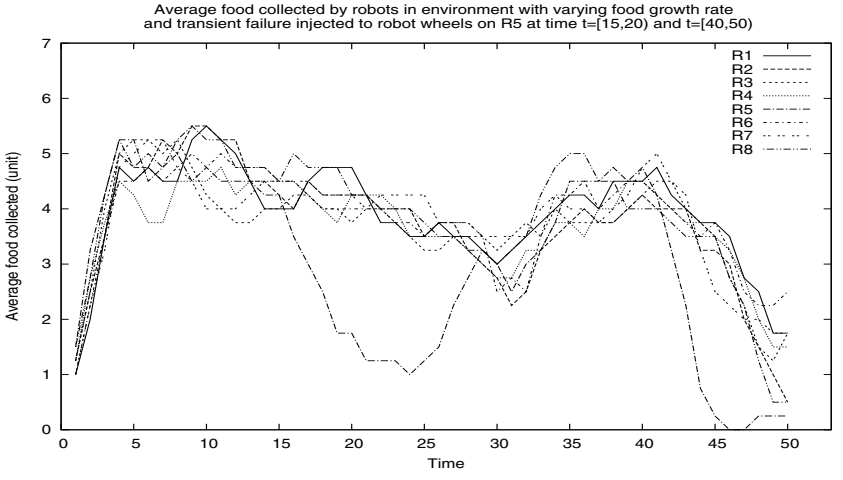


Fig. 8. μ_{food} by robots in a dynamic environment with V_{FGR} and F_4 is injected to robot wheels by R5 by setting moving speed to 0.05 m/s at time t=[15,25) and t=[40,50)

Table 4. Minimum and maximum percentage of deviation, σ (for μ_{food} by robots), from local mean in simulations with faulty robot wheels. Mod range is the range where three quarters of the deviations are located.

Scenarios	σ_{lmin} (%)	σ_{lmax} (%)	mod range (%)	σ_{R5min} (%)	σ_{R5max} (%)	mod range (%)
F_1+V_{MIX}	5.1	35.1	6.0-24.0	25.0	100	85.0-100
F_2+V_{MIX} (speed=0.05 m/s)	5.8	34.9	5.0-20.0	19.7	100	62.0-90.5
F_2+V_{MIX} (speed=0.10 m/s)	4.6	35.8	4.0-16.8	8.7	53.3	10.8-43.2
F_2+V_{MIX} (speed=0.11 m/s)	3.9	36.5	6.4-23.4	3.8	44.4	4.5-31.5
F_3+V_{MIX}(speed-=0.00001 m/s)	7.8	29.0	11.4-24.6	10.0	100	17.2-100
F_3+V_{MIX} (speed-=0.00005 m/s)	3.6	29.4	5.7-21.9	25.8	100	90.0-100
F_3+V_{MIX} (speed-=0.0001 m/s)	4.8	30.0	4.0-17.5	9.7	100	90.9-100
F_4+F_1+V_{MIX}	6.8	33.1	6.0- 25.0	9.1	100	90.9-100
F_4+F_2+V_{MIX} (speed=0.11 m/s)	4.7	32.2	9.8-18.5	5.0	52.7	12.2-35.0
F_4+F_3+V_{MIX} (speed-=0.00001 m/s)	6.8	27.6	10.4-23.6	9.9	61.1	6.1-48.8

demonstrates the unpredictability of failures identified and the need for an ADS that is able to adapt accordingly.

Table 4 summarises the minimum and maximum percentage of deviations from local mean in scenarios identified to pose the greatest challenges in detecting anomalies for faulty robot wheels. In this table *speed* refers to the moving speed of robot injected with a failure. By looking at the minimum and maximum deviations for anomalous and normal robots, it is apparent that in many cases the σ_{R5min} is less than σ_{lmax}. This means that the minimum deviation from local mean for anomalous robot R5 is still within observed normal behaviour. However, since these σ_{R5min} are only obtained from the first few slots after the

anomaly was injection such pattern is expected (due to the fact that μ_{food} is calculated over four slots).

To analyse the implicit nature of anomalies, we analysed the deviations observed over the full duration of anomalies and tabulated the range where three quarters of the deviations are located as *mod range*. To interpret the significant of mod range, we take first row of Table 4 as an example. In this case, three quarters of deviations from local mean by normal robots are in the range of 6.0% to 24.0%. However, the range for the anomalous robot R5 is between 85.0% to 100%. Thus, to detect anomalies any deviation greater than 24% from local mean maybe considered as anomalous. Looking at the mod range column for normal robots, it is apparent that the maximum range is only up to 25% from local mean as opposed to 100% for R5. Thus, it can be seen than deviation of more than 25% from local mean can be use as a threshold to differentiate between anomalous and normal instances. Similar analysis were carried out for distance travelled and energy used by normal robots and anomalous robot R5. It was discovered that deviation of more than 1.02% and 4.45% from local mean can be used as threshold for energy used and distance travelled respectively to differentiate anomalies. We also analysed the data using other methods such as Jacobson/Karels algorithm [5] that combines the short-term (based on the standard deviation between moving average and current sample) and long-term (moving average) effects of collected data but found it to be inappropriate for our SRS due to noise and inconsistency of data in dynamic environments. Besides this, analysis was also conducted to compare the local mean and individual robot internal mean (calculated over the four time slot). Again, standard deviation between the two values is apparent and should be useful in detecting anomalies.

The anomaly experiment showed the difficulties in differentiating between normal and anomalous instances in dynamic environments. Our preliminary analysis has discovered that deviations from local mean of more than 25% for μ_{food}, 1.02% for energy usage and 4.45% for total distance travelled can be used as threshold to differentiate between normal and anomalous instances. At the same time, looking using a detector over a short period of time (typically 2-3 clock cycles) can help make differences more significant. Our current work is continuing the experimentation to complete the specification. With the requirements of acceptable accuracy, response within time, lightweight and robust to dynamic environment in mind, our next stage of research involves the identification of suitable immune models that addresses such issues.

5 Conclusion

This paper has introduced immuno-engineering as a principled approach to developing artificial immune systems. Immuno-engineering advocates an initial problem-oriented perspective so as to better understand the problem domain, so that an effective, and importantly, tailored AIS can be developed. Rather than looking first at the immunology to see what can be used to develop an algorithm, the adoption of the immuno-engineering approach forces us to examine

carefully the application area first. In this paper we have taken the first steps to understanding the problem of anomaly detection in swarm robotic systems with time varying behaviour. We have made use of a well established technique known as Goal Structured Notation (GSN) to derive the set of information needed for us to establish an understanding the problem domain: from our experience we would advocate the use of the GSN within the context of immuno-engineering. With this information establish, a comprehensive series of experiments were defined to give the raw data needed to analyse the domain and hence create a set of requirements for any anomaly detection system. Our initial analysis of the swarm behaviour data showed the effects of dynamic environments on the collective task, the nature of anomalies and the need for learning and adaptation in detecting anomalies in a foraging SRS.

References

1. Andrews, P.: An investigation of a methodology for the development of artificial immune systems. Ph.D thesis, Department of Computer Science, University of York (2009)
2. Bayindir, L., Sahin, E.: A review of studies in swarm robotics. Turkish Journal of Electrical Engineering and Computer Sciences 15(2) (2007)
3. Christensen, A.: Fault detection in autonomous robots. Ph.D. dissertation (2008)
4. Freitas, A., Timmis, J.: Revisiting the foundations of artificial immune systems for data mining. IEEE Trans. on Evolutionary Computation 11(4), 521–540 (2007)
5. Jacobson, V.: Congestion avoidance and control. ACM Computer Communication Review 18, 314–329 (1988)
6. Jakimovski, B., Maehle, E.: Artificial immune system based robot anomaly detection engine for fault tolerant robots. In: Rong, C., Jaatun, M.G., Sandnes, F.E., Yang, L.T., Ma, J. (eds.) ATC 2008. LNCS, vol. 5060, pp. 177–190. Springer, Heidelberg (2008)
7. Kelly, T.: Arguing Safety - A Systematic Approach to Safety Case Management. Ph.D thesis, Department of Computer Science, University of York (1999)
8. Lau, H.K., Timmis, J., Bate, I.: Anomaly detection inspired by immune network theory: A proposal. In: Proc. of the 2009 IEEE Congress on Evolutionary Computation (IEEE CEC 2009). IEEE, Los Alamitos (2009)
9. Liu, Y., Passino, K.M.: Swarm intelligence: Literature overview. Technical report, Ohio State University (March 2000)
10. SYMBRION REPLICATOR, `http://www.symbrion.eu/`
11. Sahin, E.: Swarm Robotics: From Sources of Inspiration to Domains of Application. In: Şahin, E., Spears, W.M. (eds.) Swarm Robotics 2004. LNCS, vol. 3342, pp. 10–20. Springer, Heidelberg (2005)
12. Sahin, E., Girgin, S., Bayindir, L., Turgut, A.E.: Swarm Robotics. In: Blum, C., Merkle, D. (eds.) Swarm Intelligence: Introduction and Applications. Natural Computing Series, pp. 87–100. Springer, Heidelberg (2008)
13. Segel, L., Cohen, I.: Design Principles for the Immune System and Other Distributed Autonomous Systems. Oxford University Press, USA (2001)
14. Stage-Player, `http://playerstage.sourceforge.net/`
15. Stepney, S., Smith, R., Timmis, J., Tyrrell, A., Neal, M., Hone, A.: Conceptual frameworks for artificial immune systems. Int'l Journal of Unconventional Computing 1(3), 315–338 (2005)

16. Timmis, J., Hart, E., Hone, A., Neal, M., Robins, A., Stepney, S., Tyrrell, A.: Immuno-engineering. In: Proc. Second IFIP Int'l Conf. on Biologically Inspired Collaborative Computing, pp. 3–17. Springer, Heidelberg (2008)
17. Winfield, A., Nembrini, J.: Safety in numbers: Fault tolerance in robot swarms. Int'l Journal on Modelling Identification and Control 1(1), 30–37 (2006)

An Immune-Inspired Approach to Qualitative System Identification of the Detoxification Pathway of Methylglyoxal

Wei Pang[1,2] and George M. Coghill[2]

[1] College of Computer Science and Technology, Jilin University, Changchun, 130012, P.R. China
pangwei@jlu.edu.cn
[2] Department of Computing Science, University of Aberdeen, Aberdeen, AB24 3UE, Scotland, UK
{pang.wei,g.coghill}@abdn.ac.uk

Abstract. In this paper, a *qualitative model learning* (QML) system is proposed to qualitatively reconstruct the detoxification pathway of Methylglyoxal. First a converting algorithm is implemented to convert possible pathways to qualitative models. Then a general learning strategy is presented. To improve the scalability of the proposed QML system and make it adapt to future more complicated pathways, an immune-inspired approach, a modified clonal selection algorithm, is proposed. The performance of this immune-inspired approach is compared with those of exhaustive search and two backtracking algorithms. The experimental results indicate that this approach can significantly improve the search efficiency when dealing with some complicated pathways with large-scale search spaces.

1 Introduction

1.1 Qualitative Model Learning

Qualitative Model Learning (QML) is a branch of Qualitative Reasoning [1,2]. It involves extracting qualitative models of dynamic systems from available observed data, which can be either quantitative or qualitative.

The models that QML aims to infer are in the form of Qualitative Differential Equations (QDEs). A QDE is an abstraction of a class of Ordinary Differential Equations (ODEs), in the sense that a corresponding ODE can be obtained by parameterizing and quantizing this QDE.

A QDE is the conjunctions of a set of *qualitative constraints*, which link the *qualitative variables* in the model and express the relations among these variables. *Qualitative variables* are different from variables in quantitative models in the sense that they can only take *qualitative values* from their associated *quantity spaces*. Qualitative values can be defined as either landmark values and intervals between these landmark values [2], or fuzzy intervals [3]. The simplest and commonly used qualitative values are *qualitative signs* including *positive*, *zero*, and *negative*. These three qualitative values constitute the *signs quantity space*.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 151–164, 2009.

For the qualitative constraints, some of them are derived from mathematical relations, such as *addition*, *subtraction* and *multiplication*; others represent the incomplete knowledge about the function relations under study, such as M^+ (monotonically increasing function) and M^- (monotonically decreasing function) in QSIM (Qualitative SIMulation) [2], which state that one variable will monotonically increase with the increase (decrease) of another.

QML can be considered as an approach of system identification [4], thus it has an alternative name, *qualitative system identification*. QML is particularly useful in the situation when little knowledge about the system is known and only sparse experimental data are available. In the last two decades, several QML systems have been developed, such as MISQ [5], QSI [6], and ILP-QSI [7].

However, there exist two problems when applying QML to real-world applications. First, how to make the best use of domain-specific knowledge. None of the available QML systems can integrate domain-specific knowledge conveniently and explicitly. This limits their abilities to deal with various real-world applications. To solve the real-world problems in various domains, the special-purpose approaches which can make the best use of domain-specific knowledge are needed. Second, all of the above mentioned QML systems employ deterministic algorithms (such as the branch-and-bound algorithm in ILP-QSI) to search possible candidates in the search space, which might be intractable when dealing with the problems with large-sized search space. To address these two problems, it is necessary to developed a special-purpose QML system for a particular application.

1.2 Research on the Detoxification Pathway of Methylglyoxal

Methylglyoxal (MG) is a naturally occurring toxic electrophile that is harmful to cells. Excessive production of MG leads to cell death [8]. Most of the organisms protect themselves from the toxic effect of MG through the detoxification pathway, which has been initially studied by Booth et al. [9].

The current understanding of the detoxification pathway is shown in Figure 1. When MG crosses the cell membrane and enters the cell, glutathione (GSH) reacts spontaneously with MG to produce hemithioacetal (HTA). Catalysed by the enzyme glyoxalase I (GlxI), HTA is converted into S-lactoyl-glutathione (SLG). Catalysed by a second enzyme glyoxalase I (GlxII), SLG is converted into GSH and non-toxic D-lactate. The whole pathway is composed of three biochemical reactions: the first is a non-enzymatic reaction which follows the law of mass action; the second and third are enzymatic reactions which obey Michaelis-Menten kinetics [10]. Based on the above understanding, a quantitative model [11,12] which consists of four ordinary differential equations has been manually built.[1] However, it should be pointed out that the mechanism of the MG detoxification is not yet fully understood, and the research is still ongoing. Consequently the quantitative model is an intermediate one based on limited experimental data and incomplete knowledge. For instance, it does not consider the influence

[1] This quantitative model has not been published yet. So it will not be presented in this paper.

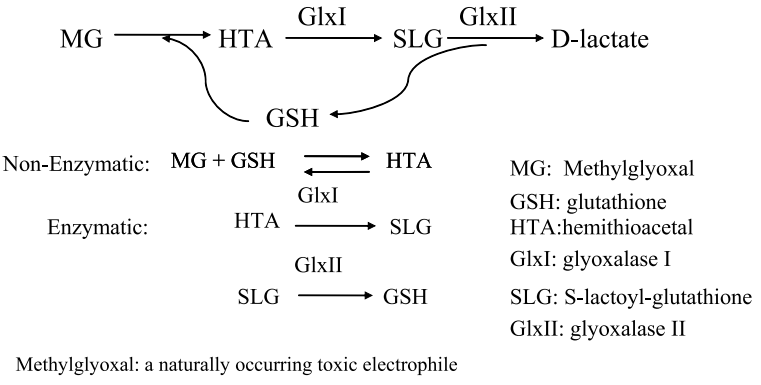


Fig. 1. The Detoxification Pathway of Methylglyoxal

of the KefB K^+ channel [13]. Furthermore, based on the latest experimental data[2], some biologists suggest that the MG detoxification pathway may be more complicated.

1.3 The Goals and Organisation of the Paper

As mentioned in the last section, the quantitative approach is not sufficient for modelling the MG detoxification pathway at the current stage. In this paper we will use the qualitative model learning as an alternative approach to study this pathway. There are two goals to be achieved: First, given only qualitative data and reasonable assumptions, we expect that qualitative model learning can effectively narrow down the range of candidate qualitative models. Second, we also expect the QML system can deal with large-scale pathways efficiently by making use of appropriate search strategies.

In order to achieve the above two goals, the rest of the paper is organised as follows: In Section 2 a converting algorithm which can convert possible pathways to qualitative models is presented. The qualitative data used in this paper is described in Section 3. In Section 4 a general learning strategy is proposed for the qualitative system identification of the MG detoxification pathway. Then a modified clonal selection algorithm is proposed in Section 5 to deal with the scalability of the learning system. This is followed by the experiments described in Section 6. Finally in Section 7 we conclude the paper and discuss the future work.

2 Converting from Possible Pathways to Qualitative Models

An algorithm is proposed to convert a pathway, which is composed of several biochemical reactions, to a qualitative model. First we make the standard

[2] Data obtained by Prof. Ian Booth's research group at Aberdeen University.

assumption that all non-enzymatic reactions obey the law of mass action and all enzymatic reactions follow Michaelis-Menten kinetics. Based on this assumption, the algorithms will convert all the reactions individually and obtain a QDE. The converting algorithm is composed of three sub-algorithms, which will be described in the rest of this section.

2.1 Algorithm A. Converting Non-enzymatic Reactions

For a non-enzymatic reaction A+B<–>C+D, according to the law of mass action, the reaction rate V is defined as follows:

$$\begin{aligned} V &= K_1[A][B] - K_2[C][D] \\ &= -\frac{1}{a} * \frac{d[A]}{dt} = -\frac{1}{b} * \frac{d[B]}{dt} = \frac{1}{c} * \frac{d[C]}{dt} = \frac{1}{d} * \frac{d[D]}{dt}, \end{aligned} \tag{1}$$

where K_1 and K_2 are the rate constants of the forward and backward reaction respectively; a, b, c, and d are stoichiometric coefficients; [A], [B], [C] and [D] stand for concentrations of the corresponding species.

Algorithm A first generates the following qualitative constraints according to formula 1:

$Aux1 = A * B$
$Aux2 = M^+(Aux1)$
$Aux3 = C * D$
$Aux4 = M^+(Aux3)$
$Aux5 = Aux4 - Aux2$

In the above, "*" and "-" stand for multiplication and subtraction respectively; M^+ is the monotonically increasing function. The variables with the prefix Aux are newly introduced auxiliary variables, which are used to "break" a mathematical expression to qualitative constraints.

Aux5 is then added as a positive derivative term to all the related substrates (A and B), and as a negative derivative term to all the related products (C and D), as shown below:

dA/dt =.....+$Aux5$
dB/dt=.....+$Aux5$
dC/dt =.....−$Aux5$
dD/dt=.....−$Aux5$

2.2 Algorithm B. Converting Michaelis-Menten Reactions

For an enzymatic reaction A–>B, the reaction rate V is defined as follows:

$$V = -\frac{d[A]}{dt} = \frac{d[B]}{dt} = V_{max} * \frac{[A]}{k_s + [A]} \tag{2}$$

In the above, V_{max} and k_s are constants. We can see that with the increase of the concentration of A, the reaction rate will increase too. This can be considered as a

monotonically increasing relation in the qualitative context. Based on the above description, Algorithm B first generates the following qualitative constraint:

$Aux6 = M^{+}(A)$

Then *Aux6* is added as a positive derivative term to B and a negative derivative term to A, as shown below:

$dA/dt = ... - Aux6$
$dB/dt = ... + Aux6$

2.3 Algorithm C. Processing the Derivative Terms

After all the reactions are processed by algorithms A and B, we will obtain the derivative terms for all species, for example, for species "A" we may get the following equation:

$$dA/dt = +Aux1 - Aux3 + Aux4 + Aux7, \tag{3}$$

Algorithm C will generate the following qualitative constraints for equation (3):

$Aux8 = Aux1 - Aux3$
$Aux9 = Aux8 + Aux4$
$dA/dt = Aux9 + Aux7$

For ease of implementation the above auxiliary variables are processed in order from left to right by Algorithm C, although in theory the processing can be performed in a random order.

Finally, the qualitative model is the conjunction of all qualitative constraints generated by Algorithms A, B and C.

2.4 The Qualitative Model for the MG Detoxification Pathway

Based on these three converting rules, any possible pathway can be converted to a qualitative model. For instance, the qualitative model for the MG detoxification pathway is given in Table 1. This qualitative model is an abstraction of the quantitative model proposed by de Moura et al. [11,12].

3 Qualitative Data

As the research on the MG detoxification pathway is still in progress, the experimental data are sparse and cannot be fully accessed. So it is not possible to use quantitative data directly in this paper. Based on the above consideration, currently qualitative data are derived directly from the qualitative model shown in Table 1. It is assumed that complete qualitative data can be obtained, and part of the qualitative data are shown in Table 2. In this table, each row stands for a possible qualitative state. For each species, the magnitude and rate of change are assigned to qualitative values, and these values are taken from the signs quantity space, which has three values: *pos* (positive), *zer* (zero) and *neg* (negative). For instance, the value <*pos*, *zer*> means the magnitude is positive and the rate of change is zero.

Table 1. The Qualitative Model for the MG Detoxification Pathway

Aux1=MG*GSH
Aux2=M^+(Aux1)
Aux3=M^+(HTA)
Aux4=Aux2-Aux3
Aux5=M^+(SLG)
Aux6=M^+(HTA)
dMG/dt=M^-(Aux4)
dGSH/dt=Aux5 - Aux4
dHTA/dt=Aux4 - Aux6
dSLG/dt=Aux6 - Aux5

Table 2. Part of Qualitative Data Provided

MG	GSH	HTA	SLG
<pos,pos>	<pos,pos>	<pos,neg>	<pos,neg>
<pos,zer>	<pos,pos>	<pos,neg>	<pos,neg>
...	...	...	...

4 A General Learning Strategy

Starting from the qualitative data and some reasonable assumptions, the QML system is expected to infer the qualitative model shown in Table 1. The general learning strategy is composed of the following four phases:

1.*Search Space Generation*: First all possible reactions are generated, then these reactions are partitioned to several subsets, each of which contains all enzymatic or non-enzymatic reactions which have the same substrates. All these subsets constitute the search space S. This can be denoted as:

$$S = \{S_1, S_2, ...S_N\}, \tag{4}$$

where S_i (i=1 to N) are the above mentioned subsets.

2. *Pathway Extraction*: A systematic or randomized search algorithm can be employed to find possible pathways from the search space S. A possible pathway must satisfy the following four constraints: (a) *Completeness*: All species are included. (b) *Unique Products*: Every combination of substrates has unique products. This means a possible pathway will include at most one reaction for each subset of S. (c) *Consistency*: There are no conflicting reactions in the pathway. (d) *Additional Domain-Specific Constraints*: Additional constraints may be introduced to reduce the search space, but they are only used in some experiments. These constraints include: one substrate is only catalysed by one enzyme, the number (or maximum number) of enzymatic or non-enzymatic reactions is given. These additional constraints heavily depend on the hypotheses being made by modellers and biologists.

3. *Qualitative Model Conversion*: all possible pathways are converted to qualitative models by the converting algorithm described in Section 2.

4. *Model Verification*: The generated qualitative models are simulated by a qualitative simulator. The simulation results are compared with given qualitative data and the consistent models will be recorded. In this work, JMorven [14] is selected as the qualitative simulator.

Note the last two phases can also be executed in the process of *pathway extraction*, that is, when a possible pathway is extracted, it will be immediately converted to a qualitative model and the converted qualitative model is verified by the qualitative simulator.

5 Clonal Selection Algorithm

The general learning strategy proposed in Section 4 told us that in the *pathway extraction* phase, the search algorithm must be properly selected to attain the best performance.

The search space of the MG detoxification pathway at the current stage is relatively small because only three reactions and four species are involved. An exhaustive search is enough for listing all possible pathways. However, as mentioned in Section 1.2, the latest biological research suggests that this pathway may be more complicated, which means the search space may be very large, and exhaustive search could be intractable.

To make the proposed qualitative model learning system scalable to future more complicated pathways, a modified clonal selection algorithm [15] (CSA) is proposed, which is based on our previous work [16]. The motivation for employing CSA as the search strategy is that it is a well established immune inspired algorithm, and its success in solving the problems presented in this paper will demonstrate that the immune inspired algorithms can be applied to the field of QML for solving real-world applications. In addition, compared with other search algorithms, CSA is particularly suitable for searching highly multi-modal search spaces, which are often the cases for learning future more complicated pathways. This is because given incomplete data there may exist many possible pathways which can cover the data and satisfy the four constraints proposed in the pathway extraction phase of the general learning strategy (see Section 4).

Similar to our previous work [16], the clonal selection algorithm performs a randomized search in the partitioned search space $\boldsymbol{S}$ and may improve the scalability of the QML system when searching a large-scale search space. The basic steps of this modified CSA are as follows:

1. **Antibody Encoding:** Similar to the previous work [16], an integer encoding strategy is employed. As the search space $\boldsymbol{S}$ is composed of several subsets, each of which contains the reactions with the same substrates, the antibody is made up of several slots, and each slot corresponds to a subset in $\boldsymbol{S}$. The value of each slot, ranging from 0 to N (N is the number of reactions in the corresponding subset), stands for the selection of a reaction from the corresponding subset except for the value "0", which means no reaction in the subset is selected. An example is given as follows: In the search space $\boldsymbol{S}$, suppose the first

Subset 1	Subset 2	....
1. MG+GSH<-->HTA+SLG	1. MG+HTA<-->GSH+SLG	...
2. MG+GSH<-->HTA	2. MG+HTA<-->GSH	...
3. MG+GSH<-->SLG	3. MG+HTA<-->SLG	...

Fig. 2. The First Two Subsets in the Search Space

two subsets are shown in Figure 2. An antibody <1,3 .....> will stand for the following selections:

MG+GSH<-->HTA+SLG
MG+HTA<-->SLG
.....

2. **Antibody Initialization:** Given the number of antibodies ***NumPop***, randomly generate all antibodies, and a population (or a antibody repertoire) **P** is constructed.

3. **Selection:** All the antibodies in **P** will be selected.

4. **Clonal Expansion:** All the antibodies will be cloned with the same copies. A temporary population ***TempP*** is constructed.

5. **Hyper-mutation:** The temporary population ***TempP*** will undergo the hyper-mutation scheme. For each slot of the antibody, its value will be replaced by a randomly generated integer with the given probability. The range of this randomly generated integer is from 0 to N, and N is the number of the reactions in the corresponding subset. The hyper-mutation will apply on each antibody in ***TempP*** with a probability ***MuProb***.

Furthermore, if there exist additional domain-specific constraints being made by the modellers as described in Section 4, the hyper-mutation operator will be modified correspondingly. For instance, if the maximum number of non-enzymatic reactions is given, say, *Num*, the hyper-mutation operator will guarantee that at most *Num* non-enzymatic reactions can be selected from different slots, and for the rest slots containing non-enzymatic reactions, the value "0" will be assigned, which means no selection. By doing this modification, the pathway obtained from a mutated temporary antibody still satisfies the additional constraints.

6. **Affinity Evaluation:** There are three criteria for the affinity evaluation: (1)*Completeness:* Whether the pathway represented by the antibody includes all the species. (2)*Consistency:* Whether there are conflicting reactions in the pathway. (3)*Coverage:* Whether the qualitative model converted from the pathway covers all the given qualitative data.

A scoring system shown in Figure 3 is set up based on the above three criteria, so that the better an antibody satisfies these three criteria, the bigger affinity value it will have. For instance, an antibody satisfying both *completeness* and *consistency* constraints will have a bigger affinity value than the one satisfying only the *completeness* constraint. If both two antibodies do not satisfy the *completeness* criterion, the one which includes more species will have a bigger affinity value.

```
If (CompleteRate<1)
      F=w1* CompleteRate
else // CompleteRate==1
       if(ConflictRate<0)
           F=w1+w2*(1-ConflictRate)
       else //ConflictRate==0
           if(CoverageRate<1)
               F=W1+W2+W3*CoverageRate
           else //CoverageRate==1
               F=w1+w2+w3
```

Fig. 3. The Affinity Evaluation of a Possible Pathway

In this scoring system, F is the final affinity value. w1, w2, and w3 are weights, and in the experiments they are all set to 10. In order to calculate the value of F, the following three variables are defined:

a. *CompleteRate:* This is defined as the number of species included in the candidate pathway divided by the number of given species.

b. *ConflictRate:* This is defined as the number of conflicting reactions divided by the total number of reactions in the candidate pathway.

c. *CoverageRate:* This is defined as the coverage rate of the corresponding qualitative model converted from this pathway.

It is pointed out that in the above scoring system, as the coverage test involves computationally expensive qualitative simulation, only the antibody which satisfies the other two constraints can be tested for coverage.

7. **Re-selection:** The re-selection will be based on the above affinity scoring system. ***numPop*** best antibodies in ***TempPop*** will be selected to constitute the new antibody repertoire.

Steps 3∼7 are executed iteratively, until the termination conditions are satisfied.

6 Experimental Design and Results

There are two tasks for the experiments corresponding to the two goals described in Section 1.3: First the validity of the proposed QML system has to be examined; second, the performance of the proposed CSA for searching all possible pathways will be tested. In this paper, all the experiments were performed on a computer cluster with 8 compute nodes (each node has two Opteron 850 (2.4GHz) CPUs and 4GB RAM).

6.1 The Validity of the QML System

Two experiments were performed to test the validity of the QML system. In the first experiment, the following assumption was made:

Assumption A: In the pathway, one substrate can only be catalysed by one enzyme.

This assumption results in a smaller search space because this means two enzymatic reactions A –> B and A–>C will be conflicting if they appear in the same pathway.

Exhaustive search was used to extract all possible pathways, and 4,110 possible pathways were obtained. Then these possible pathways were converted to qualitative models and tested for coverage. After 18 hours calculation, most of which was consumed by the qualitative simulation, 8 consistent pathways were obtained. The real pathway was among them, and it was also the simplest one.

In the second experiment, *Assumption A* was removed. The same exhaustive search was employed, and after more than 15 days calculation, 13 consistent pathways were obtained and eight of them were the same as those obtained in the first experiment. Again most of the computation time was consumed by qualitatively simulating the 18,772 possible pathways.

6.2 The Performance of CSA

To test the performance of the proposed clonal selection algorithm, another two "artificial pathways" were created: MG-Ex1 and MG-Ex2. Besides the four species in the original MG detoxification pathway, two additional species were introduced to the MG-Ex1 pathway. Correspondingly two additional enzymatic reactions and two enzymes which catalysed these two reactions were also introduced. It was similar for the construction of the MG-Ex2 pathway, which had seven additional species and enzymes.

Although these two artificial pathways do not exist in the real world, they can simulate the fact that the real MG detoxification pathway could be more complicated. So they were used to test the scalability of the proposed CSA.

In order to make comparisons, exhaustive search was used as a baseline in all experiments. In addition, two deterministic algorithms, Backtracking (BK) and Backtracking with Forward Checking (BKFC) were also developed to learn the pathways, which was similar to our previous work [17]. Thus all four algorithms were used to learn all three pathways.

In all experiments, the qualitative simulation process was not executed, and consequently the task of all the four search algorithms became simply searching all possible pathways which satisfy all constraints except the *coverage* constraint. There are two reasons for this: (1) Qualitative simulation is computationally expensive. This can be seen from the two experiments presented in Section 6.1. For the MG-Ex1 and EG-Ex2 pathways, the execution time would be intolerable if qualitative simulation were performed. The long simulation time is a limitation of the QML system, which is expected to be overcome in the future. (2) On the other hand, by ignoring the qualitative simulation, we can focus on comparing the performances of different search algorithms for searching all possible pathways.

For all experiments on the MG-Ex1 and MG-Ex2 pathways, it was assumed that the maximum number of non-enzymatic reactions was 2. The parameter

settings for CSA were: the clonal size was 10, the hyper-mutation probability was 0.9 for all experiments. The population size was 100 for experiments on the MG pathway, and 1,000 for experiments on the MG-Ex1 and MG-Ex2 pathways.

The values of the parameters in CSA were determined by the characteristics of the search space: It was found that the search spaces of all experiments were highly multi-modal, so a high hyper-mutation probability would help the antibodies escape from the local optima more easily. The clonal size and population size were determined by the size of the search space.

The experimental result is shown in Table 3. In this table, each experiment ID in the first column indicates the pathway in which this experiment is involved and the execution order of this experiment. For instance, "MG-2" stands for the second experiment on the MG pathway, while "MG-Ex1-1" stands for the first experiment on the MG-Ex1 pathway. The number of enzymes involved in the pathway is given in the third column, while the symbol "N/A" means that the number is unknown. The running time of each experiment is given in the last column, and for all the experiments using CSA, the results are the mean values based on 10 trials and the standard deviations are also calculated and enclosed in brackets. The details of the ten trials for each experiment are shown in Figure 4. In this figure, the unit of time is *millisecond* for the first three experiments and *second* for the last two experiments.

Table 3. Experimental Result

Experiment ID	Algorithm	Num. of Enzymes	Assumption A	Search Space	Running Time (MilliSec)
MG-1	Exhaustive	N/A	True	16,384	2,000
MG-2	BK	N/A	True	16,384	156
MG-3	BKFC	N/A	True	16,384	141
MG-4	CSA	N/A	True	16,384	4,304 (2,018)
MG-5	Exhaustive	N/A	False	262,144	23,547
MG-6	BK	N/A	False	262,144	734
MG-7	BKFC	N/A	False	262,144	704
MG-8	CSA	N/A	False	262,144	61,030 (60,276)
MG-Ex1-1	Exhaustive	4	True	7.5*E9	3,253,123
MG-Ex1-2	BK	4	True	7.5*E9	342,087
MG-Ex1-3	BKFC	4	True	7.5*E9	333,627
MG-Ex1-4	CSA	4	True	7.5*E9	615,000 (831,135)
MG-Ex2-1	Exhaustive	9	True	1.2*E28	563,125(Sec)
MG-Ex2-2	BK	9	True	1.2*E28	62,748(Sec)
MG-Ex2-3	BKFC	9	True	1.2*E28	57,045(Sec)
MG-Ex2-4	CSA	9	True	1.2*E28	28,781(32,059)(Sec)
MG-Ex2-5	Exhaustive	9	False	5.5*E49	4,546,359(Sec)
MG-Ex2-6	BK	9	False	5.5*E49	774,370(Sec)
MG-Ex2-7	BKFC	9	False	5.5*E49	478,894(Sec)
MG-Ex2-8	CSA	9	False	5.5*E49	136,715(159,736)(Sec)

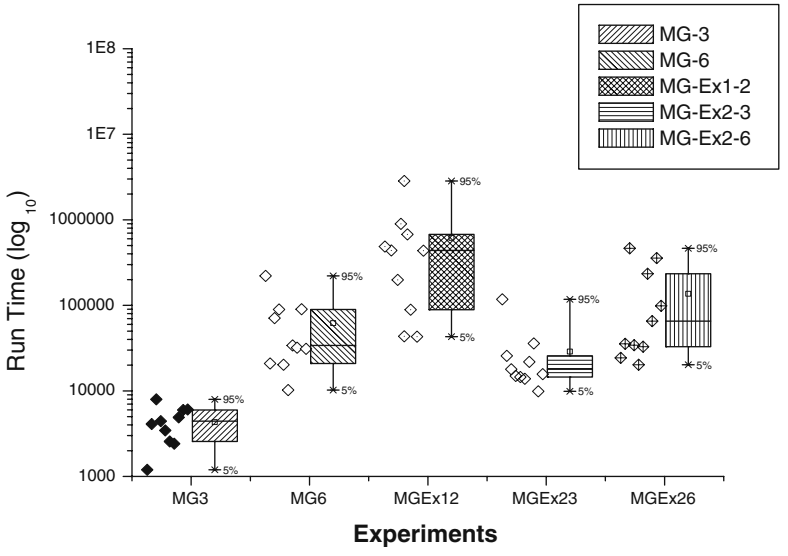


Fig. 4. The Ten Trials of CSA for Each Experiment

For the MG and MG-Ex1 pathways, the algorithms would not terminate until all possible pathways were found. For the MG-Ex2 pathway, because the search space is extremely large, without additional heuristics, the number of possible candidates is very big. So the termination condition of all algorithms on MG-Ex2 is to find 100,000 candidates for Experiments MG-Ex2-1∼4 and 50,000 for Experiments MG-Ex2-5∼8.

The experimental result told us that for searching these pathways with the increase of the size of the search space, CSA performed better than other two deterministic algorithms, which demonstrated its scalability. The success of CSA is because it is particularly suitable for searching a large and highly multi-modal landscape. The randomization of CSA enables it to explore the search space more efficiently and escape from the local optima easily, while the clonal expansion and hyper-mutation operators combined together make the antibodies efficiently locate the local optima. On the other hand, BK and BKFC are not very efficient for searching these large-sized search spaces because they always explore the search space in a deterministic way.

7 Conclusions, Discussions, and Future Work

In this paper, a QML system is presented that was used to qualitatively identify the MG detoxification pathway. The aim of this work is to help both biological experimentalists and modellers better understand this pathway at a qualitative level, provide them with guidance and suggestions. The validity of this system

is verified, and more importantly, the scalability of this system is improved by employing CSA, an efficient algorithm inspired by the immune system.

It is pointed out that there exist more efficient immune-inspired search algorithms for searching the multi-modal search space, such as opt-IA [18] and opt-AiNET [19]. As the first step we started with the classical CSA algorithm and demonstrated that the immune-inspired search strategy can be applied to solve the problem proposed in this paper. In the near future, based on the positive research results presented in this paper, other immune-inspired algorithms will be investigated on the same problem and their performances will be compared and analysed.

There is also other work that needs to be done in the future: (1) Learning with incomplete qualitative data. (2) Learning with hidden species: if there are hidden species not measured in the pathway, learning task will be more challenging. (3) Under the above two conditions, it is expected to investigate the suitability and performance of different immune-inspired approaches.

Acknowledgements

The authors would like to thank Dr Alessandro P.S. de Moura, Camilla de Almeida and Prof. Celso Grebogi for their suggestions about the modelling issues. We also thank Prof. Ian Booth and his research group for their explanations of the Detoxification Pathway of Methylglyoxal. WP and GMC are supported by the CRISP project (*Combinatorial Responses In Stress Pathways*) funded by the BBSRC (BB/F00513X/1) under the Systems Approaches to Biological Research (SABR) Initiative. WP is also supported by the National Natural Science Foundation of China under grant No. 60433020, 60673099, 60773095.

References

1. Forbus, K.D.: Qualitative reasoning. In: The Computer Science and Engineering Handbook, pp. 715–733 (1997)
2. Kuipers, B.: Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge. MIT Press, Cambridge (1994)
3. Shen, Q., Leitch, R.: Fuzzy qualitative simulation. IEEE Transactions on Systems, Man, and Cybernetics 23(4), 1038–1061 (1993)
4. Ljung, L.: System Identification – Theory For the User, 2nd edn. Prentice Hall, Upper Saddle River (1999)
5. Richards, B.L., Kraan, I., Kuipers, B.: Automatic abduction of qualitative models. In: National Conference on Artificial Intelligence, pp. 723–728 (1992)
6. Say, A.C.C., Kuru, S.: Qualitative system identification: deriving structure from behavior. Artificial Intelligence 83, 75–141 (1996)
7. Coghill, G.M., Srinivasan, A., King, R.D.: Qualitative system identification from imperfect data. Journal of Artificial Intelligence Research 32, 825–877 (2008)
8. Cooper, R.: Metabolism of methylglyoxal in microorganisms. Annual Review of Microbiology 38, 49–68 (1984)

9. Ferguson, G.P., Totemeyer, S., MacLean, M.J., Booth, I.R.: Methylglyoxal production in bacteria: suicide or survival? Archives of Microbiology 170(4), 209–218 (1998)
10. Michaelis, L., Menten, M.: Die kinetik der invertinwirkung. Biochem Z(49), 333–369 (1913)
11. de Moura, A., de Almeida, C., Grebogi, C.: Personal communication (February 2008)
12. de Almeida, C.: Modelling of methylglyoxal detoxification pathway in enteric bacteria. In: Abstract book of the 9th International Conference on Systems Biology, Gŏteborg, p. 170 (2008)
13. MacLean, M.J., Ness, L.S., Ferguson, G.P., Booth, I.R.: The role of glyoxalase i in the detoxification of methylglyoxal and in the activation of the kefb k^+ efflux system in escherichia coli. Molecular Microbiology 27(3), 563–571 (1998)
14. Bruce, A.M., Coghill, G.M.: Parallel fuzzy qualitative reasoning. In: Proceedings of the 19th International Workshop on Qualitative Reasoning, Graz, Austria, pp. 110–116 (2005)
15. de Castro, L.N., Von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation, Special Issue on Artificial Immune Systems 6, 239–251 (2002)
16. Pang, W., Coghill, G.M.: Modified clonal selection algorithm for learning qualitative compartmental models of metabolic systems. In: Thierens, D. (ed.) Genetic and Evolutionary Computation Conference (GECCO 2007), pp. 2887–2894. ACM Press, New York (2007)
17. Pang, W., Coghill, G.M.: Advanced experiments for learning qualitative compartment models. In: The 21st International Workshop on Qualitative Reasoning, Aberystwyth, UK, June 2007, pp. 109–117 (2007)
18. Cutello, V., Narzisi, G., Nicosia, G., Pavone, M.: An immunological algorithm for global numerical optimization. In: Talbi, E.-G., Liardet, P., Collet, P., Lutton, E., Schoenauer, M. (eds.) EA 2005. LNCS, vol. 3871, pp. 284–295. Springer, Heidelberg (2005)
19. de Castro, L.N., Timmis, J.: An artificial immune network for multimodal function optimization. In: Proceedings of IEEE Congress on Evolutionary Computation (CEC 2002), pp. 674–699. IEEE Press, Los Alamitos (2002)

Application of AIS Based Classification Algorithms to Detect Overloaded Areas in Power System Networks

N.C. Woolley and J.V. Milanović

School of Electrical and Electronic Engineering
The University of Manchester, Manchester, United Kingdom

Abstract. The identification of voltage collapse prone areas in a power system network is often a difficult and computationally intensive task. Artificial Immune System (AIS) algorithms have been shown to be capable of generalization and learning to identify previously unseen patterns. In this paper, an AIS algorithm - Support Vector Machine (AIS-SVM) hybrid algorithm is developed to identify voltage collapse prone areas and overloaded lines in the power system network. The applicability of AIS for this particular task is demonstrated on a 30 bus electrical power system network and its accuracy compared to a conventional un-optimised SVM algorithm across 3 different power system networks.

1 Introduction

Due to regulatory pressure power system networks around the world increasingly operate closer to their stability limits as operators try to deliver power to end users in the most economical way. In such a situation, unforeseen problems may arise and endanger voltage stability and loadability limits in the network.

The proximity of the network to voltage collapse and transmission line loadability levels are crucial parameters for power system operators in order to avoid blackouts in the system.

A standard approach to establish the proximity to voltage collapse in a power system is to monitor the system power flow Jacobian matrix [1]. This matrix becomes singular at the maximum loading point of the bus being monitored. The singularity of the Jacobian matrix causes problems when computing the voltage stability limit of a power system as the standard Newton-Raphson method used to solve the power flow problem diverges. Several methods have been discussed to establish the proximity of individual bus voltage to voltage collapse, including regulating the step length on the Newton- Raphson iterations [2], performing sensitivity analysis on the converged Jacobian matrix [1] or using the homotopy continuation [3] method to trace the power voltage relationship at each bus. Overloaded lines on the other hand, can be easily identified by using a standard power flow algorithm.

Calculating the proximity to voltage collapse for every possible loading condition in a power system can be a computationally intensive task. Artificial Immune Systems (AIS) and other artificial intelligence (AI) techniques can assist this task by helping to develop an approximate model that monitors a power network's susceptibility to these problems. Research in both [4] and [5] uses a Kohonen self organizing map (SOM) to identify weak voltage buses within the power system network. The

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 165–177, 2009.

results in [5] were compared against those obtained from singular value decomposition of the power flow Jacobian. In [6] weak voltage buses were grouped together using a fuzzy clustering algorithm.

In a similar way to other AI techniques, voltage weak areas and overloaded lines can be grouped together using an AIS clustering algorithm. AIS algorithms have been used to solve data clustering problems in the past [7, 8]. It was demonstrated in [8] that an AIS algorithm yields the best results among several considered algorithms.

AIS algorithms on their own [9] or combined with other techniques, such as Support Vector Machines (SVM) in [10] and [11], have also been used for supervised learning and classification. In [10] an AIS algorithm was used to optimize the parameters of an SVM to detect failures in induction motors. AIS optimisation algorithms have been shown to converge to a global optimum[12] (given enough time) and perform fewer evaluations to reach this optimum than hybrid genetic algorithms[13].This paper describes a novel AIS algorithm based method, to identify voltage collapse prone buses and overloaded lines in the power system network. The algorithm uses clustering, classification and optimization to arrive at reliable prediction of voltage weak areas.

2 AIS Optimized SVM Algorithm

2.1 Generating Training and Testing Data

Voltage weak buses and overloaded lines can be detected by generating a wide variety of training data from simulations[1][14]. In this study, a 30 bus test network[15] was used to generate valid operating conditions on the power system network.

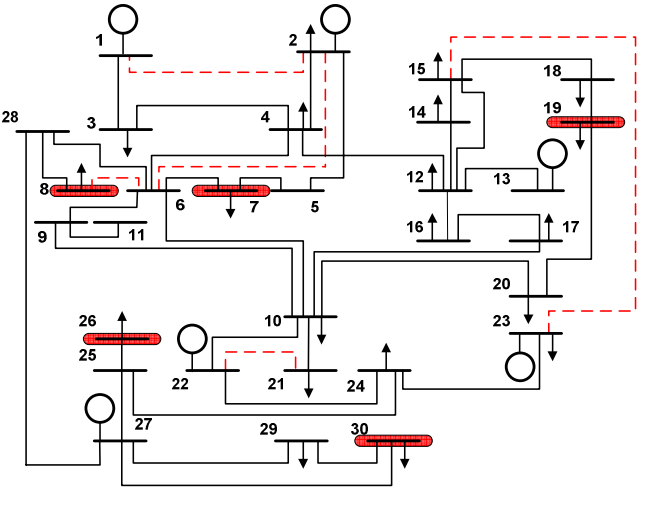


Fig. 1. 30 bus test network(adapted from [15]). The identified 5 weakest voltage buses are enclosed in shaded rounded rectangles and the 5 most commonly overloaded lines are highlighted using thick red dashed lines.

[1] The simulated results were generated using MATPOWER.

Sample data was generated by randomly raising the real and reactive power level across the whole network by up to 100% above the base load. Each load in the network was also randomly varied by up to ±20% of the original base load at that particular bus. This is mathematically shown in equation (1).

$$\boldsymbol{l} = \boldsymbol{l_b} + \boldsymbol{p l_b} + \boldsymbol{l_b}.\boldsymbol{y} \tag{1}$$

Where:

$\boldsymbol{l}$	System load at given operating conditions
$\boldsymbol{l_b}$	Base system load
p	Uniformly distributed random variable between 0 and 1
$\boldsymbol{y}$	Row vector of uniformly distributed random numbers between -0.2 and 0.2
.	Element by element multiplication of two vectors

2500 operating points are generated for this study.

2.2 Homotopy Continuation Power Flow

For each of the operating points, a static stability study was carried out on the system to determine each bus' proximity to voltage collapse. This was achieved using homotopy continuation load flow [3, 16].

A homotopy parameter λ is added to the standard Newton- Raphson load flow to represent load growth.

$$\boldsymbol{L_s(\lambda)} = \boldsymbol{L_{s0}} + \boldsymbol{\lambda L_d} \tag{2}$$

Where:

$L_s(\lambda)$	Load / generation at some increase in lambda
L_{s0}	Load / generation at current operating point, $\lambda = 0$
L_d	Loading and generation that will lead to voltage collapse
λ	Homotopy parameter or the load increase factor (loadability)

These equations can be combined with the standard load flow equations to form a homotopy function.

$$\boldsymbol{F(x, \lambda)} = \boldsymbol{L(x)} - \boldsymbol{L_s(\lambda)} \tag{3}$$

Where:

$\boldsymbol{x}$	Vector of voltage magnitudes and angles
$L(\boldsymbol{x})$	Standard load flow equation for voltage magnitudes and angles $\boldsymbol{x}$
$F(\boldsymbol{x}, \lambda)$	Homotopy function

The homotopy equation can be shown to be equivalent to solving a differential equation[3, 16] in the following form.

$$\boldsymbol{F_x(x,\lambda)}\frac{\boldsymbol{dx}}{\boldsymbol{d\lambda}} = -\boldsymbol{F_\lambda(x,\lambda)} \tag{4}$$

Where:

$\boldsymbol{F_x}$ and $\boldsymbol{F_\lambda}$ are the Jacobian's of $F(\boldsymbol{x},\lambda)$ with respect to $\boldsymbol{x}$ and λ.

The singularity of $\boldsymbol{F_x}$ causes problems at the maximum power point as the standard power flow Jacobian ($\boldsymbol{F_x}$) becomes singular. To alleviate this problem, the power flow Jacobian can be augmented with an extra row and column.

$$\boldsymbol{J_{aug}} = \begin{bmatrix} \boldsymbol{F_x} & \boldsymbol{F_\lambda} \\ \boldsymbol{e_k} & \end{bmatrix} \tag{5}$$

Where $\boldsymbol{e_k}$ is a vector of modulus 1 with its kth element equal to unity. By carefully selecting$k$, $\boldsymbol{J_{aug}}$ becomes non singular.

To determine the maximum power point on the power voltage curve the power flow equations are modified. The ith bus in the system is tested by amending the real and reactive power demand at this bus using the following equations:

$$P_{Li}(\lambda) = P_{Li0}[1+\lambda] \tag{6}$$

$$Q_{Li}(\lambda) = P_{Li0}\tan(\emptyset_i)\,[1+\lambda] \tag{7}$$

Where:

P_{Li0}, Q_{Li0}	Original active and reactive loads at bus i
$\emptyset_i$	Power factor angle at bus i

The elements described above can be combined together to form a continuation power flow algorithm that will estimate the loadability (λ) of each bus in the power network. The iterations of the continuation algorithm use a predictor corrector process.

Firstly, a tangent vector ($\boldsymbol{T}$) is calculated by solving the following equation.

$$\begin{bmatrix} \boldsymbol{F_x} & \boldsymbol{F_\lambda} \\ \boldsymbol{e_k} & \end{bmatrix} \boldsymbol{T} = \boldsymbol{e_k}^{\boldsymbol{T}} \tag{8}$$

A predicted solution is estimated using the computed tangent vector and a stepping size σ.

$$\begin{bmatrix} \boldsymbol{x'} \\ \boldsymbol{\lambda'} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\lambda} \end{bmatrix} + \boldsymbol{\sigma T} \tag{9}$$

Where $\boldsymbol{x'}$ and λ' represent predicted solutions. These solutions are corrected using a modified Newton- Raphson power flow method where the equations have one additional state variable, namely λ.

During the algorithm, the step length σ is rigorously controlled. It is allowed to grow at a rate governed by equation 10, where i is the number of iterations taken by the last correction of the Newton- Raphson method. If divergence of the Newton-Raphson method is detected, then the step length is reduced using equation 11.

$$\sigma = 6\sigma/i \tag{10}$$

$$\sigma = \sigma/3 \tag{11}$$

The continuation parameter $\boldsymbol{e_k}$ is also closely monitored. It is changed to the parameter that yields the largest tangent vector ($\boldsymbol{T}$) component. If ill-conditioning of the augmented Jacobian matrix is detected, the continuation parameter is also changed.

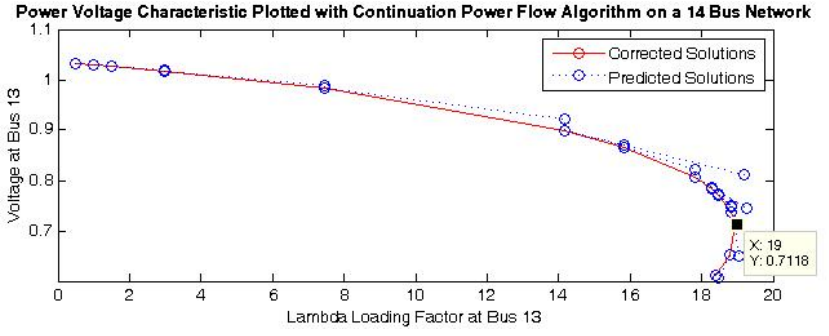


Fig. 2. A trace of a power voltage characteristic curve with the maximum value of lambda highlighted on the graph. This indicates the loadability of that bus.

Once each bus' loadability limit (λ) has been calculated for a given operating point, each bus in the network can be ranked. The bus with the lowest value of λ is ranked as the weakest bus.

2.3 Overloaded Lines

Every operating point that is generated in the study is inspected to find any lines in the network that are overloaded. Overloaded lines are defined as those lines where the apparent power (MVA) rating of the line has been exceeded.

2.4 Clustering Using AIS

Operating points with similar voltage stability conditions and overloaded line configurations were grouped together using an unsupervised artificial immune clustering algorithm (UAIC)[8].

UAIC is modelled on the clonal selection principle in which B cells undergo rapid clonal expansion to target invading antigens. Feature vectors are modelled as antigens. Antibodies and memory cells represent the prospective and evolved cluster centres respectively. The following table describes the mapping between the components in the AIS model and their power system equivalents.

Table 1. AIS and power system mapping

AIS	Power System
Antigen	Feature vector of bus ranks or overloaded lines
Antibody	Prospective cluster centre
Memory	Evolved cluster centres

2.4.1 Clustering Weak Voltage Buses

A feature vector that contains the rank of each bus is used as the input for the clustering algorithm.

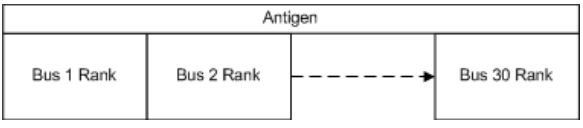


Fig. 3. Antigen structure for clustering weak voltage buses

The similarity between operating points is determined by applying a distance measure developed to compare similarly ranked operating points. The distance measure defines the number of upwards or downwards movements that each bus would have to make to map one operating point's rank to another's. For example, the vector $\boldsymbol{a} = [1\ 2\ 3\ 4]$ of a hypothetical 4 bus system is a ranking distance of 6 away from $\boldsymbol{b} = [4\ 1\ 2\ 3]$. Bus 4 moves down 3, bus 1, 2 & 3 all move down 1, giving a total distance of 6.

2.4.2 Clustering Overloaded Lines

The configuration of the network's overloaded lines can be represented as an antigen for presentation into the UAIC clustering algorithm.

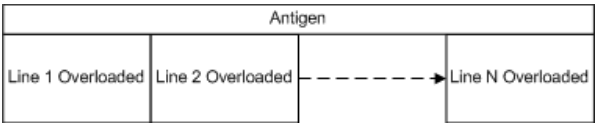


Fig. 4. Antigen for overloaded line clustering

Similarity between operating points was determined using the Manhattan distance measure, defined as[17]:

$$distance = Manhattan(x, y) = \frac{1}{N}\sum_{i=1}^{N}(|x_i - y_i|) \tag{12}$$

Where:

$\boldsymbol{x}$ and $\boldsymbol{y}$ are points in $\mathbb{R}^N$

2.4.3 Unsupervised Artificial Immune Classification Algorithm (UAIC)

The UAIC algorithm starts by generating an initial memory cell population. This initialization takes place using the Kaufman initialization procedure[18]. This procedure initializes a set of memory cells around the most centrally located antigen. Other memory cells of different classes are initialized around the centrally located cell.

To compare the similarity between antigens and antibodies, a distance function is defined. Two different distance functions are used for clustering weak voltage buses and clustering overloaded line configurations.

The training antigens are presented to the clustering algorithm in a random order. The following steps are performed on each training antigen.

1) The antigen is assigned to the class of the nearest memory cell based on a calculation of affinity with all other memory cells

$$mc_{match} = arg \max_{mc \epsilon MC} affinity(ag, mc) \tag{13}$$

Where:

$mc \in MC$ is a memory cell from the population of memory cells MC

ag is an antigen (weak bus rank or overloaded line configuration)

$affinity(ag_i, ab)$ is $1 - distance(ag_{i,} ab)$ between the ith antigen and the selected antibody

2) The antibody population of the antigen's class is evolved using the procedures of hyper-mutation and clonal selection. A number of high affinity antibodies are selected to be cloned. The number of clones generated depends on the clonal rate and the affinity between the antigen and the antibody.

$$C = \sum_{i=1}^{N} round(Cr * affinity(ag_i, ab)) \tag{14}$$

Where:

C is number of clones generated

Cr is the clonal rate defined for the algorithm (in this study, it is 10)

$ab \in AB$ is an antibody from the population of antibodies AB

3) The clones are then subjected to affinity maturation. Higher affinity antibodies are mutated less than those with a lower affinity.
4) Finally, the matured set of antibodies is evaluated against the original best classifying memory cell to decide if the new memory cell should be accepted for inclusion into the memory pool. The best new memory cell is found by searching the pool of antibodies:

$$mc_{candidate} = \arg\max_{\mathrm{ab}\epsilon \mathrm{AB}} affinity(\mathrm{ag}, \mathrm{ab}) \tag{15}$$

If $mc_{candidate}$ is a better match (has a higher affinity) to the antigen than mc_{match} then the candidate is further considered for inclusion as part of the memory cell population. $mc_{candidate}$ is only added to the memory cell population if the affinity of the candidate memory cell is greater than mc_{match} and if $(affinity\ threshold *$

$affinity\ threshold\ scalar$) is less than the affinity between $mc_{candidate}$ and the training antigen.

The $affinity\ threshold$ is calculated as the average affinity between all of the training antigens. The $affinity\ threshold\ scalar$ is set to 0.1 in this study.

After training on all the antigens is complete, the algorithm will restart and begin training on the first antigen. The algorithm terminates when the number of antigen classes changed between iterations reaches a minimum threshold value (5% in this study).

2.5 Classification Using AIS Optimized SVM

Support vector machines (SVMs) are a classification technique based on statistical analysis. They aim to separate an n-dimensional continuous feature space into two classes. SVMs represent this problem by formulating a linear separating hyperplane. This hyperplane can then be represented as shown below.

$$f(x) = w^T x + b \tag{16}$$

Where $\boldsymbol{w}$ is an n-dimensional vector of weights, $\boldsymbol{b}$ is a bias vector and $\boldsymbol{x}$ is a point in n-dimensional space. Depending on whether a point lies above or below this plane determines to which class the point belongs.

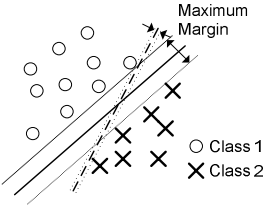


Fig. 5. Two possible linear discriminants to separate the circles from the crosses

Figure 6 shows the construction of two hyperplanes that separate the linearly separable classes: class 1 and class 2. The aim of the SVM is to construct a maximum margin hyperplane that separates the two classes. The construction of the maximum margin hyperplane can be formulated as an optimization problem.

The SVM can be easily adapted to work for non-linear classification problems. In this instance, the training instances are mapped into a higher dimensional space and a maximum margin linear hyperplane constructed to classify the data. If $\theta(\boldsymbol{x})$ defines a mapping such that $\theta(\boldsymbol{x})$: $\mathbb{R}^n \rightarrow \mathbb{R}^{n'}$ where $n' \gg n$ the support vector machine dual optimization[19] becomes:

$$\begin{cases} \min\limits_{\alpha} \left[\frac{1}{2}\sum_{j=1}^{k}\sum_{i=1}^{k} \propto_i \propto_j y_i y_j \theta(x_i)\theta(x_j) - \sum_{i=1}^{k} \propto_i\right] \\ s.t \ \sum_{i=1}^{k} y_i \propto_i = 0 \\ \propto_i \geq 0 \ for \ i = 1 \ldots k \end{cases} \tag{17}$$

Where:

$\propto_i, \propto_j$ Lagrange multipliers

$\boldsymbol{x} \in \mathbb{R}^k$

The inner product between certain sets of two functions can be calculated without ever knowing the individual mapping. This leads to a replacement of $\theta(x_i)\theta(x_j)$ with $K(x_i, x_j)$ where $K(x_i, x_j)$ is a kernel function. When the data is not linearly separable, then a penalization parameter C is introduced. The parameter C represents the trade-off between minimizing the training set error and maximizing the margin.

There are several choices of kernel function available. The classification method used in this study is the radial basis function.

$$\boldsymbol{K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)} \tag{18}$$

The parameter σ affects the above kernel function and also influences the ability of the support vector machine to accurately classify data.

The choice of values for C and σ will affect the classification performance of the classifier. To obtain a high level of classification accuracy, C and σ should be chosen carefully and optimised where possible. This can be carried out using an optimisation algorithm. The other main variable in the SVM is the kernel function. In this study, the kernel function was left constant as the radial basis function.

Support vector machines are often used for two class classification problems. However, the method can easily be extended to multi class problems by considering a multi-class problem as a set of two class classification problems.

2.5.1 Optimizing *C* and σ with an AIS Optimization Algorithm

Artificial immune systems have been successfully applied to tackle optimization problems in other areas[20, 21]. In this study, the parameters $C$and σ of the SVM were optimized using Opt-aiNET. An AIS algorithm was chosen over a genetic optimisation algorithm as because of its ability to avoid local optima[10], converge to a global optimum[12] and require fewer evaluations to reach the optimum solution[13].

Opt-aiNET works by evolving a population of antibodies which are candidate solutions to the function being optimized. The antibodies are continually selected against by comparing their fitness with the objective function. The antibodies also undergo a process of clonal expansion, mutation and interaction, where the best performing antibodies are cloned the most and mutated the least. Over time, a memory set of cells evolves that represents the best candidate solutions to the optimization function. The optimization procedure is terminated once the number of iterations no longer improves the performance of the classifier.

The classification performance of the SVM was optimized by running the classification algorithm using 10 fold cross validation whilst varying the parameters C and σ. The data set used to carry out the optimization contained 200 feature vectors with 50 vectors from each class. A reduced set was used to ensure that building the SVM model was not prohibitively time consuming.

The classification performance of the SVM with no parameter optimization was compared with that obtained with parameter optimization.

The optimization algorithm used was programmed using the optimization algorithm toolkit (OAT[22]). The toolkit was modified and integrated to the WEKA machine learning environment[23] to test the classification accuracy of the support vector machine using LibSVM[24].

2.5.2 Classification of Voltage Collapse Prone Buses and Overloaded Lines

The inputs for the classification of voltage collapse prone buses and overloaded lines were a summary of network parameters representing the current load on the network. These parameters used were as follows:

- Real power demand at each bus in the network
- Reactive power demand at each bus in the network
- Real power generation at node in the network
- Reactive power generation at each node in the network
- Total real power losses through the network
- Total reactive power losses through the network

On the 30 bus system, this means a feature vector used for input into the SVM has 40 non-zero attributes for real and reactive power demand, 12 non-zero attributes for real and reactive power generation and 2 attributes for real and reactive total power losses. Each feature vector therefore has 54 attributes, excluding the class label.

3 Case Studies and Results of Simulations

Simulations were carried out on the 30 bus network whose single line diagram is shown in figure 1. The original loading of the network in the base configuration does not cause any of the lines to be overloaded nor any bus voltage to be below statutory limit of 5%.. The optimization of the SVM was tested with a test 9 bus network [25] and a 14 bus network[26]. By varying the load in the network in the range of 100% to 200% of a base loading 2500 network operating points were generated.

The unsupervised artificial immune classification algorithm was then setup to find 4 clusters in the data. These 4 clusters represent the different rankings for the 30 buses in terms of voltage stability issues. In this way, the 2500 operating points are divided into 4 groups depending on the ranking of the voltage weak buses at that particular operating condition. Each group does not necessarily contain rankings that are all identical, there may be some variation. However, each group represents the best division of the operating points that the UAIC algorithm could achieve based using the ranking distance similarity measure described in section 2.4.1.

The 4 classes that were produced by the UAIC algorithm are shown in the following table. The table also shows the top 10 weakest buses in the 30 bus network for each of the 4 classes. The different ranks produced in each of the 4 classes are the most commonly occurring rank within that class. The variation in bus ranking between classes in the top 10 buses shows how the UAIC algorithm partitions the 2500 operating points.

Table 2. Various operating points of the 30 bus power network grouped into clusters depending on the ranking of buses within the network along with the top 10 weakest buses by class

Class	No. in Class	Top 10 Weakest Buses									
		1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th
Class 1	761	30	8	26	7	19	21	14	17	12	15
Class 2	526	8	30	26	7	19	21	17	14	12	24
Class 3	610	30	8	26	7	19	17	21	24	12	14
Class 4	603	30	8	26	7	19	21	14	17	24	12

Clustering the overloaded lines was carried out using the same AIS clustering algorithm. The groupings for the overloaded line classes are shown in the following table.

Table 3. The class groupings for overloaded lines in the 30 bus power system network

Class	No. in Class	No. Of Overloaded Lines	Overloaded Lines
Class 1	1624	2 overloaded lines	Bus 6 to 8 & 21 to 22
Class 2	188	0 or 1 overloaded lines	Bus 6 to 8
Class 3	412	5 or more overloaded lines	Bus 6 to 8, 21 to 22, 1 to 2, 2 to 6, 15 to 23
Class 4	276	3 or 4 overloaded lines	Bus 6 to 8, 21 to 22, 15 to 23 & 1 to 2

The single line diagram in figure 1 shows both the 5 weakest buses and the 5 most overloaded lines that were found from this study. After grouping, the performance of the AIS trained SVM classifier was compared with the standard SVM. The results of this comparison are shown in the following table.

Table 4. Classification accuracy comparison of the 9 bus, 14 bus and 30 bus support vector machine with optimised and un-optimised classification parameters

Test Network	Un-optimized SVM Classification Accuracy	AIS Optimized SVM Classification Accuracy
Overloaded Lines (9 Bus Network)	88.4%	99.4%
Ranking Voltage Weak Buses (9 Bus Network)	62.6%	96.4%
Overloaded Lines (14 Bus Network)	85.2%	99.7%
Ranking Voltage Weak Buses (14 Bus Network)	79.6%	94.7%
Overloaded Lines (30 bus network)	91.6%	97.0%
Ranking Voltage Weak Buses (30 bus network)	42.3%	88.4%

The results show that parameter optimisation significantly improves the default classification accuracy of the SVM. Classification accuracies of between 88.36% and 97% were achieved for the detection of overloaded lines and voltage weak buses. The classification accuracy obtained is suitably high to allow the classifier to be used to detect weak areas in the power system network.

4 Conclusion

In this paper a hybrid Artificial Immune System – Support Vector Machine algorithm was used to identify voltage collapse prone areas and overloaded lines in power system networks. An unsupervised artificial immune classification (UAIC) algorithm was first applied to the power system network to identify groups (clusters) of voltage weak buses (voltage collapse prone areas in the network) and overloaded transmission lines. The clusters were then used as labels for a support vector machine classification algorithm to build a model capable of identifying weak buses and overloaded lines in the network. Two models were built; one using an un-optimised SVM and another using an AIS-SVM hybrid where the AIS algorithm was used to optimise the parameters of the SVM.

The results show that SVM parameter optimization significantly improves the classification accuracy of the classifier. The method discussed is a viable and fast way of identifying voltage weak areas and overloaded lines of a power system network. The methodology described in this study could easily be applied to studies where the computationally intensive nature (due to the size of the power network) of performing a static voltage stability assessment and power flow is prohibitively expensive.

Acknowledgement

This research is funded by Engineering and Physical Sciences Research Council (EPSRC) and Power Network Research Academy (PNRA).

References

[1] Taylor, C.W.: Power System Voltage Stability (1998)
[2] Rousseaux, P., Cutsem, T.V.: Quasi steady-state simulation diagnosis using Newton method with optimal multiplier. IEEE Transactions on Power Systems (2006)
[3] Iba, K., Suzuki, H., Egawa, M., Watanabe, T.: Calculation of Critical Loading Condition with Nose Curve Using Homotopy Continuation Method. IEEE Transactions on Power Systems 6 (1991)
[4] Chauhan, S., Dave, M.P.: Kohonen Neural Network Classifier for Voltage Collapse Margin Estimation. Electric Power Components and Systems 33 (1997)
[5] Wan, H.B., Song, Y.H., Johns, A.T.: Kohonen Neural Network based Approach to Voltage Buses / Areas Identification. IEEE Proceedings on Generation, Transmission and Distribution 144, 340–344 (1997)

[6] An, T., Zhou, S., Yu, J.: Application of Fuzzy Maximal Tree Clustering for Weak Voltage Buses Grouping. In: The Sixth World Congress on Intelligent Control and Automation, 2006. WCICA 2006 (2006)
[7] Castro, L.N.d., Zuben, F.J.V.: An Evolutionary Immune Network for Data Clustering. In: IEEE Brazillian Symposium on Artificial Neural Networks, Rio de Janeiro (2000)
[8] Zhong, Y., Zhang, L., Huang, B., Li, P.: An unsupervised artificial immune classifier for multi/hyperspectral remote sensing imagery. IEE Transactions on Geoscience and Remote Sensing 44, 420–431 (2006)
[9] Watkins, A., Timmis, J., Boggess, L.: Artificial immune recognition system (AIRS): An immune inspired supervised machine learning algorithm. Genet. Program, Evolve. 5 (2004)
[10] Aydin, I., Karakose, M., Akin, E.: Artificial Immune Based Support Vector Machine Algorithm for Fault Diagnosis of Induction Motors. In: Electrical Machines and Power Electronics, 2007. ACEMP 2007, pp. 217–221 (2007)
[11] Yuan, S., Chu, F.: Fault Diagnosis Based on Support Vector Machines with Parameter Optimisation by Artificial Immunisation Algorithm. Mechanical Systems and Signal Processing 21 (2007)
[12] Timmis, J., Andrews, P., Owens, N.: An Interdisciplinary Perspective on Artificial Immune Systems. Evolutionary Inteligence 1, 5–26 (2008)
[13] Kelsey, J., Timmis, J.: Immune Inspired Somatic Contiguous Hypermutation for Function Optimisation. In: Cantú-Paz, E., Foster, J.A., Deb, K., Davis, L., Roy, R., O'Reilly, U.-M., Beyer, H.-G., Kendall, G., Wilson, S.W., Harman, M., Wegener, J., Dasgupta, D., Potter, M.A., Schultz, A., Dowsland, K.A., Jonoska, N., Miller, J., Standish, R.K. (eds.) GECCO 2003. LNCS, vol. 2723, pp. 207–218. Springer, Heidelberg (2003)
[14] Zimmerman, R.D., Murillo-Sanchez, C.E., Gan, D.D.: MATPOWER (2007)
[15] Alsac, O., Stott, B.: Optimal Load Flow with Steady State Security. IEEE Transactions on Power Apparatus and Systems 93 (1974)
[16] Ajjarapu, V.: Compuatational Techniques for Voltage Stability Assessment and Control. Springer, Iowa (2006)
[17] Wilson, D.R., Martinez, T.R.: Improved Heterogeneous Distance Functions. Journal of Artificial Intelligence Research 6, 1–34 (1997)
[18] Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis (1990)
[19] Cortes, C., Vapnik, V.: Support Vector Networks. In: Machine Learning, pp. 273–297 (1995)
[20] Timmis, J., Edmonds, C.: A Comment on Opt-AiNET: An Immune Network Algorithm for Optimisation. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 308–317. Springer, Heidelberg (2004)
[21] Castro, L.N.d., Timmis, J.: An Artificial Immune Network for Multimodal Function Optimization. In: Proceedings of IEEE Congress on Evolutionary Computation (2002)
[22] Brownlee, J.: Optimization Algorithm Toolkit (2008)
[23] Waikato, T.U.o.: WEKA (2009), http://www.cs.waikato.ac.nz/ml/weka/
[24] Chang, C.-C., Lin, C.-J.: LIBSVM - A Library for Support Vector Machines (2008)
[25] Chow, J.H.: Applied Mathematics for Restructured Electric Power Systems: Optimization, Control, and Computational Intelligence, p. 70 (2005)
[26] Dabbagchi, I., Christie, R.: IEEE 14 Bus Test Case (1993)

Artificial Immune System Applied to the Multi-stage Transmission Expansion Planning

Leandro S. Rezende, Armando M. Leite da Silva, and Leonardo de Mello Honório

UNIFEI – Federal University of Itajubá, Minas Gerais, Brazil
{leandrorezende,armando,demello}@unifei.edu.br

Abstract. Transmission expansion planning (TEP) is a complex optimization task to ensure that the power system will meet the forecasted demand and the reliability criterion, along the planning horizon, while minimizing investment, operational, and interruption costs. Metaheuristic methods have demonstrated the potential to find good feasible solutions, but not necessarily optimal. These methods can provide high quality solutions, within an acceptable CPU time, even for large-scale problems. This paper presents an optimization tool based on the Artificial Immune System used to solve the TEP problem. The proposed methodology includes the search for the least cost solution, bearing in mind investments and ohmic transmission losses. The multi-stage nature of the TEP will be also taken into consideration. Case studies on a small test system and on a real sub-transmission network are presented and discussed.

Keywords: Transmission Expansion Planning, Artificial Immune System, Power System, Metaheuristics.

1 Introduction

The main objective of the multi-stage transmission expansion planning (TEP) is to define where, when, and what reinforcements should be placed in the electrical network to ensure an adequate level of energy supply to customers, taking into account the load growth and new generator capacities. In a competitive energy market, TEP is a complex optimization task to ensure that the power system will meet the forecasted demand and the reliability criterion, along the planning horizon, while minimizing investment, operation and interruption costs. This practice is the only rational response to conflicting customer and regulatory demands [1, 2].

Owing to today's power network dimensions, random behavior of transmission and generation equipment, load growth uncertainties, new generator source types and locations, discrete nature of decisions, market aspects, etc., the TEP problem has become combinatorial, stochastic, and highly complex. Even considering only deterministic aspects, it is very difficult to find the optimal solution for TEP problems, since it requires the use of combinatorial algorithms, which have great difficulty in solving medium size problems. If parameter uncertainties and chronological aspects are added to these problems, the optimal solution becomes almost inaccessible.

The chronological aspect refers to the multi-stage nature of the TEP problem. It requires the consideration of multi-time periods, determining possible sequences

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 178–191, 2009.

(i.e. stage-by-stage) of transmission reinforcements. To circumvent the multi-stage nature, simplified studies (also known as static analyses) determine, for just one stage, where new transmission facilities should be installed. Even so, the search for the optimal solution is still combinatorial. Several works to solve TEP problems can be found in the literature [2-15]. However, only a few works have considered the multi-stage nature of the TEP problem [7, 9, 12, 15].

Metaheuristic methods [5-16] have demonstrated the potential of finding good feasible solutions, but not necessarily optimal. Numerous advantages can be linked to these methods: the software complexity is simple, they are able to mix integer and non-integer variables, and also present a faster time-response. The success of such methods is related to their ability to avoid local optima by exploring the basic structure of each problem. They can provide high quality solutions, within an acceptable CPU time, even for large-scale problems. Several metaheuristic tools have evolved in the last decade to solve TEP problems, e.g.: Simulated Annealing (SA) [5]; Tabu Search (TS) [6, 7, 15]; Genetic Algorithms (GA) [8, 9]; Greedy Randomized Adaptive Search Procedure (GRASP) [10, 11]; Evolution Strategies (ES) [12, 15]; Differential Evolution (DE) [13]; Particle Swarm Optimization (PSO) [14]; and Ant Colony Optimization (ACO) [15].

Another possibility to solve TEP problems is the Artificial Immune System (AIS). This metaheuristic is based on the biological principal of the bodies' immune systems. An immunological system has major characteristics that can be used in learning and optimization [17-20]. For optimization problems, four topics are especially interesting: proliferation, mutation, selection, and memory cells. While proliferation is the capability of generating an offspring from the parent cells, mutation is the ability of searching through the solution space for near optimum points. Since the Karush-Khun-Tucher (KKT) conditions can not be tested for this problem, one can not ensure the optimality of this process. The mutation occurs with two different characteristics: hypermutation, which is responsible for local searching, and receptor editing that carries out global searching. The selection process is responsible for eliminating low-affinity cells, while the activation of the memory cells results in a more intense and quicker immune reaction in comparison to activation of naive cells. These features make AIS a powerful optimization tool. The same principle can be used to search for the optimum solution of complex problems related to electric power systems.

This work presents an optimization tool based on the AIS metaheuristic to solve the TEP problem. Other heuristics, such as the one used in GRASP [10], are also used to construct better quality initial populations that in some way represent the memory cells. The TEP problem includes the search for the least cost solution, bearing in mind investments and ohmic transmission losses (i.e. *Joule* effect in the lines). Also, the multi-stage nature of the TEP will be considered. The paper is organized as follows: Section 2 shows the formulation of the TEP problem; Section 3 presents the AIS metaheuristic and how to adapt it to the TEP problem; Section 4 shows the application of AIS to solve a case study involving a simple test system; and, finally, Section 5 presents the conclusions, which include some comments about the application of AIS to solve real transmission expansion problems.

2 Transmission Expansion Planning Problem

Different from most works that perform a static planning and use other optimization tools, the present work applies, for the first time, the AIS to solve the TEP problem considering the chronology of the investments and including the cost of ohmic transmission losses in the objective function. In this type of problem, the interest is not only to define what reinforcements should be placed in the electrical network and their corresponding locations, but also when they will be added along the planning horizon to ensure an adequate level of energy supply to customers. At the end, the best expansion plans must be selected in order to minimize the present value costs involved in the objective function.

The first step, considering the mathematical formulation of the TEP problem, is to define the representation of the solution matrix S (also named sequence), which corresponds to the candidate plans. Under the AIS taxonomy, matrix S^k represents an individual or antibody, which can be defined as follows:

$$S^k = \begin{bmatrix} s^k_{y1} & s^k_{y2} & \cdots & s^k_{yl} & \cdots & s^k_{yn} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s^k_{t1} & s^k_{t2} & \cdots & s^k_{tl} & \cdots & s^k_{tn} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ s^k_{21} & s^k_{22} & \cdots & s^k_{2l} & \cdots & s^k_{2n} \\ s^k_{11} & s^k_{12} & \cdots & s^k_{1l} & \cdots & s^k_{1n} \end{bmatrix}. \tag{1}$$

where n indicates the number of possible network branches allowed to receive reinforcements, y corresponds to the number of stages considering the planning horizon, and s^k_{tl} refers to the total number of reinforcements at stage t, and branch l in relation to the base system network configuration. If only one stage is considered (static problem), one has the following solution vector:

$$S^k_t = \begin{bmatrix} s^k_{t1} & s^k_{t2} & \cdots & s^k_{tl} & \cdots & s^k_{tn} \end{bmatrix}. \tag{2}$$

During the optimization process, provided by the AIS, the selection of the candidate plans is done considering the whole horizon, aiming to minimize the present value of the objective function.

In order to build a sequence, the maximum number of reinforcements in each branch must be satisfied. In addition, it is important to ensure a coordination scheme of the added reinforcements. For example, reinforcements inserted in stage t must be mandatorily included in the following stages t+1, t+2,..., y. These constraints are defined as follows:

$$\begin{aligned} s^k_{tl} &\le N_{l\,max} \quad \forall l \in \{1,...,n\}; \quad \forall t \in \{1,..., y\}. \\ s^k_{tl} &\le s^k_{(t+1)l} \quad \forall l \in \{1,...,n\}; \quad \forall t \in \{1,..., y-1\}. \end{aligned} \tag{3}$$

where $N_{l\,max}$ refers to the maximum number of reinforcements permitted in branch l.

Once a sequence is found, for a specific planning horizon composed of y stages, the TEP problem can be described by:

$$Min\ f(S^k) = \sum_{t=1}^{y} \frac{\left(\sum_{l=1}^{n} Cinv_l m_{t\,l}^k + Closs_t^k + \alpha^T r_t\right)}{(1+e)^{h(t)}}. \quad (4)$$

where $f(S^k)$ represents the total cost function in present value associated with the k sequence; e indicates the discount rate; $h(t)$ corresponds to a function that informs the numerical difference between the year of stage t and the base year; $Cinv_l$ is the unit investment cost for the reinforcements added in the branch l; $m_{t\,l}^k$ refers to the number of reinforcements located at stage t and branch l of the sequence k, i.e. $m_{t\,l}^k = s_{t\,l}^k - s_{(t-1)\,l}^k$ (if t represents the initial stage, then $m_{t\,l}^k = s_{t\,l}^k$); $Closs_t^k$ represents the costs associated with ohmic losses cost at the stage t of the sequence k; α refers to the load shedding penalty vector; and r_t is the load not-supplied vector at stage t.

Every time that a configuration is obtained by the AIS, an evaluation is performed through a linear programming (LP) based on a linearized DC power flow model [21], as described in the following:

Minimize:

$$z = \alpha^T r\ . \quad (5)$$

subject to:

$$g + r + B\theta = d\ . \quad (5.1)$$

$$0 \le g \le g_{max}\ . \quad (5.2)$$

$$0 \le r \le d\ . \quad (5.3)$$

$$|f| \le f_{max}\ . \quad (5.4)$$

Where g is the generation bus vector; B represents the susceptance matrix; θ is the voltage angle vector; d is the load vector; g_{max} refers to the generation limit bus vector; f is the power flow vector; and f_{max} represents the power flow limit vector.

This LP, which is applied to each stage of the planning horizon, can be efficiently solved by the interior point method. The Lagrange multipliers associated with the constraints of eq. (5.1) are of high interest, since they can assist in the construction process of better quality initial sequences (memory cells). These multipliers measure the benefit in terms of load not-supplied index concerning changes on the circuit by the addition of new reinforcements. Denoting π^d as this Lagrange multiplier vector, the benefits can be estimated by [10]:

$$\pi_{ij}^d = (\theta_i - \theta_j)(\pi_i^d - \pi_j^d)\ . \quad (6)$$

where π_{ij}^d is the Lagrange multiplier associated with the branch susceptance connecting buses i and j.

Therefore, the greedy or heuristic function defined as follows, which also considers the investment costs associated with the new reinforcements, can be of high interest to assist the AIS to find sequences of better quality [15]:

$$\eta(i,j) = \frac{\pi_{ij}^{d}}{Cinv_{l}}. \tag{7}$$

where $Cinv_l$ was already defined in eq. (4) and l is the branch that connects buses i and j.

In order to include the costs associated with the ohmic losses in the optimization process, a special DC flow model is used. Basically, the losses are calculated using the voltage angle vector obtained by the LP solution of a given configuration. Afterwards, these losses are distributed as loads, where each bus i and j receives half of ohmic losses found in the circuit that connects these buses. Again, a new LP is solved considering this new increased load and a new voltage angle vector is found. This corresponds to the solution for a given configuration.

The amount of losses associated with the circuit between buses i and j can be approximated as [15]:

$$P_{ij} = (r_{ij} \times f_{ij}^{2}). \tag{8}$$

where r_{ij} is the resistance of the circuit and f_{ij} is the active power flow, and all quantities are in pu. The total ohmic losses cost (*Closs*) is given by [15]:

$$C_{loss} = 8736 \times C_{kWh} \times LF \times \sum_{\forall_{ij}} P_{ij}. \tag{9}$$

where C_{kWh} represents the loss unit cost in US$/kWh; LF is the loss factor, which modulates the load curve and the value 8736 aims at converting the incremental loss costs into annualized costs.

3 Artificial Immune System

3.1 Basic Concepts

An AIS optimization framework intends to capture some of the principles of the natural immune system (NIS), within a computational environment. As in every intelligent-based method, the AIS is a search methodology that uses heuristics to explore interesting areas in the solution space. However, unlike other intelligence-based methods, the main algorithms presented in literature, CLONALG [18] and opt-AINet [20], provide tools to perform simultaneous local and global searches. These tools are based on two concepts: hypermutation and receptor editing. While hypermutation is the ability to execute small steps towards a higher affinity antibody (hAb), leading to local optima, receptor editing provides large steps through the solution space, which may lead into a region where the search for a hAb is more promising.

The main statement of the CLONALG algorithm is that progressive adaptive changes can be achieved by cumulative blind variation based only upon an affinity increase caused by random mutation and individual selection. It also states that, through these

principles, it is possible to generate high-quality solutions to complex problems. Next, the main operators of the CLONALG algorithm will be presented. In addition, it will be described some modifications that allowed it to achieve better quality results.

3.2 Adaptation of CLONALG to the TEP Problem

In order to adapt the CLONALG algorithm to solve the TEP problem, sequence S^k, described by eq. (1), corresponds to each antibody of the immune system. In addition, each element s_{tl}^k, which refers to the reinforcement options, corresponds to a position of this antibody.

The first step in the original CLONALG algorithm is to randomly choose a population of antibodies that will evolve through generations. One huge advantage of the approach developed in this work is the use of memory cells, which are not considered in other population-based algorithms neither in the original CLONALG in optimization problems. These memory cells are used as the initial population of high affinity antibodies, which are obtained by emulating a previous scenario, rather than computational simulations. This process is named *Intelligent Initialization*, which consumes very little computational effort. Considering the TEP problem, better performance can be achieved if an initial population of good antibodies is selected, as described in several works [7, 9, 10, 12, 15]. In this case, there is a higher chance of finding better plans through the evolution of the optimization process. In this work, the heuristic function given by eq. (7) is used as the basic knowledge to build the initial good quality sequences. The computational gain of using these memory cells, along with other improvements in the original CLONALG, makes the proposed algorithm very suitable for the problem to be solved, as it will be shown in Section 4.

The *Intelligent Initialization* mechanism or process used to build good quality initial sequences (memory cells) is presented as follows:

1. An evaluation order is randomly chosen to consider the stages of the planning horizon; select the first position of the order, and go to step 2.
2. Perform a DC power flow according to eq. (5) including the evaluation of ohmic losses, based on the transmission network of the previous stage already analyzed. If a feasible configuration is found (loss of load = 0), go to step 5; otherwise, go to step 3.
3. Evaluate the heuristic function according to eq. (7) and select branch c, which connects buses i and j, to receive a reinforcement through eqs. (10) and (11). In this step, the constraints defined by eq. (3) must be satisfied. Go to step 4.

$$c = \begin{cases} arg\ \max_{(i,j)\in T^k} \{[\eta(i,j)]\}, & if \quad q \le q_0 \\ C, & otherwise \end{cases} . \tag{10}$$

where $\eta(i,j)$ represents the value of the heuristic function associated with the branch (i,j); *arg max* is a function that selects the maximum value of the heuristic function; T^k is the set of reinforcements not yet selected by the process; q represents a random number uniformly distributed in [0,1]; q_0 is an adjustable parameter ($0 \le q_0 \le 1$), which indicates the degree of intensity that the process will concentrate on the best

reinforcements suggested by the heuristic function; C corresponds to a random variable that follows a discrete distribution given by eq. (11), where $p^k(i,j)$ is the probability of adding the reinforcement (i,j) to sequence k.

$$p^k(i,j)=\begin{cases}\dfrac{[\eta(i,j)]}{\sum_{(v,w)\in T^k}\{[\eta(v,w)]\}}, & if\ (i,j)\in T^k\\ 0, & otherwise.\end{cases} \quad . \tag{11}$$

4. Perform a new DC power flow including the ohmic losses. If the new configuration is feasible, go to step 5; otherwise, return to step 3.
5. If the stage under analyses corresponds to the last one considering the evaluation order defined in step 1, store the sequence and go to step 6; otherwise, select the next stage of the evaluation order and return to step 2.
6. If the initial population has been defined, the *Intelligent Initialization* mechanism is interrupted; otherwise, the process must be restarted in order to build a new sequence; return to step 1.

According to the CLONALG algorithm, the reproduction operator is responsible for cloning the antibodies with higher affinity (identical copies). The better the antibody affinity is, the higher the number of clones generated by it. However, if the optimization process aims at locating multiple optima (the objective of the TEP problem is to identify the set of best sequences and not only the best one), it may be more interesting to select all antibodies to generate clones. In addition, independently of the antibodies affinity, the number of copies provided by each antibody may be the same, as it is observed in [18]. As a result, a better exploration of the search space may be reached when the hypermutation operator is used.

Since n_{Ab} is the number of antibodies (sequences) and n_{clo} is the number of clones generated by each antibody S^k, after the application of this operator, there will be $G^{n_{Ab}}$ groups consisted of the original antibody S^k and its respective clones $Cl^{k,u}$:

$$\begin{aligned}&G^k=\left\{S^k, Cl^{k,u}\right\},\ \text{where}\ Cl^{k,u}=S^k\\ &\forall k\in\{1,2,...,n_{Ab}\}\ ;\ \forall u\in\{1,2,...,n_{clo}\}.\end{aligned} \tag{12}$$

As already mentioned, the hypermutation operator aims to achieve higher affinity antibodies, i.e. to provide better quality antibodies. The hypermutation process consists of adding a perturbation Z_t^k to each clone $Cl_t^{k,u}$, at all stages t of the planning horizon, as shown by eqs. (13) and (14):

$$\tilde{C}l_t^{k,u}=Cl_t^{k,u}+Z_t^k\ . \tag{13}$$

$$Z_t^k=\sigma\times\left[N_{t\,1}(0,1)\quad N_{t\,2}(0,1)\quad\cdots\quad N_{t\,l}(0,1)\quad\cdots\quad N_{t\,n}(0,1)\right]\ . \tag{14}$$

where $\tilde{C}l_t^{k,u}$ represents a new antibody, which is obtained through the mutation of clone $Cl_t^{k,u}$; σ is the mutation magnitude; $N_{t\,l}(0,1)$ corresponds to a Gaussian distribution with zero mean and unit variance. As it can be seen, the perturbation Z_t^k is continuous.

Therefore, a round function must be applied to each position of the new antibodies, since the reinforcements are discrete variables.

After using the hypermutation operator, the strategy adopted by the selection process is to keep the n_b best antibodies, based on Equation (4), from each group G^k (parent and its respective clones) to create a new group G^*. It is important to observe that if just the best clone is selected, all others will be disregarded and valuable information may be lost as mentioned in [18].

Avoiding a combinatorial explosion, all antibodies of the resulted group G^* are compared at each iteration. This step is represented by the receptor editing that selects the best antibodies of group G^* in order to minimize Equation (4). In addition, it is not allowed the selection of identical antibodies. At the end, the best and distinct n_{ab} antibodies will evolve to the next generation, keeping the same size of the initial population. This process ensures a better quality and diversified population for the next generation, since there will not be identical antibodies.

In the original CLONALG, the receptor editing operator aims at substituting the lower affinity antibodies by new ones. The objective is to escape from local optima and provide a better exploration of the search space. The CLONALG proposed in this work achieves the same objective in assuring a selection of distinct antibodies for the next generation. In this case, it is unnecessary to substitute the lower affinity antibodies. Therefore, some modifications were made in the original CLONALG, mainly regarding the selection process and the receptor editing.

The proposed CLONALG algorithm used to solve the TEP problem is represented by the diagram shown in Fig. 1.

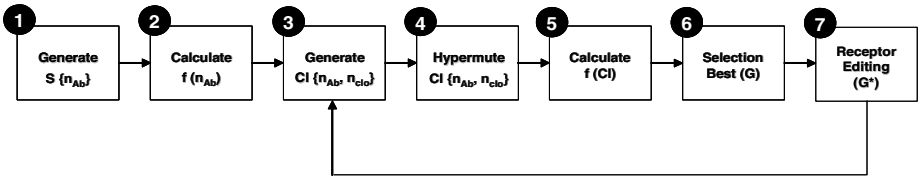


Fig. 1. Proposed CLONALG-based block diagram for the TEP problem

where each step is detailed as follows:

1. Create a population of antibodies according to the *Intelligent Initialization* algorithm;
2. Calculate the value of the objective function for each antibody. The result is stored in f and represents the population affinity for the optimization process;
3. For each antibody, a set of n_{clo} clones is generated;
4. Each clone passes through a process of hypermutation according to the eqs. (13) and (14). All clones must satisfy the constraints imposed by eq. (3). The stages of the planning horizon are evaluated one-by-one and follow a decreasing year order. For each configuration found, a DC power flow including ohmic losses is performed. Only feasible sequences are allowed;
5. Calculate the affinity of each clone;
6. Select the n_b best sequences from each group G^k, to create a new group G^*;

7. Apply the receptor editing to select the best and distinct n_{ab} antibodies from G^* that will evolve to the next generation. If the maximum number of generations is achieved, n_{ger}^{max}, the search is interrupted and the best sequences found during the optimization process are stored.

An explanation must be given in relation to the step 4. The same mutation magnitude is used to all clones generated by the different antibodies. However, considering the planning horizon, a mutation of smaller magnitude is selected for the last stage when compared with the magnitude of the other stages. If a high magnitude is chosen for the last stage, sequences with more reinforcements may be obtained and, therefore, with a worse affinity. On the other hand, since the other stages must have a coordination of the reinforcements added, the mutation operator is performed only over some branches. Therefore, if a small magnitude is chosen, the chance of not having changes in the number of reinforcements in any branch of the system is increased, thus contributing for a lower diversity of the population.

In the next section, the application of the modified CLONALG will be presented in order to solve a case study involving the TEP problem. In addition, a performance comparison between the original and the proposed CLONALG will be shown, where the benefits of using the modified version for this kind of problem will be validated.

4 Application of the Modified CLONALG to the Transmission Expansion Planning Problem

A case study on a test system is presented and discussed in the next subsection. All CPU times refer to a *Pentium Core 2 Duo* processor (2.66GHz). The programs were developed using the software MATLAB.

The test system has 6 buses (3 generation and 3 load buses) and 11 double transmission circuits. Figure 2 shows the unifilar diagram of this system. Considering the reference or starting year, the installed capacity is 260 MW and the load peak is 210 MW. All deterministic bus and circuit data can be found in [7,12,15].

The system planning horizon is 8 years, and for each year the load and generating capacities are increased by 25%, in relation to the reference year. Therefore, the load and installed capacity will be 630 MW and 780 MW, respectively, at the end of the period of analysis (8th year). New transmission reinforcements can only be added to the existing branches and are limited to 3 reinforcements per branch. In order to calculate the present value of eq. (4), it is used a discount rate (e) of 10%. In relation to the costs associated with the ohmic losses, the following parameters are used: C_{kWh}=0.10 US$/kWh and LF=0.6144.

Considering the proposed CLONALG algorithm (Fig. 1), the following parameters were selected: number of clones generated by each antibody, n_{clo}=5; number of sequences selected from each group G^k, n_b=2; mutation magnitude applied to branches in the last stage of planning horizon, σ_1=0.3; mutation magnitude of the other stages, σ_2=0.6. The algorithm is interrupted after a maximum number of generations, n_{ger}^{max} =100. The final objective of the search is to find the 25 best sequences that minimize the present value cost according to eq. (4).

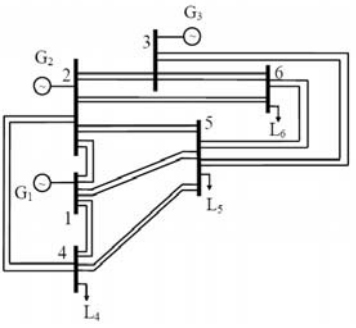


Fig. 2. Test system

The performance of the proposed methodology is measured based on two criteria: first, through the index that measures the quality of the best sequences found, *Iq* (%); second, through the identification of how many simulated cases, in percent, were able to capture the best plan known, I_{best} (%). The quality index, *Iq* (%), can be calculated as follows:

$$Iq(\%) = \frac{1}{n_{best}} \sum_{b=1}^{n_{best}} \frac{f(S^b) - f(S^{best})}{f(S^{best})} \times 100 \tag{15}$$

where n_{best} corresponds to the number of best sequences selected at the end of the optimization process; *f()* is the function that minimizes the costs in present value, given by eq. (4); S^b indicates one of the best sequences stored after the end of the search; and S^{best} refers to the best sequence known for the problem, independently if it is not found in a given simulated case.

Figure 3 shows the importance of considering an *Intelligent Initialization* mechanism, whose objective is to provide a good quality population (memory cells) at the beginning of optimization process. The results presented correspond to the simulation of 10 cases using different seeds randomly chosen. The tests considered two different sizes for the antibody population, n_{Ab}={10;15}, and in both situations the *Intelligent Initialization* process contributed to better quality indices (i.e. smaller values) and to a higher probability of finding the best known sequence. The CPU time is about the same independent of the initialization process.

It is important to mention that using the *Random Initialization* would require the definition of a higher size for the antibody population, or the interruption of the search well beyond 100 generations to achieve a similar performance of the *Intelligent Initialization*. This would result in a higher computational effort.

Table 1 presents the results achieved by the proposed CLONALG considering the application of the *Intelligent Initialization* process. It is shown the quality index, the present value of the best and worst sequence among the 25 selected ones, and the CPU time for the 10 cases simulated. In addition, the mean values are also presented. As a conclusion, one can observe that there is a little improvement in the quality index and in the probability of finding the best sequence known if it is considered a population size of 15, while the mean CPU time is increased circa 47%.

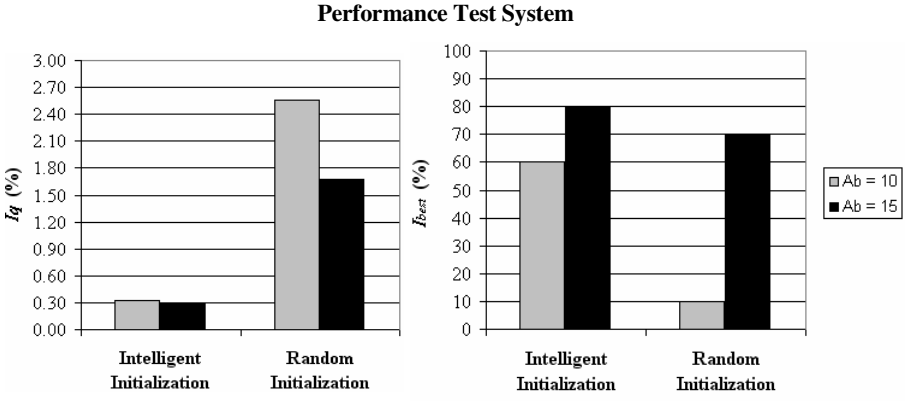


Fig. 3. Initialization process: Intelligent x Random

Table 1. Performance of the proposed CLONALG

Case	n_{Ab} = 10				n_{Ab} = 15			
	Iq(%)	Min. Val. (10^6 US$)	Max Val. (10^6 US$)	T (min)	*Iq*(%)	Min. Val. (10^6 US$)	Max Val. (10^6 US$)	T (min)
1	0.322	217.82	219.02	30.6	0.306	218.02	218.88	47.5
2	0.345	218.05	218.99	31.7	0.274	217.82	218.83	49.2
3	0.331	217.82	218.91	34.1	0.319	217.82	218.84	48.3
4	0.283	217.82	218.86	34.0	0.264	217.82	218.79	49.1
5	0.344	218.02	219.02	32.8	0.268	217.82	218.81	48.7
6	0.331	217.82	219.00	33.9	0.262	217.82	218.79	49.3
7	0.303	217.82	218.88	33.9	0.277	217.82	218.82	49.3
8	0.309	217.82	218.88	34.2	0.264	217.82	218.83	49.8
9	0.307	218.02	218.87	33.4	0.296	218.02	218.81	48.5
10	0.385	218.04	219.08	32.0	0.284	217.82	218.84	48.3
Mean	0.326	217.91	218.95	33.1	0.281	217.86	218.82	48.8

Table 2. Best sequence for the test system

Year	Reinforcements Added							Annual Costs (10^6 US$)		
	1 – 4	1 – 5	2 – 4	2 – 5	2 – 6	3 – 5	3 – 6	Inv.	Ohmic Losses	Total
8	0	0	1	0	0	0	1	80.00	7.50	87.50
7	1	0	0	1	0	0	0	45.00	7.29	52.29
6	0	1	0	0	1	1	0	65.00	6.41	71.41
5	1	0	0	0	0	0	0	25.00	5.80	30.80
4	0	1	0	0	0	0	1	60.00	4.99	64.99
3	1	0	0	0	0	0	0	25.00	4.36	29.36
2	0	1	0	0	0	0	0	20.00	3.49	23.49
1	0	0	0	0	0	0	0	0.00	2.79	2.79
0	0	0	0	0	0	0	0	0.00	2.34	2.34
Total Present Value (10^6 US$)								188.92	28.90	217.82

Table 2 presents the best sequence found for the Test System, which has a total present value cost of US$ 217.82 millions. The annual costs related to the investments and ohmic losses, as the reinforcements added along the planning horizon, are also shown.

The original CLONALG [18] is also applied to solve the same TEP problem. With this algorithm, 20% of the population is renewed by the receptor editing at each 10 generations, i.e. the worst 20% sequences are substituted by new ones. The search is interrupted after 100 generations. Considering a population size of 15 antibodies and a simulation of 10 cases, the proposed CLONALG achieved a mean quality index of 0.281% (Table 1), while the original CLONALG achieved a mean quality index of 1.351%. Including the *Intelligent Initialization* in the original CLONALG, the corresponding quality index is improved to 0.377%, which still has a worse performance than the proposed algorithm.

Regarding the number of cases that captured the best plan known for this system, 80% of the cases identified the best sequence when the proposed CLONALG (Table 1) is used, while the original CLONALG is not able to identify the best plan in all simulated cases. When the *Intelligent Initialization* is included, the original CLONALG captured the best sequence in 60% of the cases. Therefore, the improvement in the indices is not only due to the *Intelligent Initialization*, but also related to the modifications made in the selection process and in the receptor editing.

Concerning the computational effort, it was verified that the original CLONALG spent at about 63 minutes. However, when including the *Intelligent Initialization*, the computational effort is reduced to 49 minutes, which is similar to the time spent by the proposed CLONALG (Table 1).

Although several approaches have evolved in the last years to solve the TEP problem, only a few works have considered the multi-stage nature [7, 9, 12, 15], which is considered in the present work. Reference [15] makes a performance comparison among three approaches: ES, TS, and ACO. However, as the methodology to obtain the sequences is different (the chronological aspect is separately considered as the solution of several static problems), it would be unfair to make any comparison with the results achieved in the present work, since in here the multi-stage nature is solved taking into account all the stages simultaneously. A new study (extra case) has been carried out using the TS approach to solve the TEP problem considering all the stages simultaneously to obtain the best sequences. As a preliminary result, adjusting the TS parameters in order to achieve the same computational effort of the proposed CLONALG, it was found a quality index of 0.554%, which indicates a better performance of the proposed algorithm in this work. A more detailed performance comparison between these and others approaches will be presented in a future work.

5 Conclusions

This work proposed an optimization tool based on the Artificial Immune System (AIS) metaheuristic to solve the multi-stage transmission expansion planning (TEP) problem. The final objective is to find the best sequences that minimize the present value cost of investments and ohmic transmission losses. Some modifications are proposed in the basic (i.e. original) CLONALG algorithm. The most important one is related to the

definition of the initial population. It could be concluded that better results are achieved if an initial good quality population is defined, which is obtained through a process or mechanism named *Intelligent Initialization*. The results obtained using the proposed algorithm, i.e. the modified CLONALG, to solve the test system were better than both the original CLONALG and the TS approach, in terms of the solution quality and computational effort.

As an application in a real system, a case study using a sub-transmission network was also performed. This system is part of CEMIG (State Energy Company of Minas Gerais, Brazil), and it is composed by 12 buses and 22 allocation branches to receive reinforcements. An expansion horizon of 10 years is considered. Even for this small sub-transmission network, a total of 3.29×10^{38} sequences of reinforcements are, in principle, eligible for this TEP problem. The results demonstrated a better performance of the proposed CLONALG using the *Intelligent Initialization* when compared to the original one and also to the TS-based approach.

A more comprehensive study comparing different metaheuristics for large network configurations are among the future researches. A preliminary study has been carried out including some metaheuristics (Tabu Search, Evolution Strategies, Particle Swarm Optimization, Ant Colony Optimization, Differential Evolution), and the proposed CLONALG-based methodology has demonstrated to be one of the best optimization tools to solve the TEP problem.

References

1. Billinton, R., Salvaderi, L., McCalley, J.D., Chao, H., Seitz, T., Allan, R.N., Odom, J., Fallon, C.: Reliability Issues in Today's Electric Power Utility Environment. IEEE Trans. on Power Syst. 12, 1708–1714 (1997)
2. Chowdhury, A.A., Koval, D.O.: Value-based System Facility Planning. IEEE Power and Energy Magazine 2, 58–67 (2004)
3. Latorre, G., Cruz, R.D., Areiza, J.M., Villegas, A.: Classification of Publications and Models on Transmission Expansion Planning. IEEE Trans. on Power Syst. 18, 938–946 (2003)
4. Xu, Z., Dong, Z.Y., Wong, K.P.: Transmission Planning in a Deregulated Environment. IEE Proceedings-Generation, Transmission and Distribution 153, 326–334 (2006)
5. Gallego, R.A., Alves, A.B., Monticelli, A., Romero, R.: Parallel Simulated Annealing Applied to Long Term Transmission Network Expansion Planning. IEEE Trans. on Power Syst. 12, 181–187 (1997)
6. Gallego, R.A., Romero, R., Monticelli, A.: Tabu Search Algorithm for Network Synthesis. IEEE Trans. on Power Syst. 15, 490–495 (2000)
7. Leite da Silva, A.M., Manso, L.A.F., Resende, L.C., Rezende, L.S.: Tabu Search Applied to Transmission Expansion Planning Considering Losses and Interruption Costs. In: 10th Probabilistic Methods Applied to Power Systems – PMAPS, Puerto Rico (2008)
8. Gil, H.A., da Silva, E.L.: A Reliable Approach for Solving the Transmission Network Expansion Planning Problem Using Genetic Algorithms. Electric Power System Research 58, 45–51 (2001)
9. Escobar, A.H., Gallego, R.A., Romero, R.: Multistage and Coordinated Planning of the Expansion of Transmission Systems. IEEE Trans. on Power Syst. 19, 735–744 (2004)

10. Binato, S., Oliveira, G.C., Araújo, J.J.: A Greedy Randomized Search Procedure for Transmission Expansion Planning. IEEE Trans. on Power Syst. 16, 247–253 (2001)
11. Faria Jr., H., Binato, S., Resende, M.G.C., Falcão, D.M.: Power Transmission Network Design by Greedy Randomized Adaptive Path Relinking. IEEE Trans. on Power Syst. 20, 43–49 (2005)
12. Leite da Silva, A.M., Sales, W.S., Resende, L.C., Manso, L.A.F., Sacramento, C.E., Rezende, L.S.: Evolution Strategies to Transmission Expansion Planning Considering Unreliability Costs. In: 9th Probabilistic Methods Applied to Power Systems – PMAPS, Stockholm (2006)
13. Dong, Z.Y., Lu, M., Lu, Z., Wong, K.P.: A Differential Evolution Based Method for Power System Planning. In: IEEE Congress on Evolutionary Computation, Vancouver, pp. 2699–2706 (2006)
14. Jin, Y.X., Cheng, H.Z., Yan, J.Y., Zhang, L.: New Discrete Method for Particle Swarm Optimization and Its Application in Transmission Network Expansion Planning. Electric Power Systems Research 77, 227–233 (2007)
15. Leite da Silva, A.M., Sacramento, C.E., Manso, L.A.F., Rezende, L.S., Resende, L.C., Sales, W.S.: Metaheuristic-Based Optimization Methods for Transmission Expansion Planning Considering Unreliability Costs. In: Castronuovo, E.D. (ed.) Optimization Advances in Electric Power Systems. Nova Publishers, USA (2008)
16. Lee, K.Y., El-Sharkawi, M.A.: Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems. Wiley - IEEE Press Series on Power Engineering (2008)
17. Hunt, J.E., Cooke, D.E.: Learning Using an Artificial Immune System. Journal of Network and Computer Applications 19, 189–212 (1996)
18. Castro, L.N., Zubben, F.J.V.: Learning and Optimization Using the Clonal Selection Principle. IEEE Transactions on Evolutionary Computation 6, 239–251 (2002)
19. Honorio, L.M., Leite da Silva, A.M., Barbosa, D.A.: A Gradient-based Artificial Immune System Applied to Optimal Power Flow Problems. In: de Castro, L.N., Von Zuben, F.J., Knidel, H. (eds.) ICARIS 2007. LNCS, vol. 4628, pp. 1–12. Springer, Heidelberg (2007)
20. Andrews, P.S., Timmis, J.: On Diversity and Artificial Immune Systems: Incorporating a Diversity Operator into aiNet. In: Apolloni, B., Marinaro, M., Nicosia, G., Tagliaferri, R. (eds.) WIRN 2005 and NAIS 2005. LNCS, vol. 3931, pp. 293–306. Springer, Heidelberg (2006)
21. Rau, N.S.: Optimization Principles – Practical Applications to the Operation and Markets of the Electric Power industry. Wiley - IEEE Press Series on Power Engineering (2003)

Immune Learning in a Dynamic Information Environment

Nikolaos Nanas[1], Manolis Vavalis[2], and Lefteris Kellis[2]

[1] Lab for Information Systems and Services
Centre for Research and Technology, Thessaly (CE.RE.TE.TH)
n.nanas@cereteth.gr
[2] Computing and Telecomunications Department, University of Thessaly
{mav,ekel}@inf.uth.gr

Abstract. In Adaptive Information Filtering, the user profile has to be able to define and maintain an accurate representation of the user's interests over time. According to Autopoietic Theory, the immune system faces a similar continuous learning problem. It is an organisationally closed network that reacts autonomously to define and preserve the organism's identity. Nootropia is a user profiling model, which has been inspired by this view of the immune system. In this paper, we introduce new improvements to the model and propose a methodology for testing the ability of a user profile to continuously learn a user's changing interests in a dynamic information environment. Comparative experiments show that Nootropia outperforms a popular learning algorithm, especially when more than one topic of interest has to be represented.

1 Introduction

Adaptive Content-based Information Filtering (AIF) is a complex, dynamic problem. It is based on a tailored representation of the user interests called "profile". The user profile has to be able to represent a user's multiple interests and continuously adapt to changes in them. Successful user profiling could play a catalytic role for web personalisation. User profiles could be used to provide a series of personalisation services, including, in addition to the filtering of information coming from a variety of sources, personalised search/browsing, expert finding and others. However, despite the research efforts during the last few years, there are today no successful personalised information delivery systems on the web, based on adaptive content-based user profiles. This lack highlights the difficulty of the problem and indicates that existing models and algorithms can not effectively deal with it.

The viability of a user profile depends on its ability to continuously satisfy the user's information needs. It has to be able to define and maintain an accurate representation of the user interests over time. According to Autopoietic theory the immune network faces a similar problem. It is an *organisationally closed* network that reacts autonomously in order to define and preserve the organism's identity,

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 192–205, 2009.

in what is called *self-assertion* [17]. Self-assertion is a continuous process because both the organism and the environment that it inhabits change over time.

The above analogy has prompted us to develop Nootropia[1], an immune-inspired approach to adaptive user profiling. The algorithm was first introduced in [9] and is thoroughly discussed in [8]. In this paper, we describe for the first time an improved, more distributed version of the algorithm. We also propose a methodology for evaluating the ability of a user profile for continuous adaptation in a dynamically changing environment. Such a methodology has been missing. It is used here to conduct comparative experiments between Nootropia and the well established Rocchio algorithm. The experiments show that Nootropia clearly outperforms the baseline when multiple interests have to be represented and traced over time. They also indicate that Nootropia is a complex dynamic system that can self-regulate both its size and connectivity.

2 Related Work

Content-based information filtering deals mainly with textual information, e.g. documents, for which descriptive features, like terms, can be automatically extracted[2]. In AIF the user profile continuously evaluates the relevance of documents coming from one or more information streams. The documents are then appropriately presented to the user, e.g., in decreasing relevance order. In some cases, such as spam filtering, a threshold on the relevance score of documents is used to decide which documents to present and which to discard. The user's feedback on the presented documents is the basis for the profile's adaptation. Document's judged relevant provide positive feedback and vice versa.

Profile adaptation has been commonly tackled with learning algorithms. Rocchio algorithm [14] in particular (or its variations), is a popular and well studied choice. Here, its serves as the baseline for our experiments and is discussed in detail in section 4.3. The problem with linear learning algorithms, like Rocchio, is that they use constant learning coefficients to define the pace of adaptation, or, in other words, how quickly the profile learns or forgets. This rigid practice is not well suited to the variety of changes in a user's interests. As an alternative, in Reinforcement learning, the learning coefficients are progressively reduced so that what has already been learned is maintained [15]. In some cases, a distinction is made between short-term and long-term profiles which use different learning coefficients [18]. Despite these remedies, it has been argued that learning algorithms cannot easily cope with concept drifts in user interests, such as the emergence of a new topic of interest. In this case, a new profile has to be built from scratch to represent the new topic. The opposite takes place for a no longer interesting topic. The corresponding profile is removed.

The dynamic nature of profile adaptation invites the application of biologically inspired approaches. Genetic Algorithms (GAs) have already been applied to the

[1] Greek word for: "an individual's or a group's particular way of thinking, someone's characteristics of intellect and perception".

[2] Here we will concentrate on documents, although Nootropia could be generalised for other media as well.

problem with some success [5]. However, according to [6], GAs cannot easily cope with multiple and changing interests because of their tendency to converge to a single topic (optimum), thus loosing valuable diversity, and also due to their common reliance on a population with fixed size. Artificial Immune Systems (AIS) can inherently maintain and boost their diversity and can dynamically control the size of the immune repertoire. AIS have been applied for spam filtering [1] and document clustering [2], but, to our knowledge, there is no explicit application of AIS for profile adaptation in AIF. A comprehensive review of evoluationary and immune-inspired information filtering can be found in [7].

In general, models and algorithms for building a user profile representing a user's multiple interests and adapting to a variety of changes in them are lacking and this is reflected by existing methodologies for evaluating AIF systems. Existing evaluation standards, like those proposed by the well established Text Retrieval Conference (TREC)[3], concentrate on the evaluation of a separate user profile per topic of interest. The possibility of a profile representing multiple topics in parallel is ignored, while tests for the ability of profiles to learn a new topic of interest and to forget a no longer interesting topic are excluded.

3 Nootropia in Brief

Nootropia is an immune-inspired user profiling model, influenced by Autopoietic theory. It is described in detail in [8], so here we just summarise the model and we concentrate on some new improvements. When applied to textual information Nootropia maintains a weighted network of terms (single words) to represent a user's multiple interests. Terms in this network correspond to antibodies and their weights to antibody concentrations. Links between terms represent antibody-to-antibody interactions and their weights measure the affinity between antibodies. The antibody network is used to evaluate the relevance of each incoming document, through a non-linear, spreading activation process.

Profile adaptation takes place with a self-organising process, which adjusts the network's structure in response to user feedback. Given a document with positive feedback the most important terms are extracted. This process plays the role of B-cell production by the bone marrow, which introduces new antibody types in the immune repertoire. The network's *dynamics* cause a reinforcement, i.e., an increase in weight/concentration, of existing profile terms (antibodies already in the immune repertoire), which have also been extracted from the feedback document. The main difference of the new version of Nootropia is that the additional weight (concentration) of a reinforced profile term is subtracted equally from those profile terms that it is linked to and not from all profile terms indiscreetly. So, like before, the overall weight of the profile remains constant during the network's dynamics, but a competition between linked terms is introduced. With this modification, how the weight of profile terms is rearranged depends on the network's structure. The network's dynamics become localised because they only affect terms linked to the reinforced terms. This has a positive effect on the

[3] http://trec.nist.gov/

network's diversity because different, unconnected parts of the network, which may represent unrelated topics of interest, do not influence each other. It also enhances endogenous selection because now the survival of a term in the network depends explicitly on the links it establishes with other terms. A side effect of this new kind of dynamics is that there is now the chance of terms surviving indefinitely in the repertoire, even if they never get reinforced, because they do not get involved in the competition with other terms. As a remedy, we introduced in this new version a decay coefficient δ that proportionally reduces the weight of every profile term each time a feedback document is processed. The network's *metadynamics* are caused on one hand by the addition of new terms (new antibody types), extracted from feedback documents, to the profile's repertoire. On the other hand, terms that run out of weight are removed from the repertoire. In this new version, every time a term is purged, its initial weight–i.e., the term's weight when it was first introduced in the repertoire–is subtracted equally from the terms it is linked to. Unlike the previous version, the network's connectivity is again taken into account. Preliminary experiments, not reported here due to space limitations, show that the above improvements have a positive effect on Nootropia's performance. Through self-organisation, Nootropia can maintain a representation of a user's multiple interests and adapt to a variety of changes in them. In this paper, we focus on evaluating this ability with continuous learning experiments in a dynamic information environment.

4 Experimental Evaluation

In this section, we describe a new evaluation methodology for testing the ability of a user profile to represent a user's multiple interests and continuously adapt to changes in them. The methodology is used to perform simulated experiments with *virtual* users. Their advantage over experiments involving real users is that they are controlled and reproducible. They involve a collection of documents that has been manually pre-classified according to a predefined set of thematic areas, or topic categories. Instead of real users, virtual users with simulated interest in these topics provide the necessary feedback for training and evaluating user profiles. In [8], the authors proposed a variation of the evaluation methodology of TREC, which aimed at testing the ability of a single profile to represent multiple topics of interest and to adapt to a variety of changes in them. Although those experiments produced positive results, they revealed drawbacks in the methodology that were mainly due to the large number of documents in the RCV1 corpus and, consequently, the large number of relevant document's per topic. The lessons learned from the experimental work in [8] informed the specification of the improved methodology described bellow, which is a contribution of this paper.

4.1 Dataset

The experiments used the Reuters-21578 document collection[4]. The data was originally collected and labeled by Carnegie Group, Inc. and Reuters, Ltd. and

[4] Available at http://www.daviddlewis.com/resources/testcollections/reuters21578/

include 21578 news stories that appeared in Reuters newswire in 1987. The documents have been manually classified according to 135 topic categories, but for our experiments we concentrate only on the 23 topics with at least 100 relevant documents (table 1, which provide sufficient training data. This collection has been thoroughly studied and extensively used for Text Classification (TC) experiments [3], in which the collection is split into a training set and a test set. For each topic category a classifier is first trained using relevant documents from the training set and is subsequently evaluated against the test set. Here however we use the collection in a different way.

Table 1. Topics involved in the experiments and their corresponding size

topic	earn	acq	money-fx	crude	grain	trade	interest	wheat	ship	corn	dlr	oilseed
size	3987	2448	801	634	628	552	513	306	305	254	217	192
topic	money-supply	sugar	gnp	coffee	veg-oil	gold	nat-gas	soybean	bop	livestock	cpi	
size	190	184	163	145	137	135	130	120	116	114	112	

4.2 Methodology

The documents in Reuters-21578 are ordered according to publication date and their topicality changes accordingly. We exploit this ordering to test the ability of an algorithm to continuously learn, but also to forget. The 23 topics involved in the experiments are ordered according to decreasing size, i.e., the number of documents in the collection that are relevant to a topic. Table 1 includes the 23 topics and their corresponding sizes. This ordering is of no particular importance. Any other ordering could be as legitimate. As figure 1 depicts, each experimental run starts with an initially empty profile. The profile then evaluates (i.e., assigns a score to) the 21578 documents in order. The empty profile assigns a zero score to documents until it encounters a document relevant to the first of the 23 topics (i.e., earn) and then it is initialised. The initialised profile assigns a score to the remaining documents in the collection and every time it encounters a document that belongs to topic earn, it evaluates the document and then uses it as positive feedback and adapts based on it. When all 21578 documents have been evaluated, they are ranked according to decreasing relevance score and the Average Uninterpolated Precision (AUP) for each of the 23 topics is calculated on the ordered list of documents. A topic's AUP is defined as the sum of the precision–i.e., the percentage of documents relevant to that topic–at each point in the ordered list where a relevant document appears, divided by the topic's size. Figure 2 shows an example of how AUP is calculated for a topic of 8 relevant documents on an ordered list of 20 documents. Simply put, a topic's AUP is high when most of the documents at the top of the ordered list belong to the topic, which means that the profile has assigned to these documents a higher score than to the rest of the documents. After the first evaluation period, the process is reinitiated and the profile, which now represents the first of the 23 topics, starts anew to evaluate the document collection. This time, however, it uses as positive feedback documents that belong to the second of the 23 topics (i.e., acq). In other words, the profile has to forget the no longer interesting first topic

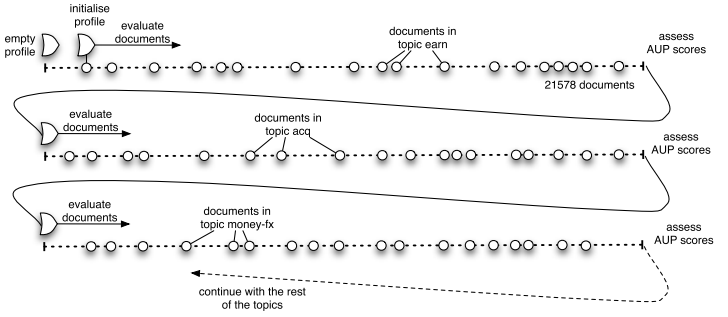


Fig. 1. Experimental Process (circles represent relevant documents)

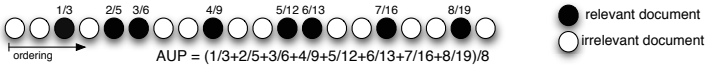


Fig. 2. AUP Calculation

(earn) and learn the new topic of interest (acq). Once all documents have been evaluated, a new set of AUP values is calculated and the process is repeated for the third topic (money-fx). The experiment finishes once all 23 topics have been used as positive feedback.

The same process has been used to perform multi-topic experiments. In particular, we performed two-topic and three-topic experiments, where profiles have to learn two and three topics in parallel, respectively. In the first case, we start, as before, with an initially empty profile, but the profile adapts whenever it encounters a document that belongs either to the first (earn), or the second (acq) of the topics. After the first evaluation period, the process is restarted, but this time the profile adapts whenever it encounters a document belonging to the second (acq) or the third topic (money-fx). The process continues until all consecutive pairs of topics have been used as positive feedback. In the case of three-topic experiments the experiment is performed for all consecutive triples of topics. We are particularly interested in these multi-topic experiments because they more accurately reflect a real situation, where users are typically interested in more than one topic in parallel.

Other aspects of the methodology include:

– Negative feedback is excluded. The space of non relevant documents is much larger than the space of relevant ones. It is very difficult for a real user to provide negative feedback to all these documents. Furthermore, the autopoietic view of the immune system holds that the latter is not antigen driven and the related computational models exclude antigens and concentrate only on the production of new antibodies by the bone marrow [16]. So here we chose to concentrate on positive feedback alone but, for our future work, we are looking into controlled and reproducible ways of choosing a subset of non relevant documents for negative feedback.

– Unlike the experiments in [8] where, in accordance with TREC, AUP calculation was performed on a list with the 1000 best scoring documents, here we deliberately used an evaluation list which includes all documents in the collection. In this way, during multi-topic experiments, we avoid a "competition" between topics for a position in the evaluation list. All documents relevant to the topics of interest are included in the evaluation list.

Overall, we propose a new approach to the evaluation of adaptive user profiles. The methodology simulates a continuous stream of documents with chronologically changing topicality and virtual users with multiple and changing interests. It also concentrates on the evaluation of a single, rather than multiple user profiles. Such a methodology has been missing from the domain of AIF. It also suggests a new way of experimenting with adaptive systems, such as AIS, in general. It tests the ability of a system for online learning (and forgetting) in a complex, multidimensional and dynamic environment. Of course, iterating over the same set of documents and suddenly switching between topics of interest, causes discontinuities rarely observed in natural environments. Nevertheless, how the system reacts to these discontinuities is an interesting test for the system's dynamics. Furthermore, one can envision an improved variation of the methodology, based on a single, long stream of documents, where user interests and the topics discussed in the documents, change naturally over time. We believe that this improved methodology requires the collection of data from the interaction of real users with a personalised information delivery system and we are working in this direction.

4.3 Baseline

The Rocchio algorithm was first proposed in [14]. It was first applied in IR for calculating an optimum query out of a set of relevant and a set of non relevant documents, but it can easily be transformed for online query modification based on a stream of relevant and non relevant documents. Since IF is commonly treated as a specialisation of IR, the online version of Rocchio algorithm (or variations of it) is often used for AIF [13,20]. It is also a popular baseline for comparative experiments [19,4].

In Rocchio algorithm, both the user profile and the incoming documents are represented as weighted keyword vectors in a common vector space, with as many dimensions as the number of unique words in the documents' vocabulary. The profile's vector is linearly moved towards the vector of an incoming document that received positive feedback and vice versa (equation 1). The coefficients α, β and γ define the relative contribution of the existing profile and of the relevant and the non relevant document respectively, on the new profile's position. Since we excluded negative feedback from our experiments, we are interested only in the first two coefficients. Their relative values define the profile's inertia. The coefficient α determines how fast the profile forgets. In other words, it determines how much previous feedback documents affect the current position of the profile's vector. It can also be seen as a decay factor that proportionally

reduces a keywords weight over time. A large α value means a small decay and vice versa. Since the decay is proportional to a keyword's weight, the latter can only converge towards zero. It is important to note that Rocchio algorithm does not include a mechanism for removing keywords from the profile, because it assumes a space with predefined dimensions. The coefficient β determines the percentage of the weight of document keywords that is added to the weight of the corresponding profile keywords. Since there is no normalisation, if β is larger than α the weights of the profile terms can keep on increasing.

$$Q_{t+1} = \begin{cases} \alpha \cdot Q_t + \beta \cdot D & \text{if } D \text{ relevant} \\ \alpha \cdot Q_t - \gamma \cdot D & \text{if } D \text{ non-relevant} \end{cases} \quad (1)$$

where:
Q_{t+1} is the new query vector
Q_t is the previous query vector
D is the vector of the feedback document

4.4 Experimental Settings

The experiments compared profiles which learn with the use of the Rocchio algorithm and profiles based on Nootropia. In both cases, incoming documents were pre-processed in the same way. After stop word removal and stemming using Porter's algorithm the words in each document are weighted using Relative Document Frequency (RelDF)[5], which calculates the difference of the probability of a word appearing in relevant documents, minus the probability of the word appearing in a random document from a general baseline collection. As already discussed, our baseline collection comprised all documents in Reuters-21578 and RCV1 corpus. The larger and broader the baseline collection is the more accurate is the estimation of the second probability. Words with weight over 0.03 were extracted from each document[6].

Unlike traditional approaches which treat documents as bag of words, both types of profiles deploy a sliding window approach with a window of size[7] 10. For Rocchio, a relevance score is calculated for each position of the window as the inner product between the profile's vector and the window's vector. In the case of Nootropia, the profile uses the spreading activation process described in [12] to calculate a relevance score for each position of the window. In both cases, a document's overall score is calculated as the aggregate window score normalised to the number of different window positions.

We experimented with different profile configurations. We tested Rocchio profiles with the following values for the pair of coefficients (α, β): (0.75, 0.25), (0.85, 0.15), (0.90, 0.25), (0.95, 0.25) and (0.99, 0.25). We were initially surprised to find out during preliminary experiments that the value of β does not have an

[5] RelDF has produced very good results in comparative experiments [11] and is easily modified for online weighting of documents.
[6] The value 0.03 was defined by systematic preliminary experimentation.
[7] The best window size has been experimentally defined in [12].

Table 2. AUP scores for single-topic experiments

topic	Rocchio	Nootropia	difference %	topic	Rocchio	Nootropia	difference %
earn	0.69781	0.73191	4.89	money-supply	0.21173	0.19678	-7.06
acq	0.42519	0.35298	-16.98	sugar	0.51879	0.61614	18.76
money-fx	0.43697	0.55646	27.35	gnp	0.45296	0.29064	-35.84
crude	0.64611	0.70712	9.44	coffee	0.70078	0.82323	17.47
grain	0.52072	0.49181	-5.55	veg-oil	0.29195	0.28732	-1.59
trade	0.56719	0.55304	-2.49	gold	0.63249	0.74839	18.32
interest	0.36168	0.46944	29.79	nat-gas	0.3467	0.3364	-2.97
wheat	0.50732	0.49092	-3.23	soybean	0.28238	0.28895	2.33
ship	0.44172	0.28172	-36.22	bop	0.33406	0.32373	-3.09
corn	0.31216	0.30038	-3.77	livestock	0.07644	0.03018	-60.52
dlr	0.37075	0.46118	24.39	cpi	0.25964	0.24591	-5.29
oilseed	0.20216	0.17715	-12.37				
						average	**-1.92**
						st. deviation	**21.44**

effect on the profile's performance, but the reason is that β does not change the relative weight of the words in the document. β can only have an effect on the profile's learning behaviour in relation to the γ coefficient, which is excluded from our experiments. Nootropia's adaptation has only one coefficient, which is the decay ratio δ. We experimented with δ values equal to 0.90, 0.95, 0.98 and 0.99. Rocchio algorithm achieved the best results for $\alpha = 0.95$ and $\beta = 0.25$. Nootropia achieved the best results for $\delta = 0.99$. Bellow, we concentrate and compare the results for these parameter configurations.

4.5 Experimental Results

Tables 2 to 4 present the AUP scores of the topic, or the topics, of interest at the end of the corresponding evaluation period. For the single-topic experiments, table 2 presents the topic of interest at each evaluation period, the corresponding AUP values for Rocchio and Nootropia, respectively, and the difference between the latter and the former in percentage. Similarly, tables 3 and 4 show the AUP values for the two-topic and the three-topic experiments, respectively. For these multi-topic experiments the tables present, for each topic combination of interest (e.g., earn:acq:money-fx), the AUP values of each constituent topics. These values are accompanied by a combined AUP score, which is calculated based on the aggregate set of relevant documents for all constituent topics. Note that due to the way AUP is calculated, the combined value is not the sum of the individual AUPs. The difference between Nootropia and Rocchio is calculated based on these combined AUP values.

In the single-topic experiments (table 2), Rocchio algorithm produces better AUP scores than Nootropia for 14 out of 23 topics. For the 9 topics for which Nootorpia is better, it achieves an average increase in AUP score of 16.97%. Overall, Nootropia is worse by 1.92%. On the contrary, the results for the multi-topic experiments are positive. For two topics (table 3), Nootropia is better for 28

Table 3. AUP scores for two-topic experiments

	Rocchio			Nootropia			
topics (1^{st}:2^{nd})	1^{st} topic	2^{nd} topic	combined	1^{st} topic	2^{nd} topic	combined	difference %
earn:acq	0.56351	0.162	0.6495	0.6816	0.13942	0.68823	5.96
acq:money-fx	0.24113	0.19463	0.37824	0.22318	0.21644	0.37571	-0.67
money-fx:crude	0.23852	0.22137	0.45529	0.25833	0.31737	0.56071	23.15
crude:grain	0.37039	0.16298	0.48952	0.36294	0.20648	0.54818	11.98
grain:trade	0.16114	0.39291	0.47554	0.19244	0.33429	0.48497	1.98
trade:interest	0.34382	0.11714	0.41337	0.22016	0.27659	0.46626	12.79
interest:wheat	0.25061	0.08166	0.31694	0.28219	0.13282	0.40759	28.60
wheat:ship	0.30307	0.1373	0.40431	0.36914	0.12792	0.44074	9.01
ship:corn	0.16537	0.17925	0.32995	0.1721	0.1899	0.34593	4.84
corn:dlr	0.11104	0.20237	0.2847	0.10053	0.34037	0.34301	20.48
dlr:oilseed	0.2361	0.05818	0.25416	0.34982	0.05951	0.32541	28.03
oilseed:money-supply	0.07141	0.09243	0.15338	0.05564	0.12091	0.15949	3.98
money-supply:sugar	0.08137	0.2207	0.26916	0.09132	0.38794	0.40195	49.33
sugar:gnp	0.17947	0.19681	0.36314	0.62478	0.08296	0.54226	49.33
gnp:coffee	0.18153	0.32469	0.45288	0.12625	0.70485	0.60162	32.84
coffee:veg-oil	0.34844	0.12714	0.42828	0.58577	0.16178	0.62112	45.03
veg-oil:gold	0.17028	0.15437	0.31528	0.15849	0.43553	0.51631	63.76
gold:nat-gas	0.27376	0.20134	0.4597	0.63245	0.15263	0.63904	39.01
nat-gas:soybean	0.15897	0.09034	0.23615	0.18858	0.14792	0.32175	36.25
soybean:bop	0.07538	0.16789	0.22285	0.03871	0.24746	0.20284	-8.98
bop:livestock	0.26222	0.01662	0.17142	0.33159	0.01272	0.19378	13.04
livestock:cpi	0.02721	0.17289	0.1445	0.02383	0.09914	0.09744	-32.57
						average	**19.87**
						st. deviation	**22.39**

out of 46 individual topics and for 20 out of 23 topic combinations and it achieves an overall increase in AUP score of 19.87%. For three topics (table 4), Nootropia is better for 39 out of 69 individual topics and for 20 out of 23 topic combinations and it achieves an overall increase in AUP score of 22.64%. Nootropia is clearly better than Rocchio, when the user profile has to represent multiple and changing topics.

The results for the single-topic experiments are not conclusive. Nootropia is overall worse than Rocchio by only a minor percentage (1.92%) while the standard deviation is much larger (21.44). Nevertheless, in comparison to the multi-topic experiments, where Nootropia is clearly better than Rocchio, the results for the single-topic experiments are not that good. We can not yet confidently justify this difference in performance. One possible reason is that due to the linkage between profile terms, Nootropia can store more information and can thus exploit better the additional positive feedback provided in multi-topic experiments. In [10], we argued and supported experimentally that Nootropia identifies those term combinations that are relevant to the user's interests and this allows a more effective representation of multiple topics in parallel. But the large differences between Nootropia and Rocchio in the current experiments is not only due to the positive effect of links on profile representation and document evaluation. More recent preliminary experiments, which are not reported here due to space limitations, show that even if we ignore links during document evaluation, Nootropia performs better than Rocchio. This means that Nootropia's adaptation, which involves a competition between linked profile terms, is better than Rocchio algorithm in updating the weight of terms (dynamics) and in defining the necessary profile repertoire (metadynamics).

Table 4. AUP scores for three-topic experiments

	Rocchio				Nootropia				
topics (1^{st}:2^{nd}:3^{rd})	1^{st}	2^{nd}	3^{rd}	combined	1^{st}	2^{nd}	3^{rd}	combined	dif. %
earn:acq:money-fx	0.4312	0.1353	0.0828	0.5873	0.6200	0.1226	0.0571	0.6661	13.42
acq:money-fx:crude	0.1970	0.1480	0.1052	0.3918	0.1938	0.1663	0.1279	0.4188	6.87
money-fx:crude:grain	0.1837	0.1714	0.0903	0.4180	0.1935	0.1978	0.1233	0.4971	18.93
crude:grain:trade	0.1860	0.1133	0.2484	0.4833	0.2032	0.1441	0.1923	0.5097	5.47
grain:trade:interest	0.1035	0.3006	0.0873	0.4010	0.1359	0.2035	0.1486	0.4571	14.00
trade:interest:wheat	0.3243	0.1001	0.0439	0.3904	0.2377	0.1764	0.0665	0.4531	16.04
interest:wheat:ship	0.1909	0.0798	0.0561	0.3070	0.2678	0.1121	0.0525	0.3940	28.33
wheat:ship:corn	0.2921	0.1091	0.1765	0.4370	0.3426	0.1077	0.1774	0.4740	8.45
ship:corn:dlr	0.0901	0.0853	0.1524	0.2920	0.0837	0.0860	0.2635	0.3402	16.49
corn:dlr:oilseed	0.1271	0.1639	0.0725	0.2914	0.1219	0.2592	0.0707	0.3447	18.31
dlr:oilseed:money-supply	0.1818	0.0342	0.0445	0.2031	0.2366	0.0375	0.0568	0.2602	28.15
oilseed:money-supply:sugar	0.0765	0.0438	0.1431	0.2314	0.0694	0.0571	0.2050	0.2858	23.50
money-supply:sugar:gnp	0.0602	0.0641	0.1684	0.2465	0.0843	0.2227	0.1048	0.3640	47.62
sugar:gnp:coffee	0.1002	0.1150	0.2349	0.3832	0.2508	0.0721	0.4629	0.5917	54.39
gnp:coffee:veg-oil	0.1038	0.2090	0.0746	0.3422	0.0913	0.3882	0.0948	0.4548	32.93
coffee:veg-oil:gold	0.2749	0.0990	0.0566	0.3543	0.2840	0.0925	0.2819	0.5784	63.24
veg-oil:gold:nat-gas	0.1256	0.0861	0.1160	0.3140	0.1500	0.2668	0.1027	0.4735	50.77
gold:nat-gas:soybean	0.1171	0.1203	0.0725	0.2913	0.2977	0.0951	0.0862	0.4124	41.56
nat-gas:soybean:bop	0.0708	0.0513	0.0927	0.1980	0.0426	0.0304	0.2016	0.1846	-6.78
soybean:bop:livestock	0.0680	0.1304	0.0193	0.1692	0.0312	0.2196	0.0154	0.1608	-4.94
bop:livestock:cpi	0.2148	0.0144	0.0655	0.1941	0.2424	0.0122	0.0524	0.1917	-1.22
								average	**22.64**
								st. deviation	**19.76**

Figures 3 and 4 show the number of terms and links in Nootropia, and the average term and link weights, respectively, throughout the two-topic experiments. Figure 5 shows the number of profile terms and the average term weight when Rocchio algorithm is used. All three graphs depict the codes of relevant documents in the order presented to the profile, hence the different evaluation periods are visualised. The graphs show that Nootropia is a complex and dynamic system capable of self-regulating its size and connectivity. The number of terms and links fluctuates, but does not escalate. The same is true for the average term and link weight. The fluctuations revolve around smoothly changing averages and become more intense towards the final evaluation periods which involve topics with a small number of relevant documents. It is also interesting that one can observe spikes in the values of the average term and link weight, which seem to coincide with the end of an evaluation period and the switch to a new pair of interesting topics. These spikes are indicative of Nootropia's reaction to the existing discontinuities in the underlying data. We believe these are very interesting observations and require further examination and elucidation, but this is beyond the scope of this paper.

The behaviour of Rocchio algorithm is more anticipated. As figure 5 depicts, the number of profile terms with weight over zero increases with decreasing pace. Rocchio algorithm does not have a mechanism for removing features from the profile. It assumes the existence of a common vector space, where both profiles and documents are expressed as weighted keyword vectors. The dimensionality of the space is predefined and does not change over time. Therefore, it does not make sense to completely remove profile features. As relevant documents with new words are encountered, the number of features in the profile's keyword

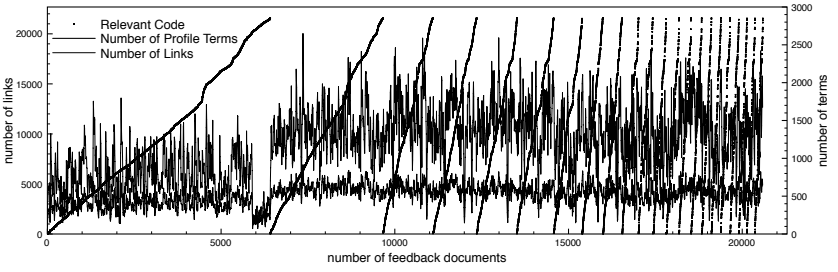


Fig. 3. The number of links and terms in Nootropia through out the learning process

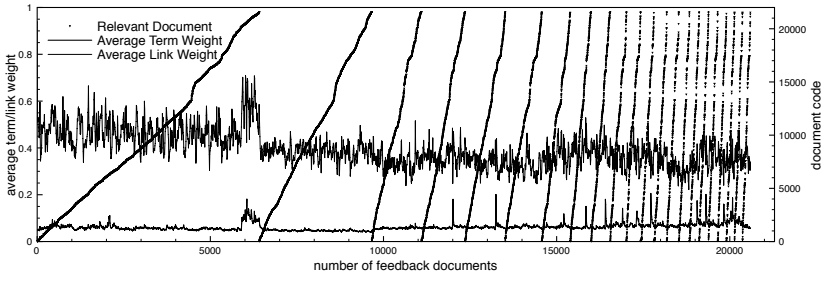


Fig. 4. Nootropia's average term and link weights through out the learning process

vector, with weight over zero, progressively increases. The number of profile terms will keep on increasing until the complete vocabulary of the underlying document collection is attained. Note that, according to figure 5, the vector space for the current experiment should have more than 21000 dimensions[8]. Existing experimental methodologies deploy spaces with a much smaller dimensionality. In [4] for instance, the authors experiment with just 379 features. Note also that, despite the large number of available words, Nootropia dynamically maintains no more than 800 words in the profile's repertoire (figure 3).

Overall, the results of the experiments have been satisfactory. They show Nootropia's ability to define and maintain a representation of a user's multiple and changing interests in a dynamic environment. This is not achieved in a linear way, but with an autopoietic process of self-organisation. The profile's immune network reacts structurally in response to user feedback and self-regulates its size and connectivity in such a way that it continuously adapts to the interest changes of the "host" user. One may argue that it becomes structurally coupled to its environment. The results are also complementary to previous experimental work, indicating that it is the network structure which allows the profile to store the additional information required for representing multiple topics of interest [11,12]. Without doubt, further comparative experiments with other algorithms are required and should be accompanied by elaborate statistical analysis.

[8] The number of words in the Rocchio profile at the end of the experiment is 21423.

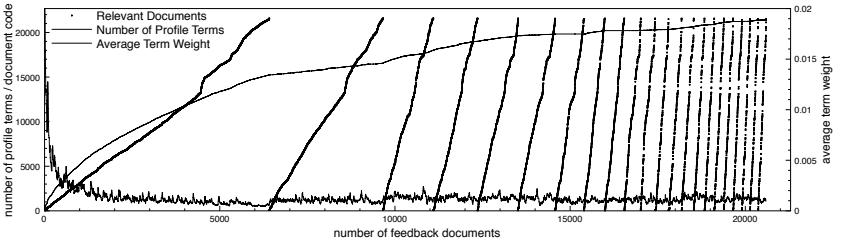


Fig. 5. Rocchio's number of terms and average term weight during the experiments

5 Summary and Future Work

Profile adaptation is a complex and dynamic problem. A user profile has to be able to represent a user's multiple interests and adapt to changes in them, while the information environment itself also changes over time. According to Autopoietic Theory, the immune network faces a similar problem. It has to define and preserve the developing organism's identity in a changing environment. Nootropia is a user profiling model, inspired by the autopoietic immune network. It uses a network of terms to represent the user's interests and a process of self-organisation, involving dynamics and metadynamics, to regulate the network in response to user feedback. The paper presented a new, improved version of the model, and proposed a methodology for setting up experiments that test the ability of profiles to continuously learn, but also to forget, in the presence of drifts in the interests of virtual users. We used the methodology to perform comparative experiments between Nootropia and the popular Rocchio algorithm. The results show that, for multiple topics, Nootropia achieves an increase in performance of more than 19%. It learns and forgets while it controls dynamically its size and connectivity. The proposed methodology provides an interesting experimental setting for performing a variety of experiments. Future plans include comparing Nootropia to evolutionary and other algorithms, but also experiments with further variations of Nootropia. Improving Nootropia is part of an ongoing effort to develop personalised information services on the Web.

References

1. Bezerra, G.B., Barra, T.V., Ferreira, H.M., Knidel, H., de Castro, L.N., Zuben, F.J.V.: An immunological filter for spam. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 446–458. Springer, Heidelberg (2006)
2. Ciesielski, K., Wierzchoń, S.R., Klopotek, M.A.: An immune network for contextual text data clustering. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 432–445. Springer, Heidelberg (2006)
3. Debole, F., Sebastiani, F.: An analysis of the relative hardness of Reuters-21578 subsets. Journal of the American Society for Information Science and Technology 56(6), 584–596 (2004)

4. Kassab, R., Lamirel, J.-C.: An innovative approach to intelligent information filtering. In: SAC 2006: Proceedings of the, ACM Symposium on Applied Computing, pp. 1089–1093. ACM, New York (2006)
5. Moukas, A., Zacharia, G., Maes, P.: Amalthaea and Histos: Multiagent systems for WWW sites and reputation recommendations. In: Klusch, M. (ed.) Intelligent Information Agents: Agent-Based Information Discovery and Management on the Internet, pp. 293–322. Springer, Heidelberg (1999)
6. Nanas, N., De Roeck, A.: Multimodal dynamic optimisation: from evolutionary algorithms to artificial immune systems. In: de Castro, L.N., Von Zuben, F.J., Knidel, H. (eds.) ICARIS 2007. LNCS, vol. 4628, pp. 13–24. Springer, Heidelberg (2007)
7. Nanas, N., De Roeck, A.: A review of evolutionary and immune inspired information filtering. Natural Computing (2007)
8. Nanas, N., De Roeck, A.: Autopoiesis, the immune system and adaptive information filtering. Natural Computing (2008)
9. Nanas, N., Uren, V., De Roeck, A.: Nootropia: a user profiling model based on a self-organising term network. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 146–160. Springer, Heidelberg (2004)
10. Nanas, N., Uren, V., De Roeck, A., Domingue, J.: Multi-topic information filtering with a single user profile. In: 3rd Hellenic Conference on Artificial Intelligence, pp. 400–409 (2004)
11. Nanas, N., Uren, V., Roeck, A.D.: A comparative evaluation of term weighting methods in information filtering. In: 4th International Workshop on Natural Language and Information Systems (NLIS 2004), pp. 13–17 (2004)
12. Nanas, N., Vavalis, M.: A "bag" or a "window" of words for information filtering. In: 5th Hellenic Conference on AI, pp. 182–193. Springer, Heidelberg (2008)
13. Pon, R.K., Cárdenas, A.F., Buttler, D.J.: Online selection of parameters in the rocchio algorithm for identifying interesting news articles. In: WIDM 2008: Proceeding of the 10th ACM Workshop on Web Information and Data Management, pp. 141–148. ACM, New York (2008)
14. Rocchio, J.: Relevance Feedback in Information Retrieval, ch. 14, pp. 313–323. Prentice-Hall Inc., Englewood Cliffs (1971)
15. Seo, Y., Zhang, B.: A reinforcement learning agent for personalized information filtering. In: Intelligent User Interfaces, New Orleans, LA, pp. 248–251 (2000)
16. Stewart, J., Varela, F.J.: Morphogenesis in shape-space, elementary meta-dynamics in a model of the immune network. Journal of Theoretical Biology 153, 477–498 (1991)
17. Varela, F.J., Coutinho, A.: Second generation immune network. Immunology Today 12(5), 159–166 (1991)
18. Widyantoro, D.H., Ioerger, T.R., Yen, J.: An adaptive algorithm for learning changes in user interests. In: ACM/CIKM 1999 Conference on Information and Knowledge Management, Kansas City, MO, pp. 405–412 (1999)
19. Yang, Y., Yoo, S., Zhang, J., Kisiel, B.: Robustness of adaptive filtering methods in a cross-benchmark evaluation. In: SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 98–105. ACM, New York (2005)
20. Zhang, Y.: Using bayesian priors to combine classifiers for adaptive filtering. In: SIGIR 2004: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 345–352. ACM, New York (2004)

Unsupervised Structure Damage Classification Based on the Data Clustering and Artificial Immune Pattern Recognition

Bo Chen[1] and Chuanzhi Zang[2]

[1] Department of Mechanical Engineering – Engineering Mechanics/Department of Electrical & Computer Engineering, Michigan Technological University, Houghton, MI, USA
bochen@mtu.edu
[2] Department of Mechanical Engineering – Engineering Mechanics, Michigan Technological University, Houghton, MI, USA
Shenyang Institute of Automation, Chinese Academy of Science, Shenyang, China
czang@mtu.edu

Abstract. This paper presents an unsupervised structure damage classification algorithm based on the data clustering technique and the artificial immune pattern recognition. The presented method uses time series measurement of a structure's dynamic response to extract damage-sensitive features for the structure damage classification. The Data Clustering (DC) technique is employed to cluster training data to a specified number of clusters and generate the initial memory cell set. The Artificial Immune Pattern Recognition (AIPR) algorithms are integrated with the data clustering algorithms to provide a mechanism for the evolution of memory cells. The combined DC-AIPR method has been tested using a benchmark structure. The test results show the feasibility of using the DC-AIPR method for the unsupervised structure damage classification.

Keywords: structural health monitoring, unsupervised structure damage classification, data clustering, artificial immune pattern recognition.

1 Introduction

Damage diagnosis is one of the major tasks of the structural health monitoring (SHM) systems. The SHM process involves the observation of a structure's dynamic response measurements from a group of sensors, the extraction of damage-sensitive features from these measurements, and analysis of these features to determine the current state of the structure [1]. Traditional SHM systems are wired data acquisition systems to collect distributed sensor data to a central data processing station. The practical use of wired SHM systems is limited due to high instrument and installation costs [2]. The wireless sensor network approach is emerging for the effective SHM since it allows dense sensing through many in-expensive sensor nodes and is easy for deployment and maintenance. While sensor network approach presents a number of advantages, SHM sensor network systems currently face many challenges. Major challenges in SHM are: 1) How can we provide sustainable long-term monitoring and

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 206–219, 2009.

control in an autonomous manner? 2) How can we detect and identify structure damage in an active way? 3) Can we develop adaptable approaches to SHM which are able to dynamically adapt to changing monitoring conditions? 4) How can we establish unsupervised damage diagnosis methodology?

The artificial immune system (AIS) approach provides an appropriate solution to address these challenges. The immune system is a rapid and effective defense mechanism for a given host against infections [3]. The novel characteristics of the immune system have inspired the development of artificial immune systems for various applications [4, 5]. The major application areas include data mining [6], pattern recognition [7, 8], and fault diagnosis [9, 10]. Due to the similarities of the human immune system and the SHM systems, the immune system model could be used as the basis for SHM strategies. This approach is well suited to this problem because: 1) The immune system is an autonomous system. The AIS-based SHM systems can automatically manage structure monitoring tasks by dynamically generating and distributing the mobile monitoring agents; 2) The immune system is an active system. Lymphocytes can circulate between lymphoid and non-lymphoid tissues. The concept of active dispatching mobile monitoring agents (mimicking B-cells) helps the distribution of specialized monitoring agents to the sites where they are needed; 3) The immune system is an adaptive system. The amount and type of molecules of the immune system can adapt themselves to the antigenic challenges via clonal selection. Selected cells with long life spans (memory cells) have faster and more effective responses to the same (or a slightly different) antigenic challenge [11]. The adaptive mechanism of the natural immune system has great value in SHM sensor networks. The selective generation of mobile monitoring agents is essential for producing large enough amounts of specialized mobile monitoring agents in resource-constrained sensor networks [12]; 4) The immune learning and memory mechanisms help the development of unsupervised damage detection and classification, which is desirable in SHM.

To facilitate the unsupervised structure damage diagnosis in the SHM systems, this paper presents an unsupervised structure damage classification algorithm based on the Data Clustering technique and the Artificial Immune Pattern Recognition (DC-AIPR). The Fuzzy Clustering (FC) algorithm is employed to generate initial memory cells for each damage pattern. These initial memory cells are then evolved by the stimulation of training data to improve the quality of memory cells to represent damage patterns. The presented unsupervised structure damage classification algorithm has been tested using a benchmark structure [13] proposed by the IASC-ASCE (International Association for Structural Control - American Society of Civil Engineers) Structural Health Monitoring Task Group. The test results show the feasibility of using the DC-AIPR method for the unsupervised structure damage classification.

The rest of the paper is organized as follows. Section 2 gives an overview of the DC-AIPR, an unsupervised structure damage classification algorithm based on the data clustering technique and the artificial immune pattern recognition. Section 3 introduces damage-sensitive feature extraction from structure's dynamic response measurements. Section 4 presents the algorithm design of the DC-AIPR. Section 5 shows the validation results of the DC-AIPR method using a benchmark structure. Section 6 discusses the performance of the unsupervised damage classification. Section 7 concludes the presented work.

2 Overview of the Combined Data Clustering and Artificial Immune Pattern Recognition (DC-AIPR) Approach

Structure damage diagnosis, in some cases, must be performed in an unsupervised learning mode in SHM systems. For unsupervised structure damage classification, the assumption is that the class labels (damage patterns) of training data are not available. In this case, appropriate data clustering algorithms are good candidate to cluster "similar" feature vectors together [14]. The presented DC-AIPR unsupervised classifier employs fuzzy clustering algorithms to find the representative for each class. The representative for each class generated by the fuzzy clustering algorithms, however, includes limited information. For example, the fuzzy clustering algorithms use one point in a multidimensional space to represent each cluster for the compact data. To obtain more informative cluster representatives and provide the evolution capability, the artificial immune pattern recognition method is employed to integrate with the data clustering techniques for the unsupervised structure damage classification. The representative for each class generated by the fuzzy clustering algorithm is used as initial memory cell for each class. The initial memory cells are also used to classify the training data based on the k-nearest neighbor criterion. The classified training data are then used to generate a new memory cell set based on the artificial immune pattern recognition algorithm.

3 Damage-Sensitive Feature Extraction

The feature selection is critical to the success of the damage classification. Feature selection is the process to identify the measurable quantities that make damage patterns distinct from each other. For structure damage diagnosis based on the time series measurement of a structure's dynamic response, the measurement data need to be standardized to reduce the environmental effects. In the DC-AIPR algorithm implementation, following steps are designed to process raw data and extract damage-sensitive feature vectors.

First, the time series measurement data are normalized using the mean and standard deviation. Let matrix $Z = (z_{ij})_{m\times n}$ denote the time series of measurement data, where each row is corresponding to the n number of data generated by one sensor and each column is the measurement data collected by the m sensors at a given time. Let $\bar{z}_i = (z_{i1}, z_{i2}, ..., z_{in}),\ i = 1,2,...,n$ denote the ith row of the matrix Z, which is the measurement data of the ith sensor. The standardized measurement data $Y = (y_{ij})_{m\times n}$ can be calculated by $y_{ij} = \dfrac{z_{ij} - \mu_i}{\sigma_i}\ j = 1,2,...,n$, where y_{ij} is the standardized value of the corresponding value of z_{ij}, μ_i and σ_i are the mean and standard deviation of the time series $\bar{z}_i$.

Second, time series measurement data sets from multiple sensors are reduced to lower dimensions by the Principal Component Analysis (PCA) [15] method for extracting a feature vector for a local area. In our implementation, the multiple data sets

from *m* number of sensors are compressed into one data set. It means that all the sensor measurement data are projected onto the principal component that has the biggest eigenvalue. Let Ψ denote the $m \times m$ covariance matrix of the standardized time series signals *Y*. The matrix Ψ can be calculated by $\Psi = \frac{1}{n-1} YY^T$. Let λ_i and $\vec{v}_i$ denote the *i*th eigenvalue and eigenvector of matrix Ψ respectively. So, Ψ, λ_i, and $\vec{v}_i$ satisfy $\Psi \vec{v}_i = \lambda_i \vec{v}_i$, where eigenvector $\vec{v}_i$ is called the principal component. Let $\vec{v}_1$ denote the vector that is corresponding to the biggest eigenvalue. The relationship between the compressed data $\vec{x} = (x_1, x_2, ..., x_n), \vec{v}_1$ and *Y* is $\vec{x} = \vec{v}_1^T Y$.

Third, the feature vector for a local area is extracted from the compressed time series. The auto regressive (AR) algorithm is chosen to model the compressed time series data. Each compressed time series *x* is fitted into an AR model of order *p* as shown in equation (1).

$$x_k = \sum_{i=1}^{p} a_i x_{k-i} + r_k \quad k = p+1, ..., n \tag{1}$$

where the r_k is the residual between the measurement data and the AR model value. The order of the AR model is chosen based on the Akaike's Information Criterion (AIC). An AIC is a measure of the goodness of fit of an estimated statistical model. Given a data set, the model having the lowest AIC is the best model. The vector $\alpha = (a_1, a_2, ..., a_p)^T \in R^p$, is selected as the ***feature vector*** of the time series. The feature vector is calculated using the Least Square (LS) method. Rewrite Equation (1) in following format:

$$A\alpha = b \tag{2}$$

where

$$A = \begin{bmatrix} x_p & x_{p-1} & \cdots & x_1 \\ x_{p+1} & x_p & \cdots & x_1 \\ \cdots & \cdots & \ddots & \cdots \\ x_{n-1} & x_{n-2} & \cdots & x_{n-p-1} \end{bmatrix}, \quad b = \begin{bmatrix} x_{p+1} \\ x_{p+2} \\ \vdots \\ x_n \end{bmatrix} \tag{3}$$

The feature vector α can be calculated as follows:

$$\alpha = (A^T A)^{-1} A^T b \tag{4}$$

The effectiveness of the AR-model-based feature vectors is tested using the experimental data of the benchmark structure [13] proposed by the IASC-ASCE SHM Task Group. The feature vectors of four damage patterns and the normal pattern are visualized using the Sammon nonlinear mapping algorithm [16] as shown in Figure 1. Although some overlapping among different patterns exists, the AR-model-based feature vectors are able to distinguish these five data patterns to a certain extent.

4 The DC-AIPR Approach for the Unsupervised Structure Damage Classification

4.1 Generate Initial Memory Cells Using Fuzzy Clustering Algorithm

For unsupervised structure damage classification, the class label of the training data (feature vectors of the structure measurement data) is not available. To generate the initial memory cell set for the AIPR algorithm, the fuzzy-ISODATA algorithm is employed. Since the damage-sensitive feature vectors are compact clusters as shown in Figure 1, a point representative is used to represent each cluster. A fuzzy *m*-clustering of $X = \{\alpha_1, \alpha_2, \cdots \alpha_N\}$ is defined by a set of functions $u_j : X \to A, j = 1,2,...,m$, where $A = [0,1]$, and $\alpha_i \in R^p$, $i = 1,2,...,N$ is the feature vector of the training data in *p*-dimensional real value space. Let $\theta_j \in R^p$ denote the parameterized representative of the *j*th cluster, $\theta \equiv \left[\theta_1^T, \theta_2^T, ..., \theta_m^T\right]^T$, U is an $N \times m$ matrix whose (i, j) element equals $u_j(\alpha_i)$, $d(\alpha_i, \theta_j)$ is the dissimilarity between α_i and θ_j, and $q(>1)$ is a parameter called a *fuzzifier*. The fuzzy clustering algorithms are derived by minimizing a cost function in equation (5) with respect to θ and U.

$$J(\theta, U) = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij}^q d(\alpha_i, \theta_j) \tag{5}$$

The results of the fuzzy *k*-means are the point representatives of clusters. These points are used as initial memory cells for *m* clusters. Once the cluster representatives are determined, they are used to classify the training data by using the nearest neighbor criterion. Given a training data, the distance to the cluster representatives are calculated. The training data is classified to the cluster with whose cluster representative the training data has the shortest distance. The classified training data are then used in the memory cell evolution process for improving the quality of the memory cell set.

4.2 Evolve Memory Cell Set Using the Artificial Immune Pattern Recognition (AIPR) Algorithms

Since the initial memory cell set generated by the fuzzy clustering algorithm only has one memory cell for each cluster, the AIPR method [17] is used to provide a mechanism to evolve the memory cell set. The AIPR algorithm in [17] is based on the CLONALG algorithm in [18] and the AIRS in [7]. The antibody set evolution is similar to CLONALG. The memory cell set update, however, is specifically designed to obtain better representatives for each damage pattern. For example, the memory cell replacement threshold defined in [17] is effective to improve the classification success rate. The evolution of the memory cell set includes two sub-processes: the evolution of the antibody set and the update of the memory cell set. The flow chart of the memory cell set evolution process is shown in Figure 2. The training data clustered by the fuzzy clustering algorithm and k-nearest neighbor criterion are used to stimulate this

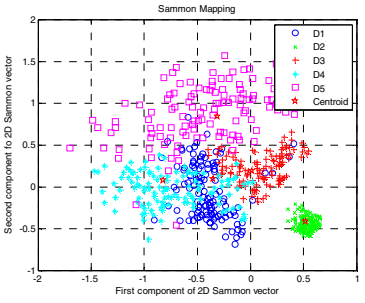


Fig. 1. The AR-model-based feature vectors of data patterns

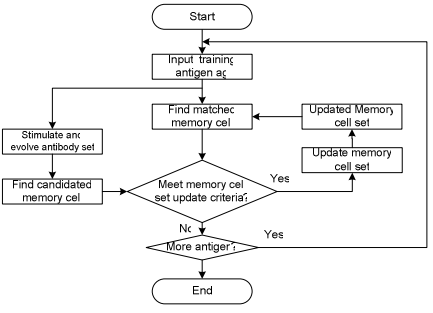


Fig. 2. The evolution of memory cells

process. The initial antibody set is generated by the random selection of antibodies from the classified training data.

Evolution of the antibody set using antigenic stimulation. The stimulation of the antibody set by an invading antigen (a training data) will cause the evolution of the antibody set. The description of the antibody set evolution algorithm is given in Table 1. For a training antigen ag, the affinity between the antigen and each antibody ab that is in the same class as the antigen is calculated. Let $ab.f = \beta = (\beta_1, \beta_2, ..., \beta_p)^T$ and $ag.f = \gamma = (\gamma_1, \gamma_2, ..., \gamma_p)^T$ denote the feature vectors of an antibody ab and the antigen ag respectively. The affinity between an antibody and the antigen is defined as

$$aff(ab, ag) = 1 - \frac{1}{2} dist(\beta, \gamma) \tag{6}$$

where $dist(\beta, \gamma)$ is the Euclidian distance between the feature vectors of β and γ. The probability that an antibody *ab* is cloned depends on its affinity with the antigen. The number of the cloned antibodies, *CloneNumber*, depends on the clonal rate *CR* and the clonal value *CV*. The *CR* is an integer value used to control the number of antibody clones allowed for the activated B-cell. The *CV* is a value that measures the response of a B-cell to an antigen. According to the natural immune system, the higher the affinity, the larger the number of antibodies is cloned. We choose the clonal value being equal to the affinity value. The *CloneNumber* is then calculated by the equation (7).

$$CloneNumber = round(CR * CV) = round(CR * aff(ab, ag)) \tag{7}$$

where $round(\bullet)$ is an operator that rounds its value to the closest integer.

The cloned antibodies undergo a maturation process that increases the diversity of the antibody set. The mutation is performed by mutating the feature vectors of the cloned antibodies as shown in equation (8).

$$ab_{mutated}.f = ab.f + MV \times \phi \tag{8}$$

where $ab_{mutated}$ is the mutated antibody and MV is the mutation value. Typically, the higher the affinity is, the smaller the mutation value. In our design, the mutation value MV is defined as $MV = 1 - CV$. In equation (8), the vector $\phi = (\phi_1, \phi_2, ..., \phi_p)^T$ is a randomly generated vector whose dimension is same as that of the feature vector. Each element ϕ_i is a normal random variable defined by $\phi_i \sim N(0, \sigma^2)$, where $N(0, \sigma^2)$ is a normal random variable with the standard deviation of σ.

The mutated antibodies are added into the antibody subset to which the *ag* belongs. Since the maximum number of each antibody subset is limited to a predefined threshold, *MaxABN*, the resulting antibody subset is sorted in a descending order according to the affinity values of the antibodies with the given antigen. The top *MaxABN* number of antibodies is selected to form the evolved antibody set. The rest of antibodies are discarded. The antibody with the highest affinity is chosen as the candidate memory cell $MC_{candidate}$ for the updating of memory cell set.

Table 1. The description of the antibody set evolution algorithm

Algorithm 1. Antibody set evolution
Begin
Input an antigen *ag*;
While (there is more antibody *ab* which is in the same class as *ag*) **do**
Clone antibody *ab* based on the affinity with the *ag*;
Mutate the cloned antibodies;
Keep the mutated antibodies staying within the unit hyper-sphere;
Form a new antibody set using top MaxABN number of antibodies;
End while
Select the highest affinity antibody as the candidate memory cell;
End

Update memory cell set. The candidate memory cell generated in the antibody set evolution process is used to update the memory cell set to enhance the representative quality of memory cells for each pattern. The description of the memory cell set update algorithm is given in Table 2. The memory cell update occurs in the following scenarios. First, when the root mean square distance, *rms*, between the candidate memory cell and the memory cells in the same class is greater than a specified threshold value *MCIT* (Memory Cell Injection Threshold), the candidate memory cell is injected into this class of memory cells. Let $ag.c$ denote the class label of the antigen ag, let $MCS_{ag.c}$ denote the memory cell subset with the same class as the given antigen ag, and let $|MCS_{ag.c}|$ denote the total number of the memory cells in the subset $MCS_{ag.c}$. The rms is defined by the equation (9).

$$rms = RMS\left(dist_1, dist_2, ...dist_{\left|MCS_{ag.c}\right|} \right) = \frac{1}{\sqrt{\left|MCS_{ag.c}\right|}} \sqrt{\sum_{i=1}^{\left|MCS_{ag.c}\right|} dist_i^2} \tag{9}$$

where $dist_i = dist\left(mc_i, MC_{candidated}\right)$, $mc_i \in MCS_{ag.c}$, and $i = 1,2,...,\left|MCS_{ag.c}\right|$. If the *rms* is greater than the *MCIT*, the candidate memory cell is added into the memory cell subset $MCS_{ag.c}$.

In the second case (the *rms* is less than or equal to *MCIT*), the candidate memory cell compares with the matched memory cell. The matched memory cell is the memory cell that has the highest affinity with the given antigen in the same class. To find the matched memory cell, the affinity values of the training antigen with the memory cells in the same class are calculated. The memory cell that has the highest affinity with the given antigen *ag* is chosen as the matched memory cell, which is denoted by $MC_{matched}$. When the affinity between $MC_{candidate}$ and the given antigen *ag* is greater than the affinity between $MC_{matched}$ and antigen *ag,* the candidate memory cell replaces the matched memory cell if the affinity between $MC_{candidate}$ and $MC_{matched}$ is greater than the *MCRT* (Memory Cell Replacement Threshold), otherwise the candidate memory cell is added into the memory cell subset $MCS_{ag.c}$.

Table 2. The description of the memory cell set update algorithm

Algorithm 2. Memory cell set update
Begin
Input antigen *ag*;
Find the matched memory cell;
Calculate the root mean square *rms* for the candidate memory cell;
If *rms* > *MCIT*
Add the candidate memory cell into the memory cell set;
Else if ((*aff*($MC_{candidate}$, *ag*)>*aff*($MC_{matched}$, *ag*)) and (*aff*($MC_{candidate}$, $MC_{matched}$)> *MCR*T)
Replace the matched memory cell by the candidate memory cell;
Else if (*aff*($MC_{candidate}$, *ag*)>*aff*($MC_{matched}$, *ag*)
Add the candidate memory cell into the memory cell set
End if
End

5 Validating the DC-AIPR Unsupervised Damage Classification Method Using a Benchmark Structure

The combined data clustering and the artificial immune pattern recognition method for the unsupervised structure damage classification has been tested using a benchmark structure [13] proposed by the IASC-ASCE SHM Task Group as shown in Figure 3. The structure data used in our study are the experimental data. In the experimental setup, a variety of damage cases were simulated by removing bracing or

loosening bolts in the test structure. The details of the damage patterns used in the validation are listed in the Table 3.

In the experimental benchmark study, a total of 15 accelerometers were used to measure the acceleration data of the structure, three accelerometers for each level. The measurement data for each damage pattern or the normal pattern were recorded in a data file. Four damage patterns (configuration 2, 4, 5, and 7) and the normal pattern (configuration 1) were selected to validate the DC-AIPR unsupervised classification method. To generate feature vectors for each data pattern, 24,000 points of data in each data file formed 116 of 1000-point time series by advancing 200 points each time. Time series data for 15 accelerometers were reduced to one time series using the PCA method. The compressed 116 time series measurement data for each pattern were then fitted into AR models. The AR order is selected to be 20 since the AIC is small when the AR order is greater or equal to 20 as shown in Figure 4. Since each pattern has 116 feature vectors, a total number of 116*5=580 feature vectors were generated for four damage patterns and the normal pattern.

Fig. 3. Benchmark test structure [13]

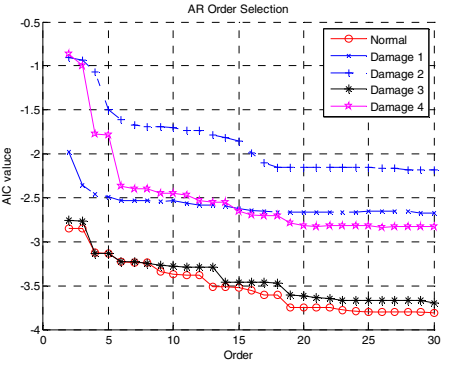


Fig. 4. AR order selection using Akaike's information criterion

Table 3. The configurations for simulating damage patterns

Configuration	Description
1	Fully braced configuration (normal pattern)
2	Missing all east side braces
4	Removed Braces on 1st and 4th floors in one bay on SE corner
5	Removed Braces on 1st floor in one bay on SE corner
7	All braced removed on all faces

These 580 feature vectors of experimental data were used to verify the unsupervised classification algorithm. During the training process, the class labels of the 580 feature vectors were erased. The fuzzy clustering algorithm was applied to find the cluster representatives for each class. Since the feature vectors of the structure data are compact clusters as shown in Figure 1, a point representative is used to represent

each cluster. The outputs of the fuzzy clustering algorithm are five point representatives for five patterns. Each point representative is a vector in R^{20}. The dimension of the point representative is the same as the order of the AR model that is used to represent feature vectors. These five point representatives were used to form the initial memory cell set and also used to classify the training data to five classes based on the nearest neighbor criterion. The classified training data were then used to generate new memory cells based on the artificial immune pattern recognition algorithms.

To test the quality of the newly generated memory cells, the previously created 580 feature vectors were re-used with known class label information. These feature vectors were classified by the memory cells to five clusters. Table 4 shows the number of the feature vectors assigned to each cluster. The classification results shown in the Table 4 demonstrate that the DC-AIPR unsupervised structure damage classification algorithm is able to distinguish damage patterns from each other.

To find the statistical distribution of the classification success rate, the DC-AIPR algorithm is used to classify the 580 feature vectors for 100 times. The resulting distribution of the classification success rate is shown in Figure 5. The numbers on the top of each bar stand for the times that the classification success rate falls into the range indicated on the x-axis. For example, the classification success rate within the range of 82.83%-83.36% occurs 21 times among 100 tests. The classification success rate is defined as the ratio of correctly classified classification data to the whole set of classification data. The system parameters used in the test are CR=8, σ=0.5, *MCRT*=0.95, *MCIT*=0.60, and q=2.

Table 4. The assignment of the training data to each cluster

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Total
pattern 1	3	4	0	8	101	116
pattern 2	0	0	116	0	0	116
pattern 3	0	0	0	95	21	116
pattern 4	0	84	0	4	28	116
pattern 5	90	22	0	0	4	116

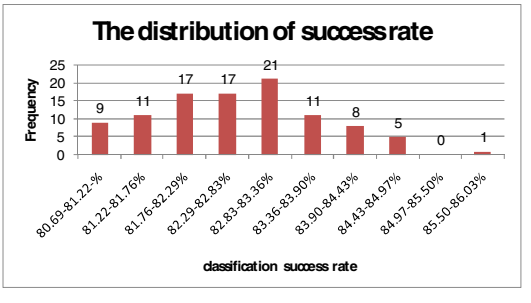


Fig. 5. The distribution of DC-AIPR success rate

6 Performance Analysis of the DC-AIPR-Based Unsupervised Structure Damage Classification

The performance analysis of the DC-AIPR-based unsupervised structure damage classification is conducted in the following aspects. 1) Investigate the impact of the *fuzzifier* of the fuzzy clustering algorithm on the success rate of the unsupervised classification. 2) The impact of the distance types used in the fuzzy clustering algorithm on the success rate of the unsupervised classification. 3) The impact of the *CR* and *MCRT* parameters used in the AIPR algorithm on the success rate and the number of the memory cells. 4) A comparative study of the DC-AIPR method with the DC-SVM (Support Vector Machines) and the DC-Naive Bayes approaches.

To investigate the effect of the *fuzzifier* parameter, the classification success rate with different *fuzzifier* values are calculated and plotted in Figure 6. From the Figure 6, we can see that the value of *fuzzifier* has a significant impact on the classification success rate. When the value of the *fuzzifier* is within the range of 1 to 2.4, the classification success rate is over 80%. The further increasing the value of the *fuzzifier*, the classification success rate will drop immediately to around 55%. The Figure 6 also shows that the classification success rates of the combined DC-AIPR method are greater than that of the fuzzy clustering only method at most values of *fuzzifier*. The differences of the classification success rate between the combined DC-AIPR method and the FC only method are shown in Figure 7.

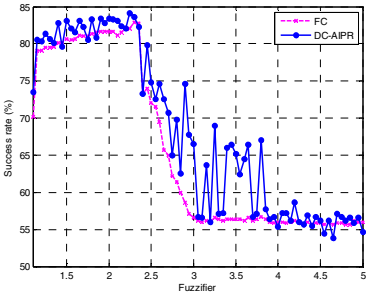


Fig. 6. Comparison of the classification success rate between the combined DC-AIPR method and the FC only method

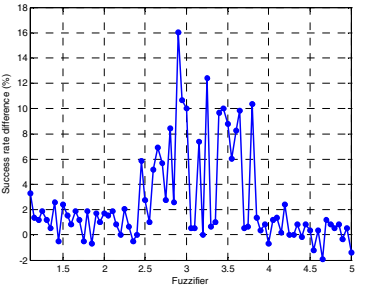


Fig. 7. The difference of the classification success rate between the combined DC-AIPR method and the FC only method

The impact of using different types of distance definitions in the fuzzy clustering algorithm on the classification success rate is also investigated. The classification success rates shown in Figure 6 are calculated using the Euclidean distance in the fuzzy clustering algorithm. Figures 8 and 9 show the classification success rates when the Mahalanobis distance and the Diagonal distance are used, respectively. Figures 6, 8 and 9 demonstrate that the classification success rates are about the same for the Euclidean distance and the Diagonal distance when the value of the *fuzzifier* is below 2.4. The classification success rates are much lower when the Mahalanobis distance is used.

Figures 10 and 11 show the impact of the AIPR parameters, the *CR* and *MCRT*, on the classification success rate and the number of memory cells respectively. The classification success rate and the number of memory cells are two major performance measurements of a classifier. The higher the classification success rate and the lower the number of memory cells, the better the classification performance. The number of memory cells is critical in the SHM sensor networks. Although a big memory cell set may raise the classification success rate, it will result in heavy computational load and slow system response. The value of the *MCRT* has a significant impact on the number of memory cells and the classification success rate as shown in Figures 10 and 11. When the value of the *MCRT* gets bigger, less matched memory cells are replaced, while more candidate memory cells are added into the memory cell set. The value of the *CR* also affects the number of memory cells, but has very little impact on the classification success rate. The appropriate values of the *CR* and *MCRT* should be chosen to limit the number of memory cells and achieve a reasonable classification success rate.

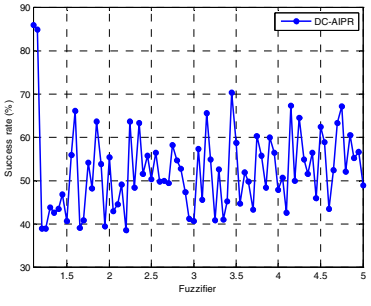


Fig. 8. Success rate using mahalanobis distance

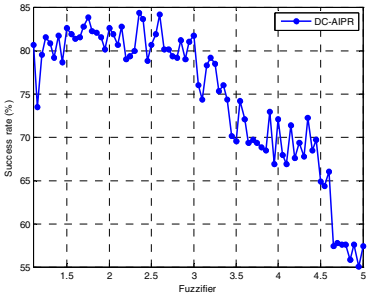


Fig. 9. Success rate using diagonal distance

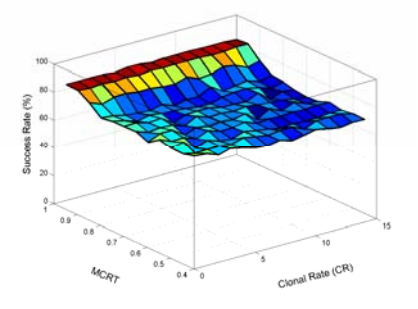


Fig. 10. CR and MCRT vs. classification success rate

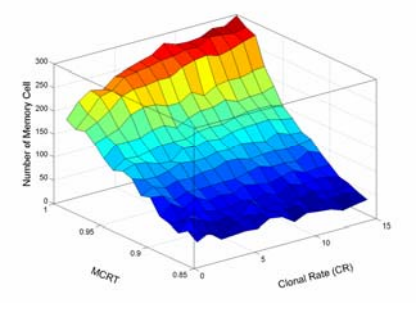


Fig. 11. CR and MCRT vs. the number of memory cells

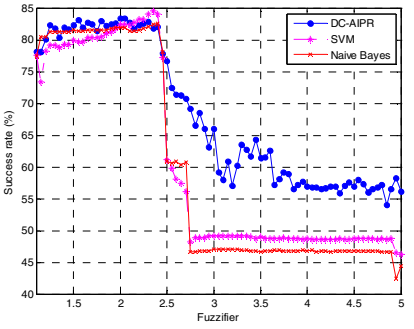


Fig. 12. Comparison of the classification success rate among the DC-AIPR, DC-SVM, and DC-Naive Bayes

The comparison of the classification success rate among the DC-AIPR, DC-SVM, and DC-Naive Bayes is shown in Figure 12. The 580 feature vectors generated above are used in the comparison study. The system parameters selected for the DC-AIPR algorithm are CR=8, σ =0.5, $MCRT$=0.95, $MCIT$=0.60. When the value of the fuzzifier q varies from 1 to 5 with step 0.05, the classification success rates for different classifiers are calculated and the results are shown in Figure 12. From Figure 12, we can see that the DC-AIPR, DC-SVM, and DC-Naive Bayes classifiers have similar classification success rate if the value of the fuzzifier q is less than 2.5. When the q value is greater than 2.5, the DC-AIPR outperforms significantly comparing to the DC-SVM and DC-Naive Bayes algorithms.

7 Conclusions

An unsupervised classification algorithm based on the data clustering technique and the artificial immune pattern recognition for unsupervised structure damage classification is presented in this paper. The fuzzy clustering method is used to generate initial memory cells for each damage pattern based on the structure's dynamic response data. The initial memory cells are then evolved using artificial immune pattern recognition algorithms to improve the representative quality of memory cells. The DC-AIPR method has been used to classify structure damage patterns using a benchmark structure proposed by the IASC-ASCE SHM Task Group. The validation results show the feasibility of using the DC-AIPR method for the unsupervised structure damage classification. The performance analysis of the DC-AIPR-based unsupervised structure damage classification illustrates that some of the classifier parameters, such as *fuzzifier,* distance types in fuzzy clustering algorithm, and the memory cell replacement threshold, have a significant impact on the classification success rate and the number of memory cells. The comparison of the DC-AIPR, DC-SVM, and DC-Naive Bayes algorithms shows that the DC-AIPR method outperforms other two methods for the unsupervised damage classification using the IASC-ASCE benchmark structure.

References

1. Kolakowski, P.: Structural Health Monitoring - a Review with the Emphasis on Low-Frequency Methods. IPPT Engineering Transactions 55, 239–275 (2007)
2. Sazonov, E., Janoyan, K., Jha, R.: Wireless intelligent sensor network for autonomous structural health monitoring. Proc. SPIE - Int. Soc. Opt. Eng. 5384, 305–314 (2004)
3. De Castro, L.N.: Fundamentals of Natural Computing: Basic Concepts, Algorithms, and Applications. Chapman & Hall/CRC, Boca Raton (2006)
4. Hart, E., Timmis, J.: Application areas of AIS: the past, the present and the future. Applied Soft Computing Journal, 191–201 (2008)
5. Dasgupta, D.: Advances in artificial immune systems. IEEE Computational Intelligence Magazine 1, 40–49 (2006)
6. Freitas, A.A., Timmis, J.: Revisiting the foundations of artificial immune systems for data mining. IEEE Transactions on Evolutionary Computation 11, 521–540 (2007)
7. Watkins, A., Timmis, J., Boggess, L.: Artificial immune recognition system (AIRS): an immune-inspired supervised learning algorithm. Genetic Programming and Evolvable Machines|Genetic Programming and Evolvable Machines 5, 291–317 (2004)
8. Zhong, Y.F., Zhang, L.P., Gong, J.Y., Li, P.X.: A supervised artificial immune classifier for remote-sensing imagery. IEEE Transactions on Geoscience and Remote Sensing 45, 3957–3966 (2007)
9. Dasgupta, D., KrishnaKumar, K., Wong, D., Berry, M.: Negative selection algorithm for aircraft fault detection. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 1–13. Springer, Heidelberg (2004)
10. Taylor, D.W., Corne, D.W.: An Investigation of the Negative Selection Algorithm for Fault Detection in Refrigeration Systems. In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 34–45. Springer, Heidelberg (2003)
11. Cesana, E., Beltrami, S., Laface, A.E., Urthaler, A., Folci, A., Clivio, A.: Current paradigms in immunology. In: Apolloni, B., Marinaro, M., Nicosia, G., Tagliaferri, R. (eds.) WIRN 2005 and NAIS 2005. LNCS, vol. 3931, pp. 244–260. Springer, Heidelberg (2006)
12. Negoita, M.: Artificial immune systems - an emergent technology for autonomous intelligent systems and data mining. In: Gorodetsky, V., Liu, J., Skormin, V.A. (eds.) AIS-ADM 2005. LNCS (LNAI), vol. 3505, pp. 19–36. Springer, Heidelberg (2005)
13. Structural Health Monitoring Benchmark Problem, `http://mase.wustl.edu/wusceel/asce.sshm/benchmarks.htm`
14. Theodoridis, S., Koutroumbas, K.: Pattern Recognition. Academic Press, London (2008)
15. Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine 2, 559–572 (1901)
16. Sammon, J.W.: A nonlinear mapping for data structure analysis. IEEE Trans. on Computers 18, 401–409 (1969)
17. Chen, B., Zang, C.: Artificial immune pattern recognition for damage detection in structural health monitoring sensor networks. In: Proc. of SPIE: Smart Sensor Phenomena, Technology, Networks, and Systems, San Diego, California (2009)
18. De Castro, L.N., Von Zuben, F.J.: Learning and optimization using the clonal selection principle. IEEE Transactions on Evolutionary Computation 6, 239–251 (2002)

A Sense of 'Danger' for Windows Processes*

Salman Manzoor, M. Zubair Shafiq, S. Momina Tabish, and Muddassar Farooq

Next Generation Intelligent Networks Research Center (nexGIN RC)
FAST National University of Computer & Emerging Sciences (NUCES)
Islamabad, 44000, Pakistan
{salman.manzoor,zubair.shafiq,momina.tabish,
muddassar.farooq}@nexginrc.org

Abstract. The sophistication of modern computer malware demands run-time malware detection strategies which are not only efficient but also robust to obfuscation and evasion attempts. In this paper, we investigate the suitability of recently proposed Dendritic Cell Algorithms (DCA), both classical DCA (cDCA) and deterministic DCA (dDCA), for malware detection at run-time. We have collected API call traces of real malware and benign processes running on Windows operating system. We evaluate the accuracy of cDCA and dDCA for classifying between malware and benign processes using API call sequences. Moreover, we also study the effects of *antigen multiplier* and *time-windows* on the detection accuracy of both algorithms.

Keywords: API Call Sequence, Artificial Immune System, Dendritic Cell Algorithm, Malware Detection.

1 Introduction

Sophisticated computer malware is becoming a serious threat to the information technology infrastructure, which is the backbone of modern e-commerce systems [2]. A recent outbreak of `Conficker` malware affected more than 9 million computers including those of Ministry of Defence, United Kingdom [3]. This incident has proved that commercial anti-virus software, even with updated malware definitions, are incapable of safeguarding our information technology infrastructure. In [7], the authors have shown that commercial anti-virus software are easily fooled using evasion attempts, such as code obfuscation, encryption and polymorphic transformations. Therefore, security experts are now focusing their attention to robust *run-time* malware detection techniques that analyze API call sequence of a process to classify it as *benign* or *malicious*. Intuitively, such *dynamic* techniques are resilient to the above-mentioned evasion attempts because malware has to eventually execute the malicious activity.

Artificial Immune Systems (AIS) have served as a natural source of inspiration to develop dynamic systems for process classification. The field of AIS was initially dominated by the *self/nonself* theory, which models the working

* Apologies to Forrest et al. [9].

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 220–233, 2009.

of *adaptive* immune system. Forrest et al. initially used the idea of *self/nonself* to develop the *negative selection algorithm* (NSA) [9]. Initially, NSA was used to classify a computer process as benign or malicious. NSA has been incrementally improved and several advanced versions are now available, such as the real-valued NSA [10], the *randomized* real-valued NSA [11], and the real-valued NSA with variable sized detectors [17]. However, Stibor et al. carried out several experiments to evaluate the appropriateness of NSA for anomaly detection. The authors showed that negative selection algorithm is not suitable for higher dimensional datasets [20], [21].

AIS research community has recently turned its attention to a new generation of immune-inspired AIS algorithms which mimic the working model of the *innate immune system* [4]. The fundamental principle of such algorithms is that the innate immune system responds to 'danger' instead of 'nonself'. Aickelin et al. proposed a new AIS algorithm called *Dendritic Cell Algorithm* (DCA) to overcome the above-mentioned shortcomings in NSA [6], [12].

The classical DCA (cDCA) consists of a number of context specific stochastic variables which makes it difficult to systematically analyze a given task. Consequently, Greensmith et al. [15] proposed a simplified and more predictable version of DCA which is called deterministic DCA (dDCA). Since its original inception, two major improvements are proposed for DCA namely *antigen multiplier* and *time-windows*. Gu et al. have initially investigated the usefulness of these concepts for DCA [16].

In this study, we investigate the relative merits/de-merits of cDCA and dDCA, coupled with *antigen multiplier* and *time-windows* concepts, for malware detection. In order to ensure real-world relevance, we have collected API call traces by running 100 benign and 416 malicious Windows executables in a virtual environment.[1] The malicious executables include trojans, viruses and worms. We quantify the efficacy of DCA in terms of its detection accuracy.

The remaining paper is organized as follows: Section 2 presents a summary of related work. Section 3 provides an overview of the DCA and its variations. In Section 4, we explain the collection process of API call traces for real-world malware and benign processes. Section 5 describes our experimental setup and presents the detailed discussions on empirical results. In Section 6, we briefly discuss major limitations of DCA and their potential countermeasures. Finally, we conclude our paper in Section 7.

2 Related Work

AISs have served as a natural source of inspiration for designing anomaly detection systems. To maintain focus, we only discuss the most relevant research.

Classical AIS algorithms are inspired by the working of adaptive immune system which follows principles of the self/nonself theory. In this paradigm, NSA has attained the status of a defacto standard. It was proposed by Forrest et al. for classification of anomalous processes in a computer system [9]. Several advanced

[1] The datasets used in this paper are available at `http://www.nexginrc.org`

versions of NSA have been proposed to date which include but are not limited to the real-valued NSA [10], the *randomized* real-valued NSA [11], and the real-valued NSA with variable sized detectors [17]. The advanced versions of NSA improve its scalability, space coverage, convergence time and formal treatment. Even with the above-mentioned improvements, NSA has been widely criticized for poor scalability behavior especially at higher dimensions [20], [21].

Danger theory proposed by Matzinger [19] claims that immune system works by sensing 'danger'. In [6], the authors investigated the feasibility of using danger theory to develop a new paradigm of AIS algorithms for network intrusion detection. In [12], Greensmith et al. developed a novel DCA based on the concepts of *danger theory*. The authors successfully applied cDCA for classification of breast cancer dataset. In [13] and [14], the authors used cDCA for SYN scan detection.

Since the seminal work of Greensmith et al., several variations of DCA have been proposed. In [16], the authors enhanced DCA with two additional features, called *antigen multiplier* and *time-windows*. DCA relies on aggregate sampling of the antigens for eventual classification; therefore, *antigen multiplier* was used to improve sampling process. Each antigen was multiplied 10, 50 and 100 times to study the effect of multiple sampling. The authors also used *time-windows* to study the aggregate effect of signals. They used fixed time-windows of 2, 3, 5, 7 and 10 instances. They also compared DCA with NSA and C4.5 decision tree for benchmark comparison.

In [15], the authors proposed the dDCA. Several stochastic variables of cDCA were removed to understand the merits/demerits of the core algorithm. Three relevant modifications introduced in dDCA were: (1) a simple signal processing procedure, (2) context evaluation based on one factor, ($\bar{k}$), which was used to ultimately calculate an anomaly score K_α, and (3) a new metric (T_k) was defined to determine threshold for K_α. The authors evaluated the detection accuracy of dDCA using the PING scan dataset. In the next section, we provide a detailed introduction of DCA and its variations.

3 Dendritic Cell Algorithm and Its Variations

Dendritic cells (DCs), of the innate immune system, are the core component of DCA. They have the ability to sense the internal conditions of a tissue by detecting various signals. A *safe* signal is produced in an event of natural cell death (apoptosis), which reflects the normal environment of a tissue. On the contrary, unnatural death of cells (necrosis) because of injury or pathogenic infection leads to the release of *danger* signals. Another strong indicator of potentially harmful environment is *pathogen associated molecular pattern* (PAMP).

Newly born DCs are in an immature state and scour a tissue for antigens (suspect) and signals (evidence). Antigens and signals together evaluate the context of a tissue as *benign* or *potentially malicious*. DCs distinguish between contexts by taking different pathways to their maturity. A *fully matured* state of a DC is the result of exposure to higher concentration of danger and PAMP signals.

Likewise, *semi-matured* state of a DC depicts exposure to higher concentration of safe signals. A collective assessment of DC population activates or suppresses the immune response. We now explain the details of different variations of DCA.

3.1 Classical DCA (cDCA)

In cDCA, proposed by Greensmith et al. [12], a population of 100 DCs is maintained. Each DC is assigned a random migration threshold which limits the amount of time it spends in a tissue. A subset of population is randomly sampled to form a sampling pool of antigens. The selected DCs spend time in a tissue to collect antigens and signals. The input signals are multiplied with predefined weights to calculate output signals. In this paper, for cDCA, we have used same weight values as proposed by the authors in [12]. Three output signals (O_0, O_1, O_2) are calculated for each DC as: $O_i = \sum_{j=0}^{j=2} W_{i\,j}\, S_j, \forall\, i$, where i refers to the category of output signal, j refers to the category of input signal, W is the weight matrix, S is the input signal vector and O is the output signal vector. O_0 is costimulatory signal (csm) and it migrates to the lymph node if the value of csm exceeds assigned migration threshold. In order to derive a context, DC computes two more outputs: (1) the semi-mature context (O_1), and (2) the mature context (O_2). The values are compared with one another and the overall context is termed as safe if O_1 is greater than O_2, and vice-versa.

DCs that have lived their allotted span migrate to the lymph node. The antigens and their corresponding contexts are saved to a log file. Each antigen is sampled multiple times so that it can appear in different contexts in a log file. In order to detect potentially malicious antigens, they are tagged with a mature context antigen value $(MCAV)$. $MCAV$ for a particular antigen i, $(MCAV_i)$, is derived by dividing the number of times that antigen (Ag_i) has appeared in the danger context (N_{di}) by total number of appearances (N_i). Mathematically, $MCAV_i = \frac{N_{di}}{N_i}$.

A threshold (T) is applied to $MCAV$ to make the final classification decision. The antigens with $MCAV$ higher than T are termed *malicious*, and vice-versa. Let ζ_m be the number of anomalous instances and ζ be the total number of instances in a dataset. We can define $T = \frac{\zeta_m}{\zeta}$.

3.2 Deterministic Dendritic Cell Algorithm (dDCA)

The DCA has provided promising classification accuracy results on a number of benchmark datasets [12], [13]. However, the basic DCA uses several stochastic variables which make its systematic analysis very difficult. In order to mitigate this problem, the authors in [15] have proposed some changes in cDCA. The new variation of DCA, called dDCA, has following enhanced features:

- Three input signal categories are reduced to two, i.e. danger and safe signal;
- Random migration threshold is replaced with uniform distribution of lifespan values in a population;
- Dedicated storage and sampling of antigens is replaced with sampling of all antigens by DCs;

- Instead of forming a sampling pool, the signals' data is processed by all DCs. As a result, output signals are calculated once for population of DCs;
- Only one factor ($\bar{k}$) is calculated for each DC to arrive at a context. Negative values of $\bar{k}$ reflect a benign context and positive values indicate a malicious context.

Signal processing is simplified by reducing the number of input signals and using a weight assigning scheme. Two outputs are calculated: (1) accumulation of signals (csm), and (2) score ($\bar{k}$), to which the threshold is applied for classification. csm is defined as $csm = D - S$, and $\bar{k} = D - 2S$, where D and S are values of danger and safe signals respectively. A new parameter K_α is defined using the values of $\bar{k}$. Its purpose is to provide real-valued scores. K_α is defined as $K_\alpha = \frac{\sum_m k_m}{\sum_m \alpha_m}$, where k_m is the $\bar{k}$ value for DC_m, and α_m is the number of antigens of type α presented by DC_m. Moreover, a threshold parameter (T_k) is also defined. The values of K_α greater than the value of T_k depict malicious context and smaller values indicate benign behavior. T_k is defined as $T_k = \frac{S_k . \bar{i}}{I_s}$, where I_s is the total number of instances in a dataset, $\bar{i}$ is the mean number of iterations per incarnation of a DC, and $S_k = \sum_{I_s} D - 2 \sum_{I_s} S$.

3.3 Antigen Multiplier

DCA has been mostly utilized for data mining problems. Most of the datasets used for data mining contain only one copy of each instance (or antigen). In order to assess the type of an antigen, it should be presented multiple times so that $MCAV$ value can be generated for it. The concept of antigen multiplier caters for this requirement [16]. Each antigen is copied multiple times in the tissue antigen vector. The classification decision is now averaged over the replicated population. Intuitively, replicating an antigen should help in improving the classification accuracy.

3.4 Moving Time-Windows

The signals in our body do not die suddenly; rather, they fade slowly over a period of time. This temporal effect of signals is captured by introducing the concept of moving time-windows in DCA [16]. New signals are computed using: $N_{i\,j} = \frac{1}{w} \sum_{n\,=\,i}^{i\,+\,w} O_{n\,j}, \forall\, j$, where $N_{i\,j}$ is new signal value of i^{th} antigen of j^{th} category, w is the window size and $O_{i\,j}$ is original signal of i^{th} antigen and j^{th} category. New signals (N) are the average of old signals (O) in a particular time-window. Intuitively speaking, averaging of signals reduces the noise in input signals.

4 Dataset

In this section, we provide statistics of the benign and malware executable files used in our study. We also describe the commercial API call tracer, API Monitor 1.5, used for logging API traces.

Table 1. Statistics of the Data used in this Study

Executable Type	Quantity	Avg. Filesize (Kilo Bytes)	Min. Filesize (Kilo Bytes)	Max. Filesize (Kilo Bytes)
Benign	100	1,263	4	104,588
Trojan	117	270	1	9,277
Virus	165	234	4	5,832
Worm	134	176	3	1,301
Total	516	50	2	1,332

4.1 Benign and Malware Executables

We have collected a set of 416 malware and 100 benign Windows executables. The malware executables consist of trojans, viruses and worms. All of them are in `Win32` portable executable (PE) format. The benign executables are obtained from a freshly installed copy of Windows XP and application installers. The malware executables are obtained from VX Heavens virus collection which is publicly available [5]. Table 1 provides statistics of the executables used in our study.

4.2 API Call Tracer

We have used API Monitor 1.5 to log the API call sequences of Windows processes. It captures these API calls and stores them in `apm` file format [1]. It has an *API* and a *process* filter. The API filter gives us the option of filtering unnecessary API calls by category. In the API filter, we can select the calls of following categories: (1) Dynamic-Link Libraries, (2) Memory Management, (3) Network Management, (4) Processes and Threads, (5) Registry, and (6) Socket. The process filter allows us to filter API calls made by different processes. We have captured API calls of all system-wide processes because some malware in our study use Windows processes like `explorer.exe` to carry out malicious activities. Therefore, it is not possible for us to exactly pin-point a set of processes for monitoring.

We install API call tracer on a fresh virtual machine of Microsoft Windows XP and also create its backup. After execution of each malware or benign executable, we replace the virtual machine source with the original backup. We capture API calls from the start of execution of a process till it finishes.

4.3 Feature Selection and Extraction

We use $n-$gram analysis for feature extraction. $n-$gram of a sequence is the normalized frequency histogram of n successive elements of the sequence [8]. $n-$grams computed with very less value of n contain insufficient information and those with very large values of n incur unacceptably high processing overheads. So, we have to choose a suitable value of n to get sufficient information from the $n-$grams while incurring reasonable processing overheads. In this study, we

have used the value of $n = 4$. Each API function is mapped to a unique random variable. We extract the most informative 4−grams from all dataset files by ranking them according to their *information gain*, also known as *average mutual information*. The information gain of a feature i is defined as [18]:

$$I(Y; X) = H(Y) - H(Y|X),$$

where X is an input attribute, Y is a class attribute, $H(Y)$ is the entropy of the class attribute variable Y and $H(Y|X)$ is the conditional entropy of Y with respect to X. Therefore, information gain of an input attribute quantifies the reduction in uncertainty of the class attribute given that we know the value of input attribute. We have selected top 500 4−grams sequences by applying threshold to information gain values. 500 4−grams are selected to ensure that adequate amount of relevant information is selected for the signal calculation process, which is explained in the next section.

For feature extraction, we check the log file of each executable file for presence or absence of the selected n−grams. We place 1 if the n−gram is present and 0 otherwise. Each executable log is mapped to a 500−dimensional binary string. All strings generated for benign executables are placed in a separate file and the same is done for trojans, viruses and worms. We then combine the separate files to create three datasets: benign-trojan, benign-virus and benign-worm. In the next section, we explain the detection accuracy of different variations of DCA using each of the above-mentioned datasets.

5 Experimental Analysis

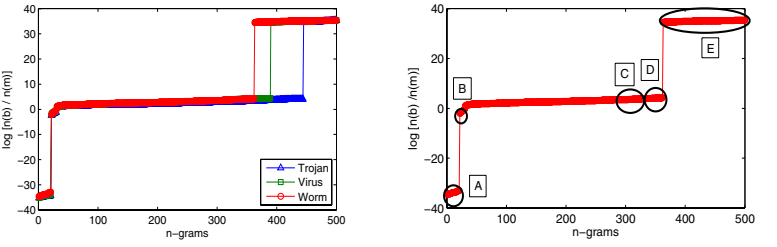
5.1 Signals and Antigens

We have used boolean information about presence or absence of top 500 n−grams, which characterize the activities of each executable.

To map information from boolean feature vectors to signals in a systematic manner, we propose an intuitive procedure. Firstly, we count the number of times a given feature (n−gram sequence) is present in benign and malicious logs. We term these counts as $n(b)$ for benign logs and $n(m)$ for malware logs. We then compute the *kappa metric* (κ) for every n−gram as:

$$\kappa = \log_e \left(\frac{n(b)}{n(m)} \right)$$

Figure 1(a) shows the plot of κ for every n−gram in all datasets. An interested reader will appreciate the peculiar nature of the plot, which can be helpful in selecting n−grams for signal calculation. Figure 1(b) shows the plot of only benign-worm dataset where potentially interesting n−gram regions are marked with black circles. Intuitively, n−grams with positive values of κ represent benign behavior and vice-versa. We now detail the formation of all types of signals: (1) safe, (2) PAMP, and (3) danger.



(a) κ distribution plot of n-grams

(b) Regions of interest for benign-worm dataset

Fig. 1. κ distribution plot of n-grams for all datasets

For safe signal, we are interested in $n-$grams which are mostly present in the traces of benign processes (region D), and those which are *never* present in benign traces (region A). For PAMP signal, we are interested in $n-$grams which are mostly present in malware traces (region B), and those which are *never* present in the traces of malware processes (region E). To derive the magnitude of safe and PAMP signals, we add a predefined value to the signal value if we are: (1) able to find $n-$gram that is supposed to be present, and (2) unable to find $n-$gram that is supposed to be absent. Finally, we normalize the value signal in a desired range.

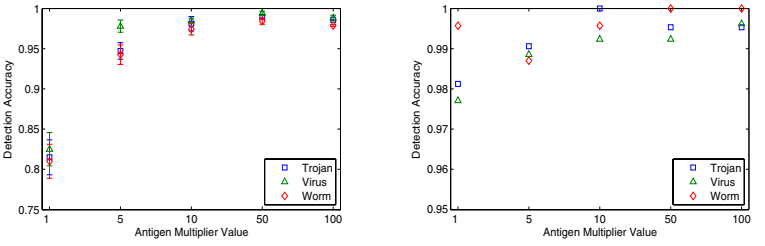
To extract danger signal, we are interested in $n-$grams belonging to region C. These $n-$grams have high probability of being present in benign traces, but not as high as required for deriving safe signal. We use the absence of these $n-$grams as the evidence of danger. Remember that large deviations from benign behavior reflect associated danger. In order to create antigens, we label each instance of all datasets with an integer value.

5.2 Experimental Setup

We now describe the experimental setup used in our study. We plan to examine the effect of antigen multiplier and moving time-windows on the performance of cDCA and dDCA in terms of detection accuracy. We perform independent experiments using all three datasets—benign-trojan, benign-virus, benign-worm.

We have designed three sets of experiments for each algorithm. First two experiments determine the effect of varying the amount of antigen multiplier value and sizes of moving time-windows independently on the detection accuracy. The objective of first two experiments is to determine the best antigen multiplier value and the best size of time-windows. In the third experiment, we analyze the combined effect of both techniques on the detection accuracy of both algorithms.

- **E.0:** Effect of antigen multiplier value on the detection accuracy of cDCA and dDCA.
- **E.1:** Effect of size of moving time-windows on the detection accuracy of cDCA and dDCA.
- **E.2:** Combined effect of antigen multiplier value and size of moving time-windows on the detection accuracy of cDCA and dDCA.



(a) Accuracy plot for basic DCA (b) Accuracy plot for dDCA

Fig. 2. Accuracy results for varying Antigem Multiplier

In our experimentation, the number of DCs are kept constant at 100 for both algorithms. All datasets are ordered, i.e., benign class followed by malicious class. The experiments are conducted on a 2.2 GHz Dell Vostro 1510 Core 2 Duo.

Parameters of cDCA. The thresholds values used in this study are derived from distributions of the datasets. We divide the number of anomalous instances with the total number of instances in a dataset to compute threshold. As a result, we have determined threshold values of 0.547, 0.630 and 0.580 for benign-trojan, benign-virus and benign-worm datasets respectively.

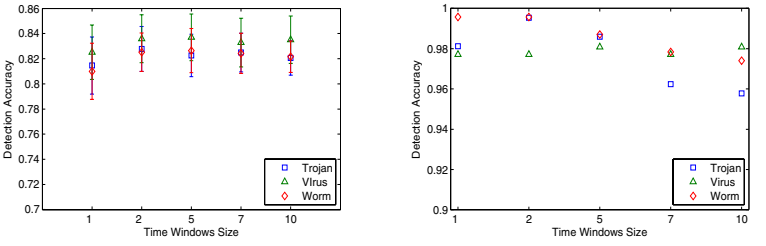
Danger and PAMP signals are normalized within a range of 0 to 100 while safe signal is normalized within a range of 0 to 66. The migration threshold is kept between 300 to 500. The use of high values ensures that the classification of current instance is affected by its neighboring instances. Each experiment is repeated 10 times and the averages, along with standard deviations, of these runs are plotted in Figures 2, 3 and 4.

Parameters of dDCA. For dDCA, we use same values of PAMP and safe signals as that of cDCA. Lifespans are uniformly distributed between a range of 300 to 500 across the DC population. The increments are computed by dividing the range of lifespan with the number of used DCs. As before, the higher values include the effect of neighboring instances on the classification of current instance, which tends to reduce the error because of noise during signal calculation.

5.3 Discussions on Results

E.0: Effect of antigen multiplier on cDCA and dDCA. Remember that by using multiplier, each antigen is copied several times into the tissue antigen vector. More than one presentations of the same antigen allows multiple DCs to evaluate its context. Intuitively speaking, the collective assessment by DC population should provide more accurate prediction since final context is less vulnerable to the wrong judgment of a single DC.

The need for antigen multiplier is justified where insufficient antigens are available for doing classification. This is a common situation in data mining



(a) Accuracy plot for basic DCA (b) Accuracy plot for dDCA

Fig. 3. Accuracy results for varying size of time-windows

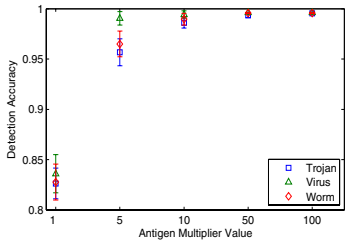
applications. For example, KDD 99 dataset used by Gu et al. in [16] and breast cancer dataset used by Greensmith et al. in [12] have only one copy of each antigen. Therefore, antigen multiplier can play an important role in improving the classification accuracy in such applications. We have used antigen multiplier values of 1, 5, 10, 50 and 100. Note that the multiplication factor of 1 refers to the case when antigen multiplication is not applied.

The results of cDCA are shown in Figure 2(a). DCA achieves poor classification accuracy when antigen multiplication is 1 because there is just one antigen of each type which may get picked by a DC having either a very high or a very low migration threshold. In the former case, it may get associated with the instances of a wrong class, and in the later case the neighboring instances would hardly have any effect because the DC would migrate to a lymph node in a single iteration. As expected, the detection accuracy of the algorithm significantly increases with an increase in the antigen multiplication factor.

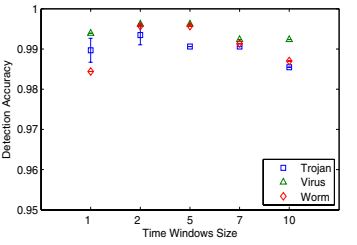
The results of varying antigen multiplier in dDCA are shown in Figure 2(b). The classification accuracy trend is similar to the one observed in cDCA. However, the classification accuracy of dDCA, even without antigen multiplication, is significantly better compared with the cDCA. This improvement can be attributed to the use of T_k, which caters for different ranges of normalization of signals and their effect on the migration threshold. As a result, the algorithm becomes more adaptive to variations of parameters. Finally, we observe that the antigen multiplication factor of 100 for *dDCA* achieves better classification accuracy than all antigen multiplier configurations of cDCA.

E.1: Effect of Moving Time-Windows on cDCA and dDCA. When we use the concept of moving time-windows, we actually take average of the magnitude of signals present in that window. This effectively reduces the signal's noise by including the effect of neighboring signals. The signals are also averaged near class boundaries. This helps in reducing the possibility of false predictions. We have used time-windows of sizes 1, 2, 5, 7 and 10 in our study.

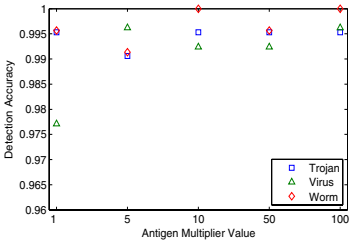
The results of moving time-windows for cDCA are plotted in Figure 3(a). The poor accuracy of cDCA at a time-window of size 1 stems in the same reasons already explained for antigen multiplication factor of 1. However, it is interesting



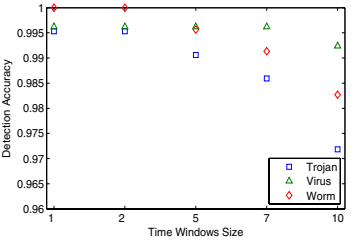
(a) Accuracy plot for varying Antigen Multiplier with Time-Windows fixed at 2 for basic DCA



(b) Accuracy plot for varying Time-Windows with Antigen Multiplier fixed at 50 for basic DCA



(c) Accuracy plot for varying Antigen Multiplier with Time-Windows fixed at 2 for dDCA



(d) Accuracy plot for varying Time-Windows with Antigen Multiplier fixed at 100 for dDCA

Fig. 4. Accuracy results for cascading antigen multiplier and time-windows

to note that increasing the size of time-windows has almost negligible effect on accuracy of the algorithm. This behavior is due to the fact that noise in the obtained signals is already fairly low, which leaves little room for removing the noise or making other relevant improvements.

The results of moving time-windows for dDCA are shown in Figure 3(b). It is interesting to observe that the accuracy drops with an increase in the size of time-window. We contemplate on a possible explanation that unrelated signals and antigens get associated with each other for larger time-windows. This effect is more evident at the boundary of two classes. If we take the example of the last entry of a benign class, most of the signals in the time-window would belong to the malware class. Now if we take average, the signal value is biased towards the malware class. dDCA still achieves better classification accuracy for all respective time-windows sizes than cDCA.

E.2: Combined effect of both antigen multiplier and moving time-windows on cDCA and dDCA. From the results of previous two experiments, it can be deduced that antigen multiplier values of 50 and 100 provide

the best classification accuracy results for cDCA and dDCA respectively. In case of time-windows, the window size of 2 has yielded best average accuracy results for both algorithms.

Figures 4(a) and 4(c) respectively show the results of varying antigen multiplier while keeping the window size fixed at 2 and varying the window size while keeping the multiplier value fixed at 50 for cDCA. Similarly, Figures 4(b) and 4(d) show the results of varying antigen multiplier while keeping the window size fixed at 2 and varying the window size while keeping the multiplier value at 100 respectively for dDCA. It is evident that the detection accuracies are better compared with the results of **E.0** and **E.1** experiments of cDCA.

The results of dDCA in Figure 4(c) show an increase in the classification accuracy reflecting an increase in antigen multiplier value. Likewise, Figure 4(d) shows decrease in the classification accuracy, similar to the trends shown in Figure 3(b), due to the increase in the size of time-windows. The results of dDCA, in Figure 4(c) with antigen multiplier value set to 100 and in Figure 4(d) with the value of time-windows set to 1 and 2, show better classification accuracy than all other settings for dDCA.

From our results, we can conclude that the use of antigen multiplication is highly recommended. While, the effect of time-windows on results of cDCA is minimal and the accuracy drops as the size of time-window increases for dDCA.

6 Limitations and Potential Solutions

In this section, we briefly present the limitations of DCA which we have observed for the presented problem. We also discuss potential solutions of these limitations.

A basic problem with DCA is that it requires adjustment of several parameters and there are no rules of thumb to determine their optimal values. This limitation introduces additional design dimensions which are to be explored for obtaining the best results. For example, there is no standard procedure to determine the weights used in DCA. Intuitively, the optimal values of these weights vary with respect to different properties of the dataset.

DCA also lacks an automated module for signal computation from high-dimensional data. Signal computation strategies vary significantly from one application to another. In this paper, we have introduced a systematic and intuitive method to group and transform high-dimensional input data for typical 2−class problems.

DCA decisions are based on the aggregate sampling of each antigen. This demands multiple presentations of every antigen. Hence, DCA does not work well in situations where insufficient number of antigens are available (such as data-mining). Our study has shown that antigen multiplication is an important concept which can be used to overcome this limitation.

DCA performs a temporal correlation between antigens and the signals for classification. It distinguishes between normal and potentially malicious antigens on the basis of neighboring antigens. This feature can be exploited by crafty attackers

(via mimicry attacks) to evade detection by DCA. The malicious entities may remain undetected by wilfully mimicking benign behavior intermittently. This vulnerability is also observable at the class boundaries.

7 Conclusions

In this study, we have analyzed the feasibility of using two variations of DCA—cDCA and dDCA—for run-time detection of malware. We have also investigated the effect of antigen multiplier and moving time-windows on the accuracy of both algorithms. The results of our experiments highlight the promise of DCA in malware detection applications and relevant 2−class problems.

The important conclusions of our experiments are: (1) danger theory based DCA has the potential in the domain of run-time malware detection, (2) dDCA consistently outperforms DCA in terms of classification accuracy, and (3) antigen multiplier shows promise to improve the detection accuracy while time-windows show little relevance in improving the detection accuracy.

References

1. API Monitor, `http://www.rohitab.com/apimonitor`
2. F-Secure Corporation, F-Secure Reports Amount of Malware Grew by 100% during 2007, Press release (2007)
3. Symantec, Internet Security Threat Report, vol. XIV (2009)
4. The Danger Project, `http://www.dangertheory.com`
5. VX Heavens Virus Collection, VX Heavens website, `http://vx.netlux.org`
6. Aickelin, U., Bentley, P., Cayzer, S., Kim, J., McLeod, J.: Danger Theory: The Link between AIS and IDS? In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 147–155. Springer, Heidelberg (2003)
7. Christodorescu, M., Jha, S.: Testing Malware Detectors. ACM SIGSOFT Software Engineering Notes 29(4), 34–44 (2004)
8. Damashek, M.: Gauging Similarity with n-Grams: Language-Independent Categorization of Text. Science 267, 843–848 (1995)
9. Forrest, S., Hofmeyr, S.A., Somayaji, A., Longstaff, T.A.: A sense of self for Unix processes. In: IEEE Symposium on Security and Privacy, USA, pp. 120–128. IEEE Press, Los Alamitos (1996)
10. Gonzalez, F., Dasgupta, D.: Anomaly Detection Using Real-Valued Negative Selection. Journal of Genetic Programming and Evolvable Machines 4(4), 383–403 (2003)
11. Gonzalez, F., Dasgupta, D., Nino, L.F.: A Randomized Real-Valued Negative Selection Algorithm. In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 261–272. Springer, Heidelberg (2003)
12. Greensmith, J., Aickelin, U., Cayzer, S.: Introducing Dendritic Cells as a Novel Immune-Inspired Algorithm for Anomaly Detection. In: Jacob, C., Pilat, M.L., Bentley, P.J., Timmis, J.I. (eds.) ICARIS 2005. LNCS, vol. 3627, pp. 153–167. Springer, Heidelberg (2005)
13. Greensmith, J., Aickelin, U., Twycross, J.: Articulation and clarification of the dendritic cell algorithm. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 404–417. Springer, Heidelberg (2006)

14. Greensmith, J., Aickelin, U.: Dendritic Cells for SYN Scan Detection. In: Genetic and Evolutionary Computation Conference (GECCO), pp. 49–56. ACM Press, UK (2007)
15. Greensmith, J., Aickelin, U.: The Deterministic Dendritic Cell Algorithm. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 291–303. Springer, Heidelberg (2008)
16. Gu, F., Greensmith, J., Aickelin, U.: Further Exploration of the Dendritic Cell Algorithm: Antigen Multiplier and Time Windows. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 142–153. Springer, Heidelberg (2008)
17. Ji, Z., Dasgupta, D.: Real-Valued Negative Selection Using Variable-Sized Detectors. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 287–298. Springer, Heidelberg (2004)
18. Kolter, J.Z., Maloof, M.A.: Learning to detect malicious executables in the wild. In: International Conference on Knowledge Discovery and Data Mining, pp. 470–478. ACM Press, USA (2004)
19. Matzinger, P.: Tolerance, danger and the extended family. Annual Review of lmmunology 12, 991–1045 (1994)
20. Stibor, T., Timmis, J., Eckert, C.: On the Appropriateness of Negative Selection defined over Hamming Shape Space As a Network Intrustion Detection System. In: IEEE Congress on Evolutionary Computation (CEC), pp. 995–1002. IEEE Press, UK (2005)
21. Stibor, T., Mohr, P., Timmis, J., Eckert, C.: Is Negative Selection Appropriate for Anomaly Detection? In: Genetic and Evolutionary Computation Conference (GECCO), USA, pp. 321–328. ACM Press, New York (2005)

An Immunity Inspired Real-Time Cooperative Control Framework for Networked Multi-agent Systems

Steven Y.P. Lu and Henry Y.K. Lau

Department of Industrial and Manufacturing Systems Engineering
The University of Hong Kong
Pokfulam Road, Hong Kong, PRC
yplu0795@hkusua.hku.hk, hyklau@hku.hk

Abstract. This paper presents a cooperative control framework developed based on the inspiration from the immune system for controlling networked multi-agent systems. The framework is inspired from the meta-dynamics of lymphocyte repertoires in the adaptive immune system, including the continual circulation, continual turnover, concentration control and other relevant mechanisms. We design this framework for the control of a team of autonomous defending agents in RoboFlag Drill, a test-bed for studying cooperative systems. Simulation results are presented to demonstrate the effectiveness of the proposed immunity inspired cooperative control framework. The results of the simulations demonstrated that a set of defenders- can intercept attacker sets with larger set sizes from entering a specific Defense Zone for 60% of the randomly generated RoboFlag Drill problem instances.

Keywords: Artificial Immune Systems, Networked Multi-agent Systems, Cooperative Control, RoboFlag Drill.

1 Introduction

The Immune system uses many strategies to defend the body against threats in a truely distributed, adaptive and robust manner. Its featured mechanisms, including layered defense, self-nonself discrimination, dynamic pathogen space coverage, complement system, immune system suppression, neutriphil swarms and so on, provide rich inspiration for engineers to create successful solutions to complex real-world problems. A new emerging biological metaphor that is inspired by theoretical immunology and observed immune functions, principles and models, called Artificial Immune Systems (AIS), emerged in the 1990s. A number of AIS models, such as negative selection [1], immune network model [2] and clonal selection [3], have been developed and applied to a wide spectrum of problems ranging from pattern recognition and classification [4], fault detection [5], computer security [6] to autonomous robotics [7]. Among them, the distributed and self-organized property of immune system is gaining researchers' attention from the discipline of cooperative control of multi-vechile systems. Lau & Wong [8]developed a distributed multi-agent control framework with the ability to evolve and learn in a dynamic environment.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 234–247, 2009.

Lau & Ko [9] presented a General Suppression Control Framework and applied it to self-balancing robots control.

This paper presents a new cooperative control framework for multiple interacting agents, named AIS-based Cooperative Reaction Framework (ACRF). This framework is inspired from the meta-dynamics of lymphocyte repertoires, including mechanisms for the generation of their diversity and the cooperation that cover the ever-changing space of all antigen patterns [10-12]. Taking the basis of concentration-dependent formation control algorithm, capability complement, manipulation process plus other relevant mechanisms, we designed the AIS-based Cooperative Reaction Framework for controlling a team of defending robots to intercept as many mobile attackers appearing randomly as possible from entering a Defense Zone at the center of a playing field, known as RoboFlag Drill – a popular test bed for studying cooperative control and optimization problems in the field of robotics and automation introduced by Earl and D' Andrea [13]. In this problem, a team of defenders is deployed to intercept an oncoming team of attackers who appear randomly. The goal is to design a team control strategy for the defenders that minimizes the number of attackers entering the Defense Zone. The RoboFlag Drill problem is a special version of a networked multi-agent systems control problem that is characterized by a collection of decision-making components, each having access to local information and limited inter-component communication, but all seeking to achieve a collective objective.

In this paper, we present the results of preliminary simulation studies to demonstrate the effectiveness of ACRF. We undertake a simple but illustrative simulation by solving randomly generated instances of N_D-on-N_A Defensive Drill problems. Simulation results show that our proposed ACRF controls the team of defenders to successfully intercepted teams of attackers with larger team sizes from entering the Defense Zone for 60% of the randomly generated RoboFlag Drill problem instances. The results illustrate that our proposed ACRF is a competitive team control strategy compared to other state-of-the-art strategies [13, 14].

2 The Immune System

Nature has long served as source of inspiration for many of the scientific and technological advances of mankind. Artificial Immune Systems (AIS) emerged in the 1990s as a computation paradigm that adopts the observations and concepts of the human immune system. In essence, human immune system is a distributed, adaptive and robust system consisting of a variety of interacting cellular and molecular elements distributed throughout our body. It takes the role of (self-) body maintenance and defense against (non-self) intruders, through detecting the immunogenic tissue states and eliciting an immediate (innate) or precise (adaptive, antigen-specific) immune response to each detected signal concurrently. The immune system has a multilayered defense architecture: the outermost layer (including the skill and the physiological conditions such as pH and temperature) is the first barrier to infection; the innate immune system consists primarily of macrophages, neutrophils and dendritic cells and clears the system of both debris and pathogens at very short notice; the adaptive immune system is a supplement to

the innate system and is adaptively acquired furing the lifetime of the organism. As the adaptive immune system provides the greatest potential from a multi-agent cooperation viewpoint, we focus our study on the mechanisms embedded within the adaptive immune system that are used to develop an Artificial Immune Systems (AIS) for the cooperative control of multiple interacting agents in the later sections.

Adaptive Immune System (also called "the acquired immune system") is the most sophisticated layer of defense and involves many different types of cells and molecules. It is adaptive in that its immunity is antigen-specific and evolves throughout the lifetime of the organism. The adaptive immune system can be viewed as a cooperative defense system consisting mainly of lymphocytes. One unique function of these lymphocytes that contribute to the defense operation of the entire system is achieving diversity, because the ability of immune system to detect most antigens requires lymphocytes with antigen-specific receptors having a huge diversity to provide a complete coverage of the space of all antigens distributed throughout the body. The immune system has several mechanisms for addressing this problem including:

(1) lymphocyte receptors, in the form of immunoglobulins on B cells and T-cell receptors on T cells, are generated through a complex and elegant genetic process that introduces a large extend of randomness; In the genome, each receptor chain cannot be encoded in full. Instead, the receptor chains are encoded in several pieces-so called gene segments. Each type of gene segment is present in multiple copies, the selection of a gene segment of each type during gene rearragement occurs at random, and the large number of possible different combinations accounts for much of the diversity of the lymphocyte receptors [15].
(2) lymphocytes are not static cells but are in continual circulation through the body. Also, lymphocytes are typically short-lived (a few days) and are continually replaced by new lymphocytes with new randomly generated receptors. The continual circulation of lymphocytes and the continual turnover of the lymphocyte population make the protection of the immune system dynamic, and increase the coverage provided by the immune system over time [16].
(3) concentration of lymphocytes of each type are maintained in a status of dynamic equilibrium when circulating throughout the body. The balanced concentration of lymphocytes in the body increases the coverage provided by the immune system over time further. The change in the concentration of lymphocytes of each type depends on the total stimulation by excitatory signals, the total effect of inhibitory signals, the rate of new entrants and the rate of natural death, according to Jerne's idiotypical network model [2].

As a whole, the lymphocytes population in the adaptive immune system provides a dynamic converage of the space of all antigens distributed throughout the body. One estimate is that there are 10^8 different lymphocyte receptors in the body at any given time, which can detect potentially 10^{16} different foreign patterns [12].

3 Cooperation in Networked Multi-agent Systems

The extent of connectivity in our society is rather amazing. Networks exist in nearly every aspect of the infrastructure that supports our daily life. Electricity, water, transportation, shopping, Internet, health care, banking and education are all brought to us by physical or social networks. There has been a boom of research motivated by the emerging applilcations of large-scale, dynamic and uncertain networked systems, where traditional approaches to control and optimization are no longer feasible due to the need to exploit real-time information and to improve robustness to dynamic uncertainties. Fueled by this increasing interest, major scientific journals have devoted special themes to the discussion and dissemination of topical researches in networked systems, including special issues of *Proceedings of the IEEE* and *SIAM Journal on Control and Optimization* [17, 18], *Nature*, *Operations Research* and *Management Science* have published several reviews on the subject [19-21]. Furthermore, the National Research Council (USA) committee has chosen the topic of 'network-centric operations' as the subject of a study for future army applications [22].

Networked systems are characterized as a collection of decision-making components – each having access to local information and having limited inter-component communication capacity, but all seeking to achieve a collective objective by coordinating their local decisions. There are two unique features of a networked system: (1) the distribution of information; (2) the computational complexity associated with a poentially large number of interacting components and the difficulties of dealing with overlapping or partial information. The central challenge for the design and operation of networked systems is to derive desirable collective behaviors through designing suitable individual agent control algorithms and the inter-agent interaction rules. Although the individual components of these networked systems are increasingly sophisticated, we lack a fundamental understanding of how to assemble and coordinate the individual components into a coherent whole.

A coordination algorithm (or protocol) is an interaction rule that specifies the information exchange between a component and all of its neighbors in the network. Cooperation (or coordination, the terms are used interchangably in this work) in networked systems has been studied by researchers from various disciplines. This includes consensus and averaging algorithms [23, 24], game-theoretic models [25-27], market-based/auction algorithms [28, 29], and so on.

In this paper, we consider a particular version of coordination problems for networks of autonomous agents called 'RoboFlag Drill' [30] illustrated in Fig.1. RoboFlag Drill serves as a fertible testbed for complex strategies of cooperative control and optimization involving multiple interacting agents. In this problem, there are two teams of robots, the defenders and the attackers, interacting on a playing field with a region called the Defense Zone at its center. The attackers are located randomly and move toward the center of the defense zone along straight line paths at a constant velocity. The defenders' collective objective is to thwart as many attackers from entering the Defense Zone as possible by intercepting each attacker before it enters the Zone. The goal is to design a cooperative control strategy for the team of defenders to minimize the number of attackers that enter the Defense Zone. We consider the class of Defensive Drill problems in which each attacker has a fixed intelligence governed by a state machine.

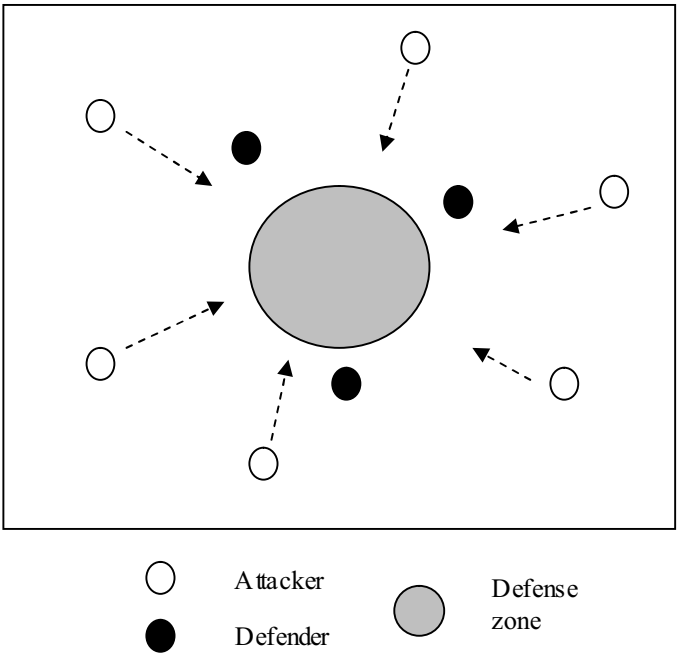


Fig. 1. RoboFlag Drill model

4 AIS-Based Cooperative Reaction Framework

The AIS-based Cooperative Reaction Framework (ACRF) presented in this paper is developed based on the distributed cooperative search and reaction mechanism of adaptive immune system. Adaptive immune system is a special type of networked multi-agent system where the agents (primarily lymphocytes) circulate through the body in the blood and lymph systems and respond to antigens in a distributed and cooperative manner. Our analogy infers that each attacker is an antigen that infects the body and each defender is a lymphocyte that reacts to the uncertaint envionment.

4.1 Initialization of the AIS-Based Cooperative Reaction Framework

The Defensive Drill - We consider N_D-on-N_A Defensive Drill problems [30]. This is a generalized case and involves N_D defenders and N_A attackers. We take the Defense Zone to be circular with radius R_{dz}. We seek a control strategy for the team of N_D that minimizes the number of attackers that enter the Defense Zone. The initial state of the Defensive Drill is shown in Fig. 2.

The attacker - There are two states for an attacker: attack and being inactive. The attacker starts in the attack state at the beginning of play. When in attack mode, the attacker moves toward the center of the Defense Zone at constant velocity along a straight line path. If the attacker is intercepted by a defender or if it enters the Defense

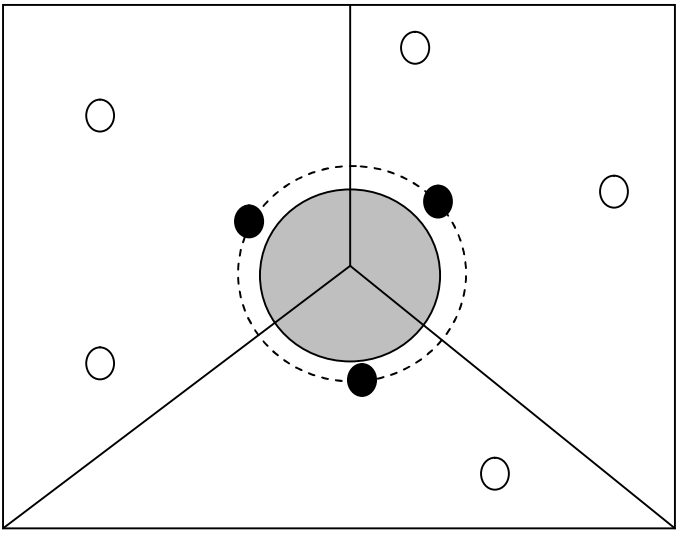


Fig. 2. Initialization of the AIS-based cooperative reaction framework

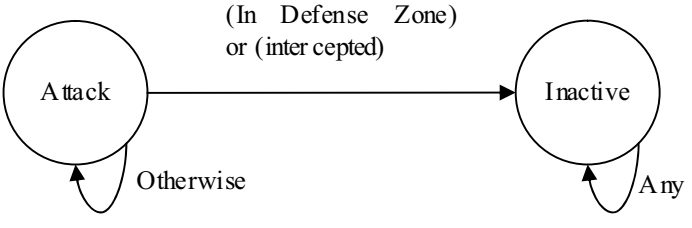


Fig. 3. State transition model of the attacker

Zone, it transits to the inactive mode. When in the inactive state, the attacker does not move and remains inactive for the remainder of play. The state transition model of an attacker is shown in Fig. 3. The initial position and the velocity for each attacker are given by r_a, θ_a, v_a, and values of r_a, θ_a, v_a are randomly picked from a uniform distribution over the intervals $[r_a^{min}, r_a^{max}]$, $[0, 2\pi]$, and $[v_a^{min}, v_a^{max}]$, respectively. There are a set of attacker types [31] , and each attacker in our Defensive Drill problems belongs to a specific type and is different from other attackers.

The defender - There are two major states for a defender: circulation and reaction. The defender starts in the circulation state at the beginning of play. When in circulation mode, the defender circulates on a predefined orbit around the Defense Zone at constant velocity. If the defender binds to an attacker within its sensing region, it transits to reaction mode. When in reaction mode, the defender moves toward the attacker, intercepts it, and then returns to the orbit. Once the defender returns to the orbit, it transits to circulation mode again. The state transition model of a defender is shown in Fig. 4. The velocity for all defenders in circulation mode is given by v_d, and value of v_d is randomly picked from a uniform distribution over

the intervals [v_d^{min}, v_d^{max}]. Each defender only has the capability to react to a specific type of attacker at any given time. Initially, the defenders on the orbit have different capabilities with each other. To define what it means for a defender to intercept an attacker, we introduce an intercept region rigidly attached to the defender. The intercept region is a circular region with radius R_{inc}. If the attacker is in this region, it is considered being intercepted. The defender should not move inside the circular region formed by the orbit when reacting to an attacker. Therefore, the radius of the orbit on which defenders circulate also depends on the radius of the intercept region, and is given by:

$$R_{orb} = R_{dz} + R_{inc}$$

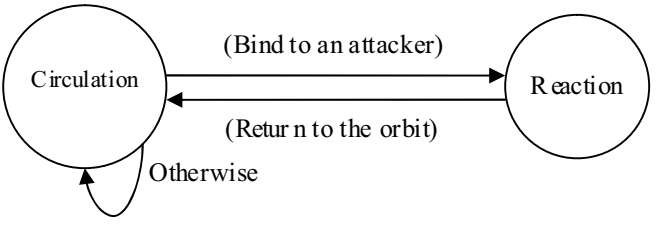


Fig. 4. State transition model of the defender

4.2 Control Logic for an AIS-Based Defender

The control logic for defenders bears a close analogy to the mechanisms used by lymphocytes in the immune system. In circulation mode, the AIS-based defenders achieve coverage of the playing field by a concentration-dependent formation control algorithm, and the diversity of AIS-based defenders' capabilities is achieved through a capability complement and manipulation process.

Concentration-dependent formation control algorithm - concentration $C_{D1\text{-}D2}$ is defined as the reciprocal of the arc distance between two adjacent defenders (*D1, D2*) on the orbit. The AIS-based defenders keep formation by the rule:

$$C_{D-adjacentD} - C_{D-theOtherAdjacentD} = 0$$

$$C_{D-adjacentD} = \frac{1}{Dis_{D-adjacentD}}$$

This formation is maintained dynamically. When a defender deviates from the orbit for a reaction or goes into the orbit after finishing the reaction, the formation will be adjusted according to the concentration-dependent rule stated above. The concentration-dependent formation control algorithm is shown in Fig. 5. Accordingly, the sensing region of an AIS-based defender is designed to be a fan-shaped region with an averaged central angle of: $360 \big/ N_{D-on-the-orbit}$.

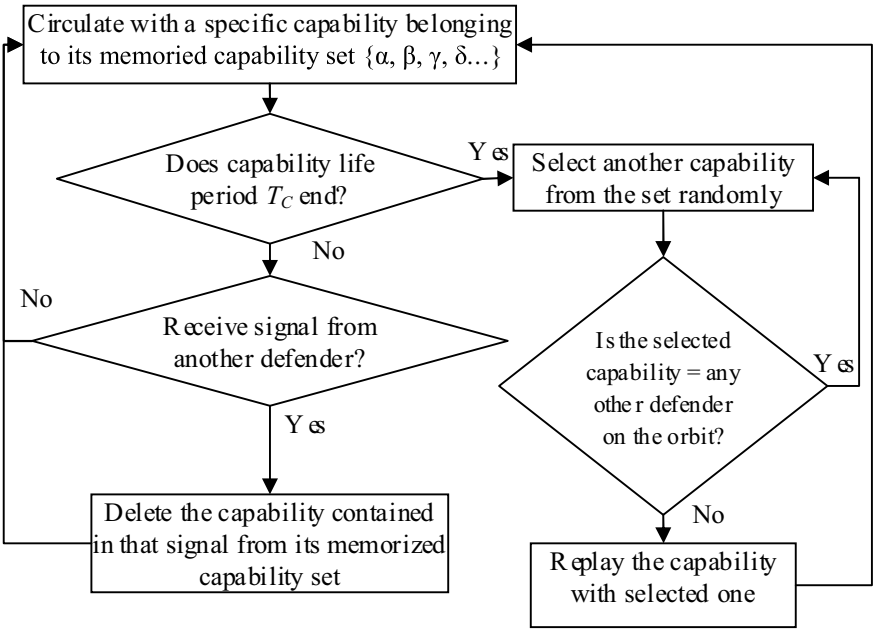


Fig. 5. The concentration-dependent formation control algorithm

Capability complement and manipulation process **-** In order to have the ability to intercept as many attackers as possible, it requires a diversity of the defenders' capabilities. The diversity of AIS-based defenders' capability is achieved by a dynamic and coordinated capability complement and manipulation process. Each AIS-based defender keep a dynamic memory of a capability set {α, β, γ, δ...} that contains all the possible types of attackers that appear. There are two immune-inspired mechanisms used in this process: first, each AIS-based defender maintains its capability for a predefined period T_C (T_C is counted from the moment when the defender transitions to the circulation mode) and its capability will be replaced by another randomly selected capability in the memorized capability set at the end of T_C if it does not react to any attacker in this period. Secondly, when an AIS-based defender receives the signal from another defender who is going to react to an attacker, it deletes the capability of reacting to that attacker from its memorized capability set. Furthermore, in the capability manipulation process, the AIS-based defender ensures that its capability is different from those of other defenders on the orbit, so that the capabilities of defenders on the orbit complement each other. The control logic of the capability complement and manipulation process is shown in Fig. 6.

In reaction mode, the AIS-based defender first sends a signal with information on the type of the attacker to its neighbors, and then approaches the attacker along a calculatedstraight line trajectory at its maximum velocity v_d^{max}. When the attacker is in its intercept region, it returns to the orbit at the same velocity along the straight line linking it with the closest point on the orbit. The overall control logic of an AIS-based defender is shown in Fig. 7.

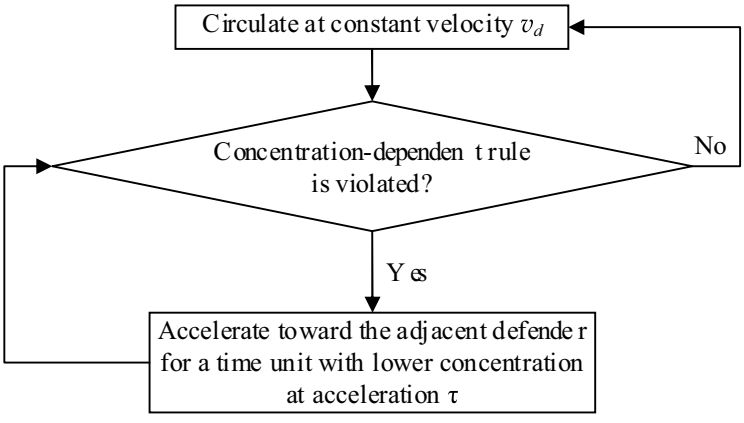


Fig. 6. The capability complement and manipulation process

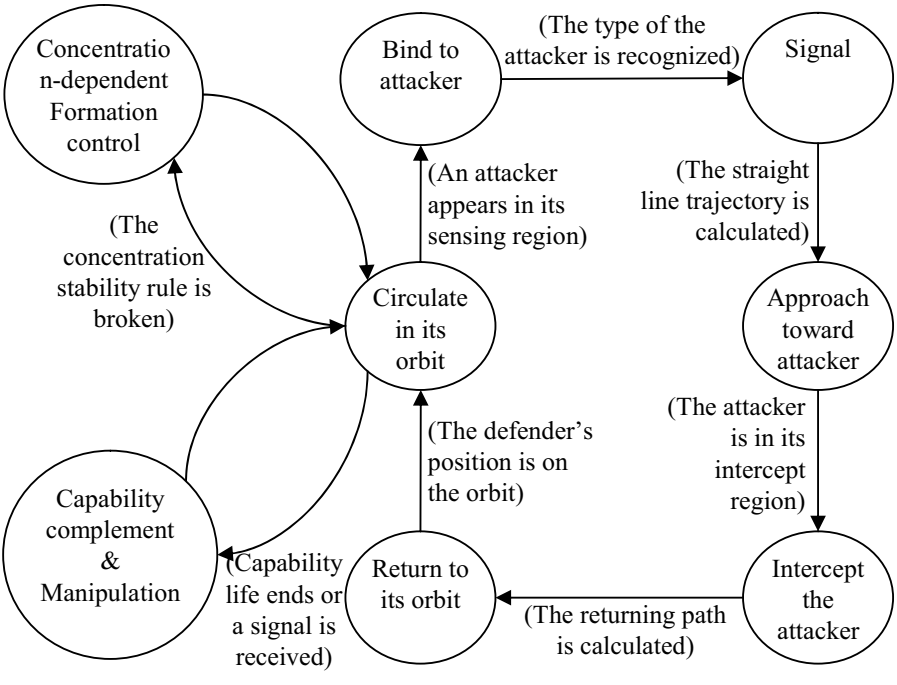


Fig. 7. Control logic for an AIS-based defender

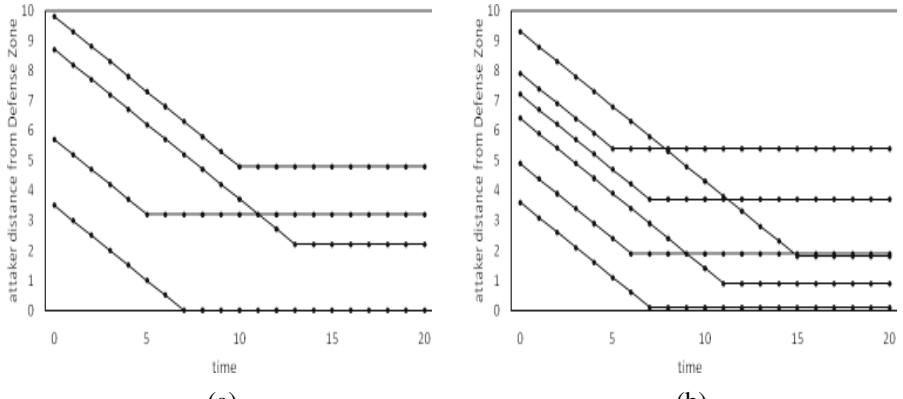
5 Simulation and Discussion

In this section, we demonstrate the effectiveness of the proposed control framework by solving randomly generated instances of N_D-on-N_A Defensive Drill problems. To undertake a simple but illustrative simulation, we consider instances with defender

set of size three (N_D=3) and attacker sets of size N_A = 1, 2, 3, 4, 5, and defender set of size four (N_D=4) and attacker sets of size N_A = 3, 4, 5, 6, 7. The size of these problem instances is the same with those adopted in RD' Andrea [32]. We compare the computational complexity and the effectiveness of the proposed framework with those of RD' Andrea [32], which models the multi-vehicle cooperative control problem in a MILP framework, and is widely regarded as the state-of-the-art for Robo-Flag problems.

For each of the 10 N_D-on-N_A Defensive Drill problems stated above, 10 random instances were generated. Each instance is generated by randomly picking parameters from a uniform distribution over the intervals defined below. We take the playing field to be circular with radius R=11, and we set the radius of the Defense Zone R_{dz} = 1. The intervals used to generate the instances are $r_a \in$ [4.5, 11] and $v_d \in$ [1.0, 1.5]. We also set R_{inc}=0.5, v_a=0.5, $T_C = 2\pi * (1+0.5) / v_d$. We assume that the capability set that each AIS-based defender keeps in memory at the beginning of the play contains 5 types of capabilities for the problems with N_D=3, and 7 types of capabilities for the problems with N_D=4. They are given by { α, β, γ, δ, ε } and { α, β, γ, δ, ε, σ, τ} respectively. For each problem instance, the attacker set should be a subset of the corresponding capability set kept in memory of each AIS-based defender at the begining of the play.

In Fig. 8, we plot the attacker's distance from the boundary of the Defense Zone versus time for two problem instances: (a) N_D=3, N_A = 4; (b) N_D=4, N_A = 6. The distance is calculated by (r_a - R_{dz}). As these figures show, in the instance with 3 defenders and 4 attackers, the attacker with an initial distance of 3.5 enters the Defense Zone. In the instance with 4 defenders and 6 attackers, all attackers were intercepted successfully before entering the Defense Zone. Among these attackers, those with an initial distance of 3.6 was intercepted when they were 0.1 units away from the Defense Zone.



(a) (b)

Fig. 8. The solutions to two instances of the Defensive Drill. For Fig (a), N_D=3 and N_A = 4. For Fig (b), N_D=4 and N_A = 6. In Fig (a), an attacker enters the Defense Zone. In Fig (b), the defenders intercept all attackers from the Defense Zone.

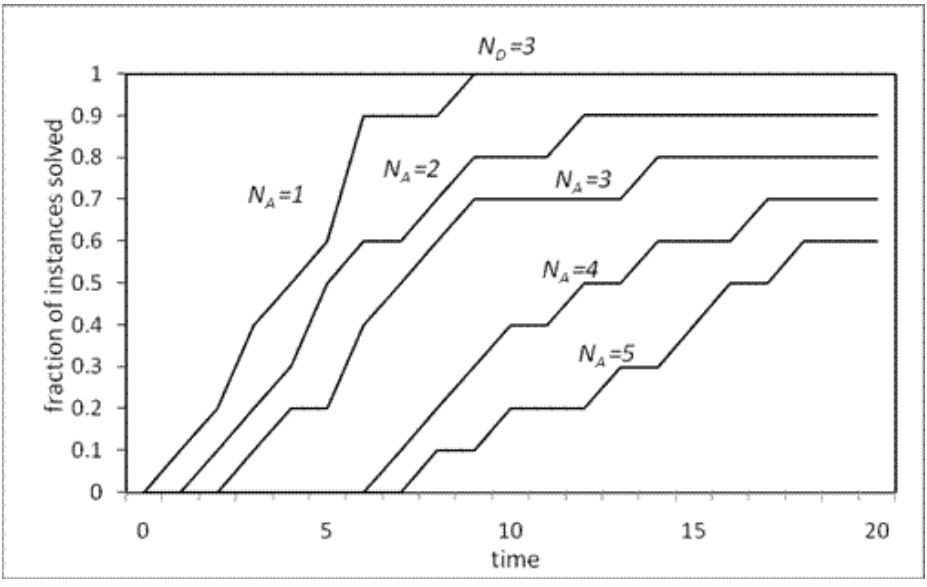


Fig. 9. Fraction of instances solved versus time for instances with a defender set of size N_D=3 and attacker sets of size N_A = 1, 2, 3, 4, 5

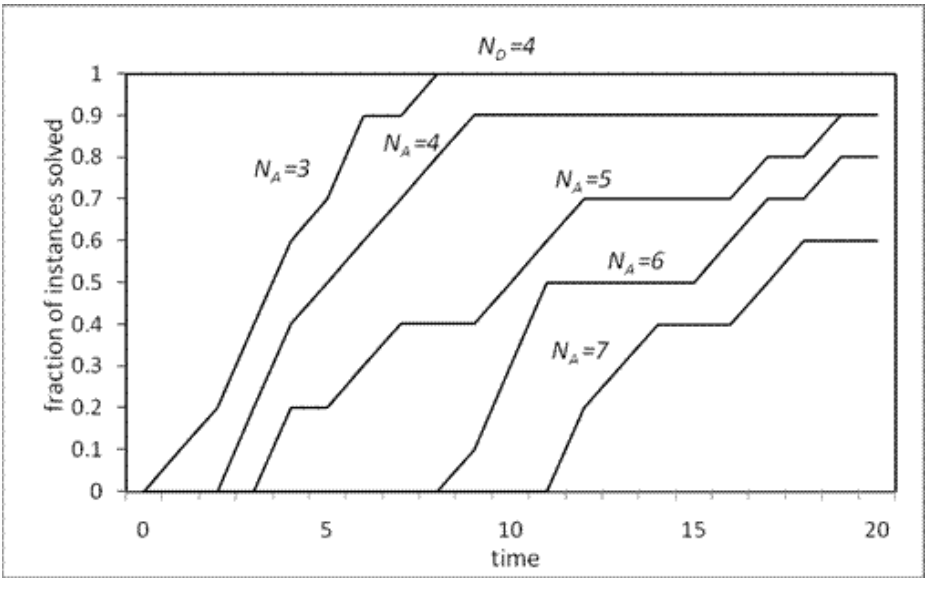


Fig. 10. Fraction of instances solved versus time for instances with a defender set of size N_D=4 and attacker sets of size N_A = 3, 4, 5, 6, 7

In Fig. 9., we plot the fraction of instances solved versus time for the Defensive Drill problems with a defender set of size N_D=3 and attacker sets of size N_A = 1, 2, 3, 4, 5. To generate each curve, 10 random instances were solved. In Fig. 10., we plot the fraction of instances solved versus time for the Defensive Drill problems with a defender set of size N_D=4 and attacker sets of size N_A = 3, 4, 5, 6, 7. Togenerate each curve, 10 random instances were solved. As shown in these figures, 60% of the instances can be solved (If all attackers are intercepted before entering the Defense Zone, it is considered solved) within 18 time units under the control and coordination of our proposed framework.

In Table 1., we compare the effectiveness of the proposed AIS-based framework with that of the MILP framework proposed in Andrea's work [32]. The measurement index adopted is the computation time necessary to solve 50% of the randomly generated instances for cases where the number of attackers is larger than the number of defenders. From the table, it can be seen that the proposed AIS-based framework is less computationally intensive than the MILP framework proposed by Andrea for cases where the size of attacker set is larger than that of defender set. Furthermore, the scalability of the proposed AIS-based framework is much better than that of the MILP framework.

Table 1. Comparison of simulation results of the proposed AIS-based framework and the Andrea's MILP framework

Frameworks / Problem Cases	AIS-based framework	MILP framework
N_D =3 and N_A = 4	50% of the instances solved in 13 time units	50% of the instances solved in 22 time units
N_D =3 and N_A = 5	50% of the instances solved in 16 time units	50% of the instances solved in 65 time units
N_D =4 and N_A = 5	50% of the instances solved in 9 time units	50% of the instances solved in 10 time units
N_D =4 and N_A = 6	50% of the instances solved in 11 time units	50% of the instances solved in 16 time units
N_D =4 and N_A = 7	50% of the instances solved in 17 time units	50% of the instances solved in 48 time units

6 Conclusion and Future Work

This paper presents a cooperative control framework for networked multi-agent systems, named AIS-based Cooperative Reaction Framework (ACRF), based on the meta-dynamics theories of lymphocyte repertoires in the adaptive immune system. In the framework, a collection of AIS-based agents engages in cooperative search and defense task in N_D-on-N_A RoboFlag Drill problems. The main featured mechanisms employed in the ACRF include the concentration-dependent formation control algorithm and capability complement and manipulation process.

The paper also presents a simulation study on ACRF to solve randomly generated instances of N_D-on-N_A Defensive Drill problems. The simulation results successfully demonstrated the effectiveness of the proposed ACRF for team control in RoboFlag

Drill. Future study will aim to examine the performance of this framework with respect to the rate of convergence and computational complexity through solving randomly generated instances of N_D-on-N_A Defensive Drill problems with a larger sets of attackers and defenders.

Acknowledgments. The work described in this paper was partly supported by the Research Grant Council of the Hong Kong Special Administrative Region, PRC under the GRF Project No. HKU713707E.

References

1. Forrest, S., et al.: Self-nonself discrimination in a computer (1994)
2. Jerne, N.: Towards a network theory of the immune system (1974)
3. Burnet, F.: The clonal selection theory of acquired immunity. Vanderbilt University Press, Nashville (1959)
4. Hunt, J., Cooke, D.: Learning using an artificial immune system. Journal of network and computer applications 19(2), 189–212 (1996)
5. Timmis, J., Neal, M., Hunt, J.: An artificial immune system for data analysis. Biosystems 55(1-3), 143–150 (2000)
6. Kephart, J.: A biologically inspired immune system for computers. MIT Press, Cambridge (1994)
7. Jun, J., Lee, D., Sim, K.: Realization of cooperative strategies and swarm behavior in distributed autonomous robotic systems using artificial immune system (1999)
8. Lau, H.Y.K., Wong, V.W.K.: An Immunity-Based Distributed Multiagent-Control Framework. IEEE Transactions on systems, man, and cybernetics-part A: systems and humans 36(1), 91–108 (2006)
9. Ko, A., Lau, H.Y.K., Lau, T.L.: An Immuno Control Framework for Decentralized Mechatronic Control. Int. Journ. of Unconventional Computing 1, 255–280 (2005)
10. Mehr, R.: Modeling the Meta-Dynamics of Lymphocyte Repertoires. Archivum Immunologiae et Therapiae Experimentalis 49, 111–120 (2001)
11. Somayaji, A., Hofmeyr, S., Forrest, S.: Principles of a Computer Immune System. In: 1997 New Security Paradigms Workshop, Langdale, Cumbria UK, pp. 75–82 (1997)
12. Tonegawa, S.: Somatic generation of antibody diversity. Nature 302, 575–581 (1983)
13. Earl, M., D'Andrea, R.: A study in cooperative control: The RoboFlag drill. In: Proceedings of the American Control Conference, Anchorage, Alaska (2002)
14. Vecchio, D.D.: Cascade estimators for systems on a partial order. Systems & Control Letters 57, 842–850 (2008)
15. Janeway, C., et al.: Immunobiology: the immune system in health and disease (1997)
16. Hofmeyr, S., Forrest, S.: Architecture for an artificial immune system. Evolutionary Computation 8(4), 443–473 (2000)
17. Antsaklis, P.J., Baillieul, J.: Special Issue on Technology of Networked Control Systems. Proceedings of the IEEE, Special Issue on Networked Control Systems 95(1) (2007)
18. Bullo, F., Cortés, J., Piccoli, B.: Special Issue on Control and Optimization in Cooperative Networks. SIAM Journal on Control and Optimization 48(1) (2009)
19. Watts, D.J.: Connections: A twenty-first century science. Nature 445(489) (2007)
20. Alderson, D.L.: Catching the "Network Science" Bug: Insight and Opportunity for the Operations Researcher. Operations Research 56(5) (2008)

21. Amaral, L.A.N., Uzzi, B.: Complex Systems—A New Paradigm for the Integrative Study of Management, Physical, and Technological Systems. Management Science 53(7), 1033–1035 (2007)
22. The NRC report, Network Science (2005)
23. DeGroot, M.H.: Reaching a consensus. Journal of American Statistical Association 69(345), 118–121 (1974)
24. Fax, J.A., Murray, R.M.: Information flow and cooperative control of vehicle formations. IEEE Trans. on Automatic Control 49(9), 1465–1476 (2004)
25. Marden, J.R., Arslan, G., Shamma, J.S.: Connections Between Cooperative Control and Potential Games. IEEE Transactions on Systems, Man and Cybernetics. Part B: Cybernetics (2009)
26. Hanaki, N., et al.: Cooperation in Evolving Social Networks. Management Science 53(7), 1036–1050 (2007)
27. Shoham, Y., Leyton-Brown, K.: Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations 2009. Cambridge University Press, Cambridge (2009)
28. Dias, M.B., et al.: Market-Based Multirobot Coordination: A Survey and Analysis, Robotics Institute, Carnegie Mellon University (2005)
29. Gerkey, B., Mataric, M.: Sold!: Auction methods for multirobot coordination. IEEE Transactions on Robotics and Automation 18(5), 758–768 (2002)
30. D'Andrea, R., Babish, M.: The RoboFlag testbed. In: Proceedings of the American Control Conference (2003)
31. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, New York (2002)
32. Earl, M., D'Andrea, R.: Multi-vehicle cooperative control using mixed integer linear programming. Arxiv preprint cs.RO/0501092 (2005)

Managing Diversity on an AIS That Solves 3-Colouring Problems

María-Cristina Riff and Elizabeth Montero

Department of Computer Science,
Universidad Técnica Federico Santa María, Valparaíso, Chile
{María-Cristina.Riff}@inf.utfsm.cl, {Elizabeth.Montero}@inf.utfsm.cl

Abstract. Constraint Directed Network Artificial Immune System is an artificial immune algorithm, recently proposed, to solve constraint satisfaction problems. The algorithm has shown to be able to solve hard instances. However, some problems are still unsolved using this approach. In this paper, we propose a method to improve the search done by the algorithm. Our method can be included in other immune algorithms which manage constraints. The tests are carried out to solve very hard instances randomly generated of 3-colouring problems. The results show that using our method, the algorithm is able to solve more problems in less execution time.

1 Introduction

A (finite domain) Constraint Satisfaction Problem (CSP) consists of a set of variables, their related domains, and a set of constraints between them. The goal is to find a value for each variable, from their respective domains, such that the constraints are satisfied [20], [21], [18].We are interested in the stochastic approaches, specifically artificial immune systems to solve CSPs. In the following section, we define a CSP and its components. The Constraint Directed Network Artificial Immune System (CD-NAIS) has been recently proposed for solving constraint satisfaction problems, [12]. We briefly review the algorithm in section 3. CD-NAIS has given very good results. However, more complex problems where some constraints are more difficult to be satisfied than others are still unsolved by CD-NAIS. Because it is known that using an appropriate heuristic algorithm can make a tremendous difference in the time required to solve a CSP [16], in the constraint research community, we have analysed both the performance and the behaviour of CD-NAIS, in order to define a method that can help it solve more complex CSP problems. This analysis allows us to propose, in section 4, a new Build Network procedure, which is focused on constraints satisfaction diversity more than instantiation diversity, as does the original procedure. In order to evaluate our method we have done many tests using hard instances of the 3-colouring graph. The results that are discussed in section 5, are very encouraging and show that CD-NAIS, outperforms the original algorithm, using our approach.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 248–259, 2009.

The contributions of this paper are:

- A Build Network procedure that helps CD-NAIS to explore new parts of the search space, where other constraints can be satisfied.
- Improve CD-NAIS in order to be able to solve problems where the constraints have a diverse degree of difficulty.
- To show that an immune artificial algorithm can solve very hard problems well, using its particular characteristics as a negative selection and diversity.

2 Preliminaries

Here we will give the basic notions on a CSP. We restrict our attention to binary CSPs, in which the constraints involve two variables. It is possible to convert a CSP with many constraints to another equivalent binary CSP [17].

2.1 Notions on CSP

A *Constraint Satisfaction Problem* (CSP) is composed of a set of *variables* $V = \{X_1, \ldots, X_n\}$, their related *domains* $D_1, \ldots, D_n$ and a set θ containing η *constraints* on these variables. The domain of a variable is a set of values to which the variable may be instantiated. The domain sizes are $m_1, \ldots, m_n$, respectively,and we let $\mathbf{m}$ denote the maximum of the m_i. Each variable X_j is *relevant*, to a subset of constraints $C_{j_1}, \ldots, C_{j_k}$ where $\{j_1, \ldots, j_k\}$ is some subsequence of $\{1, 2, \ldots, \eta\}$. A constraint which has only one relevant variable is called a *unary constraint*. Similarly, a *binary constraint* has exactly two relevant variables. If two values assigned to variables that share a constraint are not among the acceptable value-pairs of that constraint, this is an *inconsistency* or constraint violation. Because CD-NAIS strongly uses the topology of the constraints for searching, we require the following definitions to explain how the algorithm works.

Definition 1. *(Constraint Arity)*
We define the Constraint Arity *for a constraint* C_α*,* a_α*, as the number of relevant variables for* C_α*.*

Definition 2. *(Constraints Graph)*
Given a binary CSP, we define a Constraints Graph *G=(V,E), such that the set of vertices V corresponds to the CSP variables. An edge that connects two vertices* X_i *and* X_j *exists if and only if there is at least one constraint for which both* X_i *and* X_j *are relevant.*

Definition 3. *(Constraints Involved Variables)*
We define the Constraints Involved Variables **CI** *as a mapping from a n-tuple of values* $(CI_1, \ldots, CI_n) \rightarrow DCI_i \in [0, number\ of\ constraints], \forall i$*, such that for a variable* X_i*,* CI_i *is the number of constraints for which* X_i *is a relevant one.*

Definition 4. *(Violation Involved Variables)*
We define the Violation Involved Variables **VI** *as a mapping from a n-tuple of values* $(VI_1, \ldots, VI_n) \rightarrow DVI_i \in [0, number\ of\ constraints], \forall i$*, such that for a variable* X_i*,* VI_i *is the number of violated constraints for which* X_i *is a relevant one.*

Definition 5. *(Instantiation)*
An Instantiation **I** *is a mapping from a n-tuple of variables* $(X_1, \ldots, X_n) \rightarrow D_1 \times \ldots \times D_n$*, such that it assigns a value from its domain to each variable in V,e.g.* $(X_1, \ldots, X_n) \rightarrow (14, 18, \ldots, p, q)$

A solution consists of an Instantiation which does not violate any constraint.

3 Description of CD-NAIS

Constraint Directed Network Artificial Immune System (CD-NAIS) is an immune algorithm which strongly takes into account the constraints graph in order to guide its search. It uses an antigen which is an array of size n where each element i corresponds to the CI_i value as it has been defined above. Thus, an antigen represents static information about the problem constraints. For each variable, the antigen value is the number of constraints for which this variable is a relevant one. The B-cells antibodies are composed of two structures. The first one is an array of instances for all the variables (structural antibody), and the second one is an array corresponding to the number of violated constraints where each variable is involved (conflicting antibody). B-cells are maintained into a memory of B-cells. At each iteration, the algorithm uses a Clonal Selection to select a set of B-cells from the memory, taking into account the constraints violations and the participation of each variable into the violations. A Clonal Expansion generates clones of the selected B-cells following a hypermutation procedure, which is a local-search to repair the instantiation part of the antibody. Then the Build Network procedure inserts new B-cells from the clones set into the memory such that the new B-cells do not violate more constraints than the ones already in the memory, and the instantiations of the variables of the new B-cells are as far as possible different from those already in the memory (more diversity).

3.1 Affinities in CD-NAIS

CD-NAIS has two types of affinity. The first one is to measure the affinity between the antigen and a conflicting antibody (violation involved). The second one is to measure the affinity between two structural antibodies (instantiations).

Affinity between Antigen and a Conflicting Antibody. CD-NAIS uses a function named A_{csp} to compute the affinity between the antigen and a conflicting antibody. This function does not only prefer a pre-solution with a minimal number of violated constraints, but it also takes into account how hard for the algorithm to repair this pre-solution could be. This is done by including a conflict dispersion measure. Thus, given two pre-solutions which satisfy the same

number of constraints, the algorithm prefers the one with the smallest number of variables (nodes) involved in the constraints violations. The algorithm is searching for pre-solutions having both a reduced number of constraints violations and a reduced number of involved variables.

Affinity between Structural Antibodies. Roughly speaking this affinity is a measure between two instantiations where both are candidates to be a pre-solution of the CSP. The key idea is to have a criteria which prefers one of them. Given two pre-solutions with the same number of violated constraints, CD-NAIS prefers the pre-solution which is a different instantiation to the others found previously by the algorithm. It is searching for instantiation diversity. This affinity is used in the Build Network procedure. In this procedure some B-cells are selected to be included in memory by preferring those having a larger instantiation diversity respect to the B-cells already in memory.

The algorithm CD-NAIS is shown in figure 1.

```
Algorithm CD-NAIS(CSP) returns memory B-cells
Begin
    Ag ← Determine constraint graph connections(CSP, n);
    Initialise B-cells
    For i ← 1 to bcells_number do                    (1) Antigen Presentation
        Compute affinity value A_csp(B-cells[i])
    End For
    j ← 1;
    While (j ≤ MAX_ITER) or (not solution) do
        Select a set of B-cells by Roulette Wheel    (2) Clonal Selection
        Generate Clones of the selected B-cells      (3) Clonal Expansion
        Hypermutate Clones                           (4) Affinity Maturation
        For k ← 1 to CLONES_NUM do
            Compute affinity value A_csp(Clones[k])
        End For
        B-cells ← build_network(CLONES);
        B-cells ← metadynamics(B-cells);
    End While
    Return B-cells;
End
```

Fig. 1. CD-NAIS Pseudocode

4 New Build Network - Searching Constraints Diversity

We have analysed in detail, the behaviour and the performance of CD-NAIS when it is solving difficult problems, where some constraints are more difficult to be solved than others. From the behaviour of the algorithm for these problems we have observed that the unsatisfied constraints are the same among B-cells in memory, whereas there is a good diversity of the instantiation part or structural antibody. That means that the conflicting antibodies in memory are quite similar. In this case, the algorithm is not able to explore new parts of the search space

where other constraints can be satisfied. In most of the CSPs, some constraints are more difficult to be satisfied. Taking into account this issue we propose to change the point of view of managing the B-cells diversity in memory. We propose to change from having a diversity of instantiations in memory, to having a diversity of the constraints that are being satisfied by the B-cells in memory. We can have a diversity of values in memory, but the algorithm can always be satisfying the same constraints.

In our method, two conflicting antibodies have a high affinity level when they are quite different in terms of the constraints that they are violating. The idea is to quantify how similar two conflicting antibodies are, not only in the number of unsatisfied constraints but in which variables are involved in the violations. To compute this interaction our strategy uses the Hamming distance between conflicting antibodies.

Definition 6. *Given two B-cells B_i and B_j and their conflicting antibodies Ab_{c_i} and Ab_{c_j} respectively, we define the* network conflicting affinity *as the hamming distance between the conflicting antibodies of the B-cells. That is:*

$$\textit{network conflicting affinity} = \textit{Hamming-distance}(Ab_{c_i}, Ab_{c_j}) \tag{1}$$

Our Build Network procedure prefers to include in memory, those new B-cells having a larger value of the network conflicting affinity, among those B-cells already in memory. This can be interpreted as the algorithm is now focused on putting B-cells, which are satisfying different constraints (constraints satisfaction diversity), in memory.

The new Build Network procedure uses the immune network concepts to generate a set of conflicting diverse memory B-cells. This procedure, described in figure 2, uses a set of hypermutated clones ordered from the highest to the lowest affinity A_{csp} to generate a set of memory B-cells.

A suppression counter is defined in order to evaluate how similar a clone is to the B-cells in memory. The key idea is the constraints diversity. The selection process is done by including clones that are quite different from the B-cells that are already in memory. It begins with the clones without suppression condition. At each step, the suppression condition is updated considering the new clone included in memory.

5 Experimental Comparison

In this section, we experimentally evaluate and compare the CD-NAIS implementations: The original version and this using our new build network procedure.

5.1 Experimental Setup

The hardware platform for the experiments was a PC Pentium IV Dual Core, 3.4Ghz with 512 MB RAM under the Linux Mandriva 2008 operating system. The algorithm is implemented in C. The code for CD-NAIS is available on a website[1].

[1] http://www-sop.inria.fr/orion/personnel/Marcos.Zuniga/CSPsolver.zip

```
procedure New Build Network (CLONES, n)
Begin
While (num_added_b_cells < b_cells_to_be_selected) do
   For j ← 0 to clones_number − 1 do
      {H = HA^a(conflicts − clone[j], conflicts − MEMORY_BCELLS);
      clone[j].network_conflicting_affinity = H;
      If clone[j].network_conflicting_affinity < ϵ then
        suppression-condition for clone j;
      else
        {MEMORY_BCELLS[num_added_b_cells] ← clone[j];
        num_added_b_cells ← num_added_b_cells + 1;}
   End For
End While
Return MEMORY_BCELLS;
End
```

[a] Hamming Distance between

Fig. 2. Pseudo-code for *New Build Network* procedure

5.2 Test Suite

We repeat the same tests that have been done for evaluating the original version of CD-NAIS. The goal of the tests is to compare the performance of CD-NAIS in front of CSPs as the well-known 3-colouring problem. Applications of the 3-colouring problems are scheduling, frequency assignment, or register allocation. We have used the Joe Culberson library[2] to generate random graphs to be coloured. These graphs have at least one solution.

For these tests, we have generated graphs with a sparse topology, as they are known to be the most difficult ones to solve. For each number of constraints from $\{60, 90, 120, 150, 180, 210, 240, 360, 400, 600, 800, 1000, 1200\}$ we have generated 500 random 3-colouring graph problems. In order to discard the *easy problems* we have applied DSATUR [24]. DSATUR is a greedy like algorithm, one of the best algorithms to solve this kind of problems.

DSATUR tries to color a graph G=(V,E) searching for using a minimum number of colours, respecting the adjacency constraints between nodes. The order in which nodes are chosen to be instantiated is determined dynamically, based on the number of colours that cannot be used because of conflicts with previously coloured nodes. Nodes with the fewest available colours are coloured first. Pseudo code is shown in figure 3. The performance of DSATUR depends on the initial order of nodes which are randomly generated or using by some heuristics. For the most difficult 3-colouring problems DSATUR is able to find solutions using 4 colours. In the transition phase the choices at each node are very similar, thus for DSTAUR is difficult to discriminate which is the best option.

[2] http://web.cs.ualberta.ca/joe/

Begin
Give an initial ordering of nodes as $V = \{v_1, v_2, ...v_i, ...v_n\}$;
Find a largest clique V^* of G, assign each vertex in V^* a distinct colour class;
$V = V - V^*$;
While(V != NULL)
    Find a vertex v in V, which is adjacent to the largest number of distinctly coloured nodes
    Assign v to the lowest indexed colour class that contains no nodes adjacent to v;
    If(no available colour class) create a new colour class for v;
        Move v out of V;
    endIf
endWhile
return colour classes.
End

Fig. 3. DSATUR pseudocode

Finally, the problems used for testing are 100 problems in the transition phase, for each category, selected from those that DSATUR can not solve. We run the algorithm using 5 different initial seeds for the random numbers generator to solve the same problem. We have modified the parameters found for tuning CD-NAIS. We have obtained better results just using 3 clones and 10 B-cells, instead of 100 clones and 5 B-cells reported in the original approach. The parameter values used for both versions are shown in figure 4.

Parameters	CD-NAIS	CD-NAIS-New-Tuned
n_1	0.3	0.3
n_2	0.4	0.4
ϵ	0.4	0.4
$B - cells$	5	10
$clones$	100	3

Fig. 4. Parameter values for CD-NAIS and CD-NAIS-New-Tuned

In figures 5 and 6, we report the percentage of problems solved by CD-NAIS (original version), CD-NAIS-New-Tuned using the new parameter values and CD-NAIS-New-Build-Network using our new method for diversity. All of them using 5000 iterations. In figure6 we show the percentage of additional problems solved using our technique in comparison to the two other versions of CD-NAIS: The original one and the re-tuned one. We consider that a problem is solved if at least one of the five executions has solved the problem. For instance, for the problems with 800 constraints using our new technique CD-NAIS has been able to solve 71% more problems than the re-tuned version and 96% more than the original approach. The more constraints the problem has, the more problems are solved using our new procedure compared to the original re-tuned CD-NAIS.

Number of Constraints	% of Problems Solved CD-NAIS	% Problems Solved CD-NAIS-New-Tuned	% Problems Solved CD-NAIS-New-Build-Network
60	96	100	100
90	92	100	100
120	88	100	100
150	78	100	100
180	71	100	100
210	63	100	100
240	56	100	100
360	42	99	100
400	0	100	100
600	0	84	100
800	0	25	96
1000	0	6	75
1200	0	1	30

Fig. 5. Comparison of the Percentage of Problems Solved for the 3-colouring problem

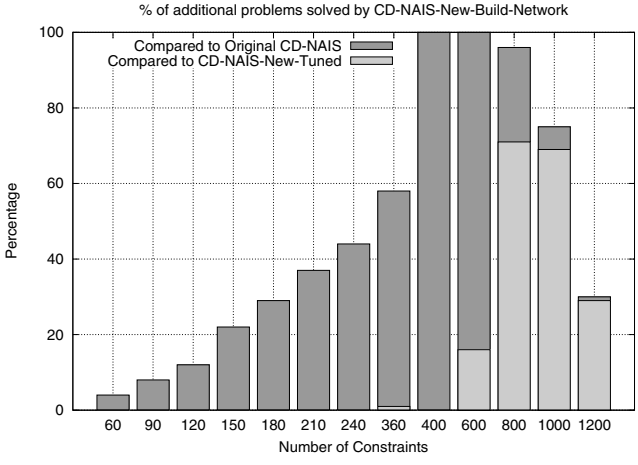


Fig. 6. Different problems tested, comparison of: % Problems Solved

In figure 7 we show the percentage of graphs solved for each connectivity. We do not report the results using the original tuned CD-NAIS here because we have clearly obtained better solutions using the new parameter values. For each connectivity there are 500 runs, corresponding to 5 executions with different seeds for each of the 100 problems. In figure 8, we show the percentage of additional runs that has solved a problem using 5 different initial seeds, compared to the re-tunned version of CD-NAIS. As we expected we observed that different seeds make an enormous difference in the quality of the solutions found by the algorithm. CD-NAIS using our method, has obtained more successful executions (find a solution) than the original re-tuned one.

Number of Constraints	% of Runs Solved CD-NAIS-New-tuned	% Runs Solved CD-NAIS-New-Build-Network
60	100	100.0
90	99.6	100.0
120	100	99.8
150	98.8	98.6
180	99.2	99.6
210	98.2	97.2
240	94.4	96.4
360	74.2	86.2
400	88.2	95.6
600	36.0	81.6
800	6.2	50.6
1000	1.2	22.8
1200	0.2	7.8

Fig. 7. Comparison of the Percentage of Executions Solved for the 3-colouring problem

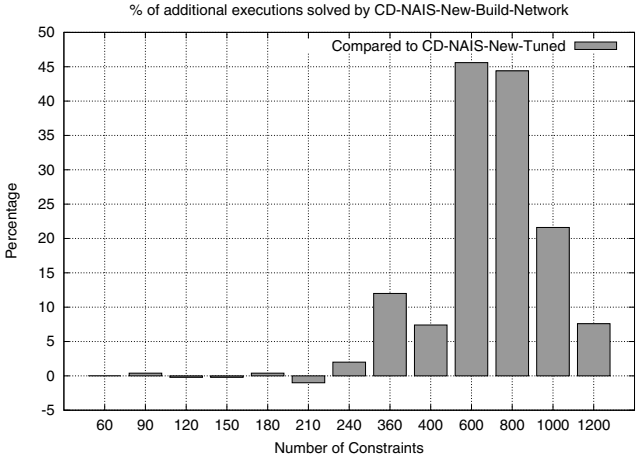


Fig. 8. Different problems tested, five executions for each problem, comparison of: % Executions Solved

The modification in the criteria of memory diversity has allowed to improve the performance of CD-NAIS when it is forefront of solving the most difficult 3-colouring problems. Furthermore, as is for all other known solvers, the problem in the transition phase is also the most difficult one for CD-NAIS using both strategies.

6 Related Work

The most popular method in stochastic approaches to solve a CSP is *min-conflicts hill-climbing* [22]. The success of this method is for both simplicity

and performance. This method creates a complete, but inconsistent assignment and then repairs constraint violations until a consistent assignment is achieved. This method is guided by a simple ordering heuristic for repairing constraint violations. Minton et al. [22] empirically demonstrated that *min-conflicts* is considerably more efficient than traditional constructive backtracking methods. Stochastic methods are proposed, in general, to solve large scale constraint satisfaction problems. In the bio-inspired computation community, some approaches have also been proposed to tackle CSPs with success, such as evolutionary algorithms [5], [6], [9], [15], [11], [3], ants algorithms [14], immune algorithms [12]. In order to improve the search performance for the hard problems where some constraints are more difficult to be solved, the researchers have proposed many different strategies [23].In general, these problems are tackled given a higher penalty to constraints that have not been satisfied for a while. This implies defining new parameters and criteria as a penalty value, and how and when to change this penalty, as sometimes different penalties are also included, ordered by an unsatisfied ranking of constraints. Our new procedure does not require more parameters.

7 Conclusions

The artificial immune systems have some interesting characteristics from the computational point of view: pattern recognition, affinity evaluation, immune networks and diversity. In this paper, we focused our attention on the concept of diversity and how to guide the search of the algorithm to satisfy more constraints. The B-cell structure is useful in determining both the solution to the problems and also in identifying the involved variables in the conflicts. The conflicting antibody is used by the algorithm to guide the reparation of the solutions (hypermutation process), giving more priority to the variables involved in a higher number of conflicts. Now, it is also used to learn and keep in memory those B-cells that satisfy different constraints but have a good constraint satisfaction level. For the problems in the hardest zone, CD-NAIS-New-Build-Network has solved, on average, 14% more problems than the CD-NAIS-New-Tuned and 47.4% more problems than the original one.

8 Future Work

A promising research area is to incorporate some other heuristics coming from the constraint research community into the algorithm. We also expect that recombining B-cells could help the algorithm to access other dimensions of the search space.

Acknowledgements

This work has been supported by Fondecyt Project 1080110.

References

1. de Castro, L.N., Timmis, J. (eds.): Artificial Immune Systems: A New Computational Intelligence Approach. Springer, London (2002)
2. Cheeseman, P., Kanefsky, B., Taylor, W.: Where the really hard problems are. In: Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI 1991), pp. 163–169 (1991)
3. Craenen, B.G.W., Eiben, A.E., van Hemert, J.I.: Comparing evolutionary algorithms on binary constraint satisfaction problems. IEEE Transactions on Evolutionary Computation 7(5), 424–444 (2003)
4. Dasgupta, D. (ed.): Artificial Immune Systems and Their Applications. Springer, Heidelberg (2000)
5. Dozier, G., Bowen, J., Homaifar, A.: Solving constraint satisfaction problems using hybrid evolutionary search. IEEE Transactions on Evolutionary Computing 2(1), 23–33 (1998)
6. Eiben, A.E., van Hemert, J.I., Marchiori, E., Steenbeek, A.G.: Solving binary constraint satisfaction problems using evolutionary algorithms with an adaptive fitness function. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 201–205. Springer, Heidelberg (1998)
7. Garrett, S.M.: Parameter-free, adaptive clonal selection. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2004), vol. 1, pp. 1052–1058 (2004)
8. Mackworth, A.K.: Consistency in network of relations. Artificial Intelligence 8, 99–118 (1977)
9. Marchiori, E.: Combining constraint processing and genetic algorithms for constraint satisfaction problems. In: Proceedings of the 7th International Conference on Genetic Algorithms (ICGA 1997), pp. 330–337 (1997)
10. Nannen, V., Eiben, A.: Relevance estimation and value calibration of evolutionary algorithm parameters. In: Proceedings of the Joint International Conference for Artificial Intelligence (IJCAI), pp. 975–980 (2007)
11. Riff, M.C.: A network-based adaptive evolutionary algorithm for csp. In: Metaheuristics: Advances and Trends in Local Search Paradigms for Optimisation, ch. 22, pp. 325–339. Kluwer Academic Publisher, Dordrecht (1998)
12. Riff, M.C., Zuniga, M.: Towards an immune system to solve csp. In: Proceedings of the IEEE Congress on Evolutionary Computation, Singapur, pp. 2837–2841 (2007)
13. Smith, B.M., Dyer, M.E.: Locating the phase transition in binary constraint satisfaction problems. Artificial Intelligence 81(1-2), 155–181 (1996)
14. Solnon, C.: Ants can solve constraint satisfaction problems. IEEE Transactions on Evolutionary Computation 6(4), 347–357 (2002)
15. Tsang, E.P.K., Wang, C.J., Davenport, A., Voudouris, C., Lau, T.L.: A family of stochastic methods for constraint satisfaction and optimization. In: Proceedings of the First International Conference on The Practical Application of Constraint Technologies and Logic Programming (PACLP), London, pp. 359–383 (1999)
16. Minton, S.: Automatically Configuring Constraint Satisfaction Programs: A case study. Constraints 1(1) (1996)
17. Rossi, F., Petrie, C., Dhar, V.: On the equivalence of Constraint-Satisfaction Problems, Technical Report ACT-AI-222-89, MCC Corp., Austin, Texas (1989)
18. Freuder, E.: A sufficient condition of backtrack-free search. J. ACM 29, 24–32 (1982)

19. Freuder, E.: The Many Paths to Satisfaction. In: Meyer, M. (ed.) Constraint Processing. LNCS, vol. 923, pp. 103–119. Springer, Heidelberg (1995)
20. Haralick, Elliott: Increasing tree search efficiency for Constraint Satisfaction Problems. Artificial Intelligence 14, 263–313 (1980)
21. Kumar: Algorithms for constraint satisfaction problems: a survey. AI Magazine 13(1), 32–44 (1992)
22. Minton, J., Philips, L.: Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. Artificial Intelligence 58, 161–205 (1992)
23. Minton, S.: Integrating heuristics for constraint satisfaction problems: A case study. In: Proceedings of the Eleventh National Conference on Artificial Intelligence (1993)
24. Brelaz: New methods to color vertices of a graph. Communications of the ACM 22, 251–256 (1979)

An Error Propagation Algorithm for Ad Hoc Wireless Networks

Martin Drozda, Sven Schaust, Sebastian Schildt, and Helena Szczerbicka

Simulation and Modeling Group
Dept. of Computer Science
Leibniz University of Hannover
Welfengarten 1, 30167 Hannover, Germany
{drozda,svs,hsz}@sim.uni-hannover.de, sebastian.schildt@reparts.org

Abstract. We were inspired by the role of co-stimulation in the Biological immune system (BIS). We propose and evaluate an algorithm for energy efficient misbehavior detection in ad hoc wireless networks. Besides co-stimulation, this algorithm also takes inspiration from the capability of the two vital parts of the BIS, the innate and the adaptive immune system, to react in a coordinated way in the presence of a pathogen. We demonstrate that this algorithm is also applicable in situations when a (labeled) data set for learning the normal behavior and misbehavior is unavailable.

1 Introduction and Motivation

Ad hoc and sensor wireless networks can become an object of attacks and intrusions. The motivation for attacking an ad hoc network can range from a desire to benefit from the network's services to an intent to make it non-functional. Faults that are a result of software or hardware failures can be equally damaging. Correcting the consequences of some faults or attacks might only be possible by a costly human intervention, or not at all. Even though secure protocols address issues connected with data integrity and user authentication, the experience with the Internet shows that flaws in these protocols are continuously being found and exploited [1].

This establishes the basic motivation for designing autonomous detection and response systems that aim at offering an additional line of defense to the employed secure protocols. Such systems should provide several layers of functionality including the following: (i) distributed self-learning and self-tuning with the aspiration to minimize the need for human intervention and maintenance, (ii) active response with focus on attenuation and possibly elimination of negative effects of faults or attacks on the network.

In many scenarios ad hoc and sensor networks are expected to be based on wireless devices with limited (battery) resources. In order to stimulate the survivability of such networks, it is essential that autonomous detection and response systems reflect these resource constraints.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 260–273, 2009.

Best current practices for misbehavior detection in ad hoc wireless networks are almost exclusively done on a domain knowledge basis; see [2] and references therein. Although such an approach allows to find a good predictor for a specific type of misbehavior, it fails to address a broader range of issues. Furthermore, the area of energy efficient misbehavior detection remains to be a challenging problem.

We assume that upon deployment of an ad hoc network, an enforcement of operational strategies in an energy efficient way is desired. Such operational strategies may impose performance limits in the form of e.g. a maximum data packet loss at a node. One possibility for determining whether a node keeps forwarding data packets is monitoring by other nodes. This is however a very costly approach since the monitoring nodes cannot enter a sleep mode to preserve their energy resources. It is therefore necessary that monitoring will be kept to a minimum.

Our aim was to design a system offering a reasonable tradeoff between energy efficiency and misbehavior classification performance. Our contribution is an algorithm that allows for energy efficient misbehavior detection that can also be applied to e.g. just deployed ad hoc networks. Such ad hoc networks have none or a very limited data basis for an efficient learning of the normal behavior and misbehavior. To stimulate energy efficiency, we exploit an inherent property of ad hoc networks in order to propagate the classification error among two parts of our detection system. These two parts are inspired by the role of the innate and adaptive immune systems and by their ability to communicate. Such error propagation allows for an adaptive approach to the costly explicit behavior monitoring by the participating nodes.

2 The Biological Immune System

The Biological immune system (BIS) [3] can quickly recognize the presence of foreign microorganisms in the human body. It is remarkably efficient, most of the time, in correctly detecting and eliminating pathogens such as viruses, bacteria, fungi or parasites. When confronted with a pathogen, the BIS often relies on a coordinated response from both of its two vital parts:

- the *innate system*: the innate immune system is able to recognize the presence of a pathogen or tissue injury, and is able to signal this to the adaptive immune system.
- the *adaptive system*: the adaptive immune system can develop during the lifetime of its host a specific set of immune responses and provide immunological memory. An immunological memory serves as a basis for a stronger immune response, should a pathogen re-exposure happen.

The form and amplitude of immune responses is pathogen dependent. In many forms of immune reactions, the innate immune system plays an important role in triggering a reaction from the adaptive immune system, and vice versa. A cell gets eliminated, if it was classified by the adaptive immune system as a pathogen, and at the same time, the innate immune system signals that the cell causes some damage to the human organism. Under this specific immune reaction, only

damage inflicting or *infectious* cells get eliminated by the BIS. This implies that a two-way communication, hereafter referred to as *co-stimulation*, between the innate and adaptive immune systems is necessary.

Communication capabilities within the BIS received an increased interest from the Artificial immune systems (AIS) community and evolved into an independent research direction. Several different types of danger, safe and amplifying signals were proposed within the Danger theory due to Aickelin et al. [4].

In our work, we were motivated by the ability of the BIS to act in a coordinated way when confronted with a pathogen. We show that co-stimulation in ad hoc networks can have very positive effects in stimulating energy efficiency.

3 Related Work

The pioneering work in adapting the BIS to networking has been done by Stephanie Forrest and her group at the University of New Mexico. In one of the first BIS inspired works, Hofmeyr and Forrest [5] described an AIS able to detect anomalies in a wired TCP/IP network. Co-stimulation was in their setup done by a human operator who was given 24 hours to confirm a detected attack.

Sarafijanović and Le Boudec [6] introduced an AIS for misbehavior detection in mobile ad hoc wireless networks. They used four different features based on the network layer of the OSI protocol stack. They were able to achieve a detection rate of about 55%; they only considered simple packet dropping with different rates as misbehavior. A co-stimulation in the form of a danger signal emitted by a connection source was used to inform nodes on the forwarding path about perceived data packet loss. Such signal could then be correlated with local detection results and suppress false positives.

An AIS for sensor networks was proposed by Drozda et al. in [7]. The implemented misbehavior was again simple packet dropping; the detection rate was about 70%.

Classification techniques proposed in [5,6,7] are based on negative selection, a learning mechanism applied in training and priming of T-cells in the thymus. In the computational approach to negative selection due to D'haeseleer et al. [8], a complement to an n-dimensional vector set is constructed. This is done by producing random vectors and testing them against vectors in the original vector set. If a random vector does not match anything, according to some matching rule in the original set, it becomes a member of the complement (detector) set. These vectors are then used to identify anomalies (faults/misbehavior). Timmis et al. [9] recently showed that negative selection is NP-complete, if the n-dimensional vectors are represented as bit-vectors and the matching/testing is done using the r-contiguous bits matching rule [8]. In general, the negative selection process, in the current interpretation, does not seem to have the potential to be used for misbehavior detection in ad hoc wireless networks.

An approach based on the Danger theory avoiding the inefficiency of the negative selection was proposed by Kim et al. in [10]. Several types of danger signals, each having a different function are employed in order to detect routing

manipulation in sensor wireless networks. The authors did not undertake any performance analysis of their approach.

Drozda et al. [11] used a forward feature selection process together with a wrapper approach [12] to identify a suitable set of features for misbehavior detection. A co-stimulation inspired architecture with the aim to decrease the false positives rate while stimulating energy efficiency is proposed and evaluated.

Even though the BIS seems to be a good inspiration for improving misbehavior detection in ad hoc and sensor networks, approaches based on machine learning and similar methods received much more attention; see [2] and the references therein. Despite recent efforts, *energy efficient* misbehavior detection however remains to be a challenging problem.

4 Protocols and Assumptions

We now review several protocols, mechanisms, assumptions and definitions relevant to our experiments.

An *ad hoc network* can be defined as a net $N = (n(t), e(t))$ where $n(t), e(t)$ are the set of nodes and edges at time t, respectively. Nodes correspond to wireless devices that wish to communicate with each other. An edge between two nodes A and B is said to exist when A is within the radio transmission range of B and vice versa. A *sensor network* is a static ad hoc network deployed with the goal to monitor environmental or physical conditions such as humidity, temperature, motion or noise.

AODV is [13] is an on-demand routing protocol that starts a route search only when a route to a destination is needed. This is done by flooding the network with RREQ (= Route Request) control packets. The destination node or an intermediate node that knows a route to the destination will reply with a RREP (= Route Reply) control packet. This RREP follows the route back to the source node and updates routing tables at each node that it traverses. A RERR (= Route Error) packet is sent to the connection originator when a node finds out that the next node on the forwarding path is not replying.

At the MAC (Medium access control) layer, the medium reservation is often contention based. In order to transmit a data packet, the IEEE 802.11 MAC protocol uses carrier sensing with an RTS-CTS-DATA-ACK handshake (RTS = Ready to send, CTS = Clear to send, ACK = Acknowledgment).

In *promiscuous mode*, a node listens to the on-going traffic among other nodes in the neighborhood and collects information from the overheard packets. Promiscuous mode is energy inefficient because it prevents the wireless interface from entering sleep mode, forcing it into either idle or receive mode. There is also extra overhead caused by analyzing all overheard packets. According to [14], power consumption in idle and receive modes is about 12-20 higher than in sleep mode. Promiscuous mode requires that omnidirectional antennas are used.

We do not assume any node location knowledge or time synchronization among nodes. We assume that packets are authenticated, i.e. the sender of any packet can be easily identified as well as can be changes in the packet body. This is a reasonable assumption in line with e.g. the ZigBee specification [15].

5 Misbehavior Modeling and Classification Performance Evaluation

Node misbehavior can be the result of an intrusion or failure. We considered three types of misbehavior: (i) DATA packet dropping: 30% DATA packets were randomly and uniformly dropped at misbehaving nodes. (ii) DATA packet delaying: 30% DATA packets were randomly and uniformly delayed by 0.1 second at misbehaving nodes. (iii) Wormholes [16]. Wormholes are private (out-of-band) links between one or several pairs of nodes. They are added by an attacker in order to attract data traffic into them to gain control over packet routing. This could lead to packet data load manipulation.

In order to simplify the experiments, we decided to merge these three types of misbehavior into a single class called "misbehavior". This means, when evaluating the classification performance of our approach, we will apply a "normal vs. misbehavior" classification scheme. The necessary traffic samples for these two classes were created through network simulation by applying one of the above misbehavior models or running a misbehavior free simulation. Natural data packet losses (noise) due to e.g. collisions at the MAC layer amounted to $0 - 15\%$. The detailed experimental setup is discussed in one of the following sections.

Classification performance in our experiments was evaluated in terms of detection rate and false positives (FP) rate. The two measures were computed as follows:

$$det.\ rate = \frac{c_{c_j}}{n_{c_j}} \times 100.0\% \qquad FP\ rate = \frac{FP_{c_j}}{n_{c_j}} \times 100.0\% \tag{1}$$

where $c_j = \{normal, misbehavior\}$. n_{c_j} is the number of vectors (samples) labeled with the class c_j; note that $n_{c_j} > 0$ in all our experiments. c_{c_j} is the number of vectors that were correctly classified by the induction algorithm as belonging to the class c_j. FP_{c_j} is the number of samples incorrectly predicted as belonging to c_j. 95% confidence intervals ($CI_{95\%}$) were computed for each measure. Classification performance with respect to a vector set was evaluated by means of the classification error:

$$class.\ error = \frac{\sum_{c_j} FP_{c_j}}{\sum_{c_j} n_{c_j}} \times 100.0\% \tag{2}$$

6 Co-stimulation in Ad Hoc and Sensor Networks

Drozda et al. [11] introduced and evaluated an architecture inspired by the interplay between the innate and adaptive immune system. 24 features suitable for misbehavior detection from several layers of the OSI protocol stack were considered. These features were divided into several subsets with respect to their energy requirements and protocol assumptions. A wrapper approach [12] was used to identify features with a *statistically significant contribution* to the classification process. Due to their relevance for our experimental setup, the features that were found significant are listed below.

6.1 The Features

Let $s_s, s_1, ..., s_i, s_{i+1}, s_{i+2}, ..., s_d$ be the path between s_s and s_d determined by a routing protocol, where s_s is the source node, s_d is the destination node. Features in the following feature set f are averaged over a time window. We use the feature labels (M3, M4, ...) as they were introduced in [11].

MAC Layer Features:

M3. **Forwarding index (watchdog):** Ratio of data packets sent from s_i to s_{i+1} and then subsequently forwarded to s_{i+2}.

M4. **Processing delay index (quantitative watchdog):** Time delay that a data packet accumulates at s_{i+1} before being forwarded to s_{i+2}.

Routing Layer Features:

R5. **Average distance to destination:** Average number of hops from s_i to any destination.

R9. **Connectivity index:** Number of destinations with known routes as recorded in the routing table of node s_i.

R12. **Diameter index:** Number of hops to the furthermost destination as recorded in the routing table of node s_i.

Transport Layer Features:

T1. **Out-of-order packet index:** Number of DATA packets that were received by s_i out of order. This assumes that each DATA packet has a unique ID computed by the connection source. Normalized by the time window size.

T2. **Interarrival packet delay index 1:** Average delay between DATA packets received by s_i. The delay was computed separately *for each connection* and then a master average was computed.

T3. **Interarrival packet delay variance index 1:** Variance of delay between DATA packets received by s_i. The variance was computed separately for *each connection* and then a master average was computed.

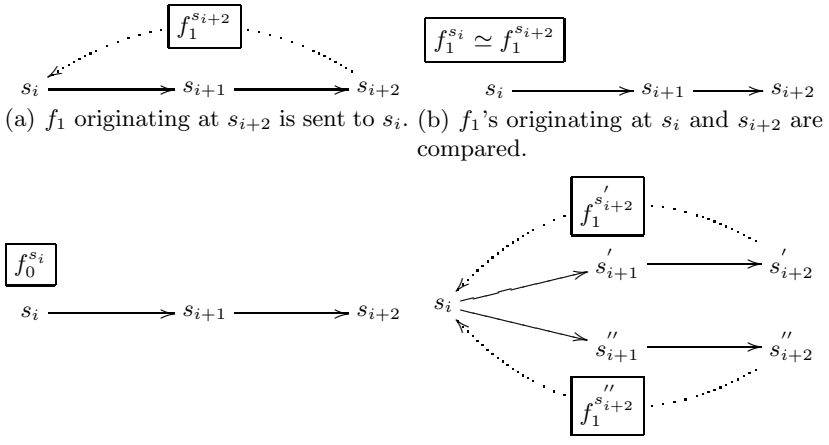
T4. **Interarrival packet delay index 2:** Average delay between DATA packets received by s_i.

T5. **Interarrival packet delay variance index 2:** Variance of delay between DATA packets received by s_i.

All the above features can be locally computed. The features T2, T3 and T4, T5 are identical, if only DATA packets belonging to a single connection are received by s_i. M3 and M4 require operation in promiscuous mode and therefore can be considered energy inefficient. Sudden changes in R5, R9 and R12 can help detect a topological change caused by a wormhole. This type of misbehavior increases the number of nodes lying within a fixed number of hops from s_i. We consider the following two subsets of the feature set f:

1. $f_0 = \{M3, M4\}$.
2. $f_1 = \{R5, R9, R12, T1, T2, T3, T4, T5\}$.

Besides energy efficiency, these two feature subsets have a very different expressive power. For example to show that s_{i+1} is misbehaving, either f_0 computed at s_i is necessary or f_1 computed at both s_i and s_{i+2} is necessary [11].



(a) f_1 originating at s_{i+2} is sent to s_i.

(b) f_1's originating at s_i and s_{i+2} are compared.

(c) *Co-stimulation:* if (b) indicates that s_{i+1} is misbehaving, f_0 will be computed and evaluated.

(d) If s_i forwards data packets for several connections, multiple f_1's are computed and sent.

Fig. 1. Immuno-inspired misbehavior detection approach

In other words, in case of f_1 measurements from several nodes are required, i.e. the output measured at s_{i+2} must be compared with the input at s_i in order to identify s_{i+1} as misbehaving. In the following, we use $f_0^{s_i}$, $f_1^{s_i}$ to denote f_0, f_1 computed by the node s_i, respectively.

6.2 A Co-stimulation Based Approach

To achieve a robust detection performance, a co-stimulation inspired mechanism is considered in [11]. This mechanism attempts to mimic the ability of communication between various players of the innate and adaptive immune system. More specifically, the inspiration was based upon the observation that a cell will not be marked as infectious and thus eliminated as long as no co-stimulation from other BIS players is received. Thus by analogy, a node should not mark another node as misbehaving as long as it does not receive a co-stimulatory signal from other parts of the network.

The co-stimulation inspired approach is depicted in Fig. 1. Each node in the network computes the feature set f_1. This feature set is then proliferated upstream (towards the connection source); see Fig. 1(a). Each f_1 travels exactly two hops; in our example from s_{i+2} to s_i. If s_i receives f_1 originating at one of its neighbors, it will forward f_1 in the upstream direction. If f_1 is not originating at one of its neighbors, it is used but not forwarded.

Since the computation of f_1 is time window based, the frequency with which f_1 gets sent depends on the time window size. Upon receiving f_1 from a two-hop neighbor, the node s_i combines and evaluates it with its own f_1 sample; see Fig. 1(b). Based on this, a behavior classification with respect to the node s_{i+1} is done. If s_i classifies s_{i+1} as misbehaving, then it computes a sample

based on the f_0 feature set. This means, a *co-stimulation* from the f_1 based classification approach is needed in order to activate the energy less efficient f_0 based classification approach; see Fig. 1(c). If misbehavior gets confirmed, the node s_{i+1} is marked as misbehaving. Note that s_i can receive f_1 samples from several two-hop neighbors; see Fig. 1(d) for an example.

The proliferation of f_1 can be implemented without adding any extra communication complexity by attaching this information to CTS or ACK MAC packets. As long as there are DATA packets being forwarded on this connection, the feature set can be propagated. If there is no DATA traffic on the connection (and thus no CTS/ACK packets exchanged), the relative necessity to detect the possibly misbehaving node s_{i+1} decreases. Optionally, proliferation of f_1 can be implemented by increasing the radio radius at s_{i+2}, by broadcasting it with a lower time-to-live value or by using a standalone packet type.

If the node s_{i+1} decides not to cooperate in forwarding the feature set information, the node s_i will switch, after a time-out, to the feature set f_0 computation. In this respect, not receiving a packet with the feature set f_1 can be interpreted as a form of negative co-stimulation. If the goal is to detect a misbehaving node, it is important that the originator of f_1 can be unambiguously identified, i.e. strict authentication is necessary. An additional requirement is the use of sequence numbers for feature sets f_1. Otherwise, the misbehaving node s_{i+1} could interfere with the mechanism by forwarding outdated cached feature sets f_1.

We introduce the following notation in order to keep track of composite feature sets f_1 computed at the nodes s_i and s_{i+2}: $\mathcal{F}_1^{s_i} = f_1^{s_i} \cup f_1^{s_{i+2}}$. For simplicity, we will omit the superscript.

The mechanisms of the innate immune system bear a certain resemblance to the f_0 based classification phase; see Fig. 1(c). The innate system is for example very efficient in signaling tissue injury or damage to the adaptive immune system. To a certain degree it relies on some very rudimentary methods such as recognizing an unusually high level of dead or damaged self cells (e.g. blood cells). This can be directly compared with the very straightforward functionality of watchdogs. Similarly, the more learning extensive classification approach based on $\mathcal{F}_1$ (Fig. 1(b)) can be compared with the adaptive immune system.

6.3 An Error Propagation Algorithm for Ad Hoc Networks

In [11] it was concluded that as the size of the time window decreases, the classification ability of the feature sets f_0 and $\mathcal{F}_1$ will equalize. That is:

$$\lim_{win.\ size \to 0} class.\ error(\mathcal{F}_1) \approx class.\ error(f_0) \tag{3}$$

This is a natural consequence of the fact that instead of observing a data packet's delivery in promiscuous mode by the node s_i, it can be equally well done in a cooperative way by s_i and s_{i+2}, if the window size (sampling frequency) is small. In other words, if the time window is small enough that it always includes only a single event, the relationship between events at s_i and s_{i+2} becomes explicit. This is however connected with a very high communication

cost (each packet arrival at s_{i+2} must be explicitely reported to s_i). The following observations can be formulated:

- If using f_0, the thresholds for the watchdog features can be set directly, for example M3 could be set to 0.90 (90% data packets must be correctly forwarded) in order to classify s_{i+1} as misbehavior free.
- If using $\mathcal{F}_1$ with $win.\ size \gg 0$, learning based on data traffic at both s_i and s_{i+2} must be done. Feature averaging over a time window increases the classification task complexity. Frequency of the extra communication between s_i and s_{i+2} depends on the time window size.

Considering the above two observations and Eq. 3, we propose the following algorithm.

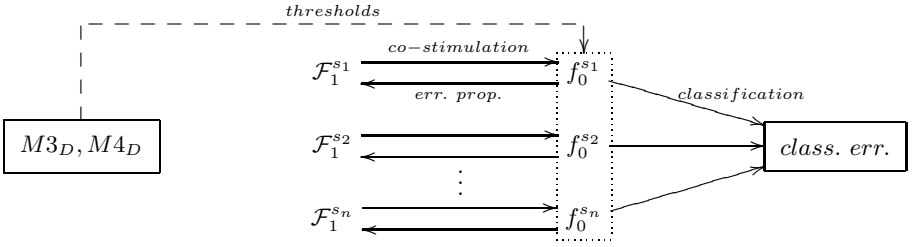
Error Propagation Algorithm

1. Choose desired levels (thresholds) for the forwarding and processing delay indexes denoted as $\mathrm{M3}_D$ and $\mathrm{M4}_D$, respectively.
2. The network starts operating. Allow all nodes to use promiscuous mode. Each node builds f_0 and $\mathcal{F}_1$ sample sets. Disable promiscuous mode at a node, when the sample set size reaches a target value.
3. Label the $\mathcal{F}_1$ based samples. A $\mathcal{F}_1$ based sample will be labeled as "normal", if in the f_0 based sample from the corresponding time window, $M3 \geq M3_D$ and $M4 \leq M4_D$, and as "misbehavior" otherwise.
4. Error propagation: compute a classifier by using only the samples based on $\mathcal{F}_1$.
5. Use the $\mathcal{F}_1$ based classifier computed in the above step to classify any fresh $\mathcal{F}_1$ samples. Co-stimulation: if a sample gets classified as "misbehavior", allow s_i to compute a f_0 sample in promiscuous node.
6. Classification: apply the $\mathrm{M3}_D$ and $\mathrm{M4}_D$ thresholds to the fresh f_0 based sample. If the "misbehavior" classification gets confirmed, mark s_{i+1} as misbehaving. Normal behavior can get classified in a similar way.

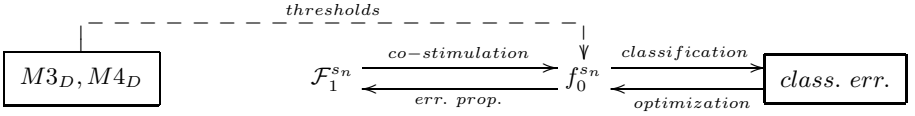
The algorithm is schematically depicted in Fig. 2(a). The error propagation step utilizes the rule described in Eq. 3. The classification error induced by the $M3_D$ and $M4_D$ threshold values gets this way propagated onto the $\mathcal{F}_1$ feature set. Since for practical reasons $win.\ size \gg 0$ must hold, a co-stimulation ($\mathcal{F}_1 \longrightarrow f_0$) is necessary for improving the classification performance. The less energy effcient f_0 based classification is thus used only if a co-stimulation from its $\mathcal{F}_1$ based counterpart is present. In other words, any detection based on $\mathcal{F}_1$ (adaptive immunity) must coincide with some damage being explicitely detected. This damage detection is based on f_0 (innate immunity).

The rule in Eq. 3 implicates that the final classification performance should equal the classification ability of the two watchdog features M3 and M4, if a small time window is applied. This implies that the values for $M3_D$ and $M4_D$ must be reasonably chosen in order to minimize the classification error.

The error propagation algorithm can be extended with an optimization phase; see Fig. 2(b). The classification outcome can be used to find new thresholds for



(a) An approach with error propagation and co-stimulation.



(b) An approach extended with optimization.

Fig. 2. Error propagation algorithm

the f_0 based samples. In this case, the threshold values $M3_D, M4_D$ serve as seed values for the whole process. Then it is possible to relabel the $\mathcal{F}_1$ samples and to recompute the related classifier. Co-stimulation and classification phases are then executed. In general, any suitable optimization procedure with several optimization and error propagation cycles could be applied. In order to use this extended approach in a distributed way, an estimation of natural packet losses (noise) at nodes must be done.

As shown in Fig. 1(d), any $\mathcal{F}_1$ computation is also based on the information available at s_{i+2}. This two-hop dependency can be compared to synapses among neurons. Under this comparison, our extended approach bears a certain similarity to the *backpropagation algorithm* for artificial neural networks [17].

7 Experimental Setup

The network simulation was done with the JiST/SWANS [18] network simulator.

Topology, Connections, Data Traffic and Protocols: We used a topology based on a snapshot from the movement prescribed by the Random waypoint movement model [19]. There were 1,718 nodes simulated; the physical area size was 3,000m $\times$ 3,000m. We used this topology because it has been quite extensively studied by Barrett et al. in [20]; results reported therein include many graph-theoretical measures that were helpful in finding suitable parameters for our experiments.

We modeled data traffic as Constant bit rate (CBR), i.e. there was a constant delay when injecting data packets. This constant delay in our experiments was 2 seconds (injection rate of 0.5 packet/s); the packet size was 68 bytes. CBR data packet sources correspond e.g. to sensors that transmit their measurements in predefined constant intervals. CBR can be considered due to its synchronized

nature an extreme model for data packet injection. In fact, the results published in [21] show that when using a stochastic injection model, such as the Poisson traffic model, one can expect a better performance of the detection system.

We used 50 concurrent connections. The connection length was 7 hops. In order to represent a dynamically changing system, we allowed connections to expire. An expired connection was replaced by another connection starting at a random source node that was not used previously. Each connection was scheduled to exist approximately 15 to 20 minutes. The exact connection duration was computed as

$$\delta + r_U \lambda \tag{4}$$

where δ the desired duration time of a connection, r_U a random number from the uniform distribution $[0, 1]$ and λ the desired variance of the connection duration. In our experiments, we used $\delta = 15\ min$ and $\lambda = 5\ min$.

We used the AODV routing protocol, IEEE 802.11b MAC protocol, UDP transport protocol and IPv4. The channel frequency was set to 2.4 GHz. The bandwidth was set to 2 Mbps. Antenna and signal propagation properties were set so that the resulting radio radius equals 100 meters.

Misbehavior: There were 20 wormholes in each simulation run; each wormhole was designed to bypass 15 hops, i.e. the source and sink were 15 hops away before a given wormhole was activated. There were 236 nodes randomly chosen to execute DATA dropping or delaying misbehavior. Our intention was to model *random failure occurrences*, assuming a uniform failure distribution in the network. As it is hard to predict the routing of packets, many of these nodes could not execute any misbehavior as there were no DATA packets to be forwarded by them. In our case, about 20-30 misbehaving nodes were concurrently active.

Experiments: We did 20 independent runs for each misbehavior type and 20 misbehavior free (normal) runs. The simulation time for each run was 4 hours. We used a non-overlapping time window approach for features' computation. We used four different time window sizes: 50, 100, 250 and 500 seconds. In case of a 500-second time window, there were 28 non-overlapping windows in each run (4 hours/500 seconds = 28.8). This gave us $5 \times 28 \times 20 = 2{,}800$ vectors (samples) for each node. This also determined the max. target sample size for Step 2 of our algorithm.

$M3_D$ and $M4_D$ threshold values: We set $\mathrm{M3}_D$ to 97.5% and $\mathrm{M4}_D$ to 4ms. These values were found through the extended approach shown in Fig. 2(b) using the "trial and error" method (as a substitute for a formal optimization approach).

Induction algorithm: We used a decision tree classifier. To decide whether a node within the decision tree should be further split (impurity measure), we used the information gain measure [17]. As the decision tree classifier is a well-known algorithm, we omit its discussion. We refer the interested reader to [17]. We used the decision tree implementation from the Rapidminer tool [22].

The estimation of classification performance was done using a stratified k-fold cross-validation approach [17] with $k = 20$.

Table 1. Performance. $M3_D = 0.975$, $M4_D = 4$ms.

	Normal				Misbehavior			
Win. size[s]	Det. rate	$CI_{95\%}$	FP rate	$CI_{95\%}$	Det. rate	$CI_{95\%}$	FP rate	$CI_{95\%}$
	f_0 ($M3_D$ and $M4_D$)							
50	96.15	1.96	0.46	0.02	99.25	0.05	6.36	7.01
100	96.25	2.01	0.20	0.01	99.68	0.03	6.16	7.10
250	96.17	2.11	0.15	0.01	99.76	0.03	6.28	7.33
500	95.84	2.17	0.12	0.01	99.81	0.04	6.71	7.20
	$\mathcal{F}_1$							
50	93.56	2.37	2.09	0.20	96.54	0.61	10.59	9.08
100	92.38	2.64	2.81	0.49	95.27	1.71	12.45	9.68
250	89.05	2.91	5.04	0.90	91.55	2.29	17.97	11.78
500	86.10	3.93	7.49	1.60	87.69	1.78	22.36	11.94
	$\mathcal{F}_1$ and then f_0 ($M3_D$ and $M4_D$)							
50	93.23	2.40	0.12	0.01	95.48	0.89	5.41	5.67
100	92.24	2.66	0.05	0.00	94.24	2.44	5.24	6.02
250	88.97	3.00	0.06	0.00	90.35	3.00	5.23	6.10
500	86.04	3.95	0.06	0.01	85.51	2.39	4.84	5.31

8 Performance Evaluation

The results achieved by applying the error propagation algorithm are reported in Table 1. It can be seen that as the time window size decreases, the performance of f_0 and $\mathcal{F}_1$ becomes more and more comparable. It can be also seen that the final co-stimulation with f_0 improves the performance only in a limited way compared to the f_0 only approach. More specifically, the FP rate for "normal" behavior could be significantly decreased. The confidence intervals ($CI_{95\%}$) belonging to the FP rates for "normal" behavior are almost zero. The FP and $CI_{95\%}$ rates for "misbehavior" could not be significantly decreased compared to f_0. This performance gap between "normal" and "misbehavior" originates from the fact that learning "normal" is simpler than learning "misbehavior". In the latter case, samples which belong to "normal" are often misinterpreted as misbehaving. This is a result of the unsophisticated initial sample separation (labeling) based only on the two watchdog features. This increases the FP rate and the $CI_{95\%}$ of the misbehavior class.

The average sample size per node was 6,941, if the window size was set to 50 seconds. As not every node was continuously involved in data packet forwarding, the sample size was lower than the theoretical max. size.

Notice that according to our approach, when detecting a misbehavior, the more costly f_0 based detection will only get used, if (i) a true positive was detected by $\mathcal{F}_1$ or (ii) a false positive was mistakenly detected by $\mathcal{F}_1$. This means, for a misbehavior free ad hoc network, the energy saving over an exclusive f_0 approach is proportional to $1 - FP\ rate$, where $FP\ rate$ is in this case the $\mathcal{F}_1$ based false positives rate for the misbehavior class.

9 Conclusions

We proposed and tested an approach inspired by the role of co-stimulation in the BIS. Our preliminary results show that that this approach has a positive effect on both energy efficiency of misbehavior detection and the false positives rate. The achieved detection rate was in the $86 - 95\%$ range depending on the monitoring window size. Our error propagation algorithm can be used for both misbehavior detection as well as for checking whether an ad hoc network complies with its operational strategy. We pointed out that our approach can be extended with an optimization technique that would allow for a classification error minimization.

Acknowledgments

This work was supported by the German Research Foundation (DFG) under the grant no. SZ 51/24-2 (Survivable Ad Hoc Networks – SANE).

References

1. Yegneswaran, V., Barford, P., Ullrich, J.: Internet intrusions: global characteristics and prevalence. In: Proc. of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, pp. 138–147 (2003)
2. Anantvalee, T., Wu, J.: A survey on intrusion detection in mobile ad hoc networks. Wireless/Mobile Network Security, 159–180 (2007)
3. Murphy, K., Travers, P., Walport, M.: Janeway's immunobiology. Garland Pub. (2008)
4. Aickelin, U., Bentley, P., Cayzer, S., Kim, J., McLeod, J.: Danger Theory: The Link between AIS and IDS? In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 147–155. Springer, Heidelberg (2003)
5. Hofmeyr, S., Forrest, S.: Immunity by design: An artificial immune system. Proc. of Genetic and Evolutionary Computation Conference (GECCO) 2, 1289–1296 (1999)
6. Sarafijanovic, S., Le Boudec, J.: An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal and memory detectors. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 342–356. Springer, Heidelberg (2004)
7. Drozda, M., Schaust, S., Szczerbicka, H.: AIS for Misbehavior Detection in Wireless Sensor Networks: Performance and Design Principles. In: Proc. IEEE Congress on Evolutionary Computation (CEC), pp. 3719–3726 (2007)
8. D'haeseleer, P., Forrest, S., Helman, P.: An Immunological Approach to Change Detection: Algorithms, Analysis and Implications. In: IEEE Symposium on Security and Privacy, pp. 110–119 (1996)
9. Timmis, J., Hone, A., Stibor, T., Clark, E.: Theoretical advances in artificial immune systems. Theor. Comput. Sci. 403(1), 11–32 (2008)
10. Kim, J., Bentley, P., Wallenta, C., Ahmed, M., Hailes, S.: Danger Is Ubiquitous: Detecting Malicious Activities in Sensor Networks Using the Dendritic Cell Algorithm. In: Bersini, H., Carneiro, J. (eds.) ICARIS 2006. LNCS, vol. 4163, pp. 390–403. Springer, Heidelberg (2006)

11. Drozda, M., Schildt, S., Schaust, S.: An Immuno-Inspired Approach to Fault and Misbehavior Detection in Ad Hoc Wireless Networks. Technical report, Leibniz University of Hannover (2009)
12. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
13. Perkins, C.E., Royer., E.M.: Ad hoc On-Demand Distance Vector Routing. In: Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pp. 90–100 (1999)
14. Feeney, L., Nilsson, M.: Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In: Proc. of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM, vol. 3 (2001)
15. ZigBee Alliance: ZigBee Specification (2005)
16. Hu, Y., Perrig, A., Johnson, D.: Wormhole attacks in wireless networks. IEEE Journal on Selected Areas in Communications 24(2), 370–380 (2006)
17. Alpaydin, E.: Introduction To Machine Learning. MIT Press, Cambridge (2004)
18. Barr, R., Haas, Z., van Renesse, R.: JiST: an efficient approach to simulation using virtual machines. Software Practice and Experience 35, 539–576 (2005)
19. Johnson, D., Maltz, D.: Dynamic source routing in ad hoc wireless networks. Mobile Computing 353, 153–181 (1996)
20. Barrett, C., Drozda, M., Engelhart, D., Kumar, V., Marathe, M., Morin, M., Ravi, S., Smith, J.: Understanding protocol performance and robustness of ad hoc networks through structural analysis. In: Proc. of the IEEE International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob 2005), vol. 3, pp. 65–72 (2005)
21. Schaust, S., Drozda, M.: Influence of Network Payload and Traffic Models on the Detection Performance of AIS. In: Proc. of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2008), pp. 44–51 (2008)
22. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T.: Yale: Rapid prototyping for complex data mining tasks. In: Proc. of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 935–940 (2006)

Grammar-Based Immune Programming for Symbolic Regression

Heder S. Bernardino and Helio J.C. Barbosa

Laboratório Nacional de Computação Científica,
Av. Getulio Vargas, 333, 25.651-075, Petrópolis-RJ, Brazil
hedersb@gmail.com, hcbm@lncc.br

Abstract. This paper presents a Grammar-based Immune Programming (GIP) that can evolve programs in an arbitrary language using a clonal selection algorithm. A context-free grammar that defines this language is used to decode candidate programs (antibodies) to a valid representation. The programs are represented by tree data structures as the majority of the program evolution algorithms do. The GIP is applied to symbolic regression problems and the results found show that it is competitive when compared with other algorithms from the literature.

Keywords: Artificial immune system, grammatical evolution, immune programming, symbolic regression.

1 Introduction

For the automatic generation of programs, the genetic programming (GP) paradigm has become very popular. The first results of the GP technique were reported in the early eighties by Smith [23,24]. The method was also developed by other authors and greatly expanded by Koza in [12] (see the survey in [21]).

In GP a candidate solution is often represented as a tree structure [5,12] and can, in principle, be decoded into a computer program, a numerical function in symbolic form, or a candidate design, such as an analog circuit. An objective function which translates a conveniently defined performance index of the candidate program, function, or design is then maximized or minimized.

However, relatively few applications of this type can be found using artificial immune systems (AISs). Johnson [11] proposed AIS programming inspired in the GP paradigm and based on the clonal selection principle. That method uses a tree-based representation [5] and was applied to symbolic regression problems. In 2006, Musilek et. al. [16] proposed the Immune Programming (IP) technique, which employs a stack-based framework, and was also applied to symbolic regression problems. The IP was extended and applied to evolve a model of disinfection of Cryptosporidium parvum in [14] where the results found were better than other available in the literature. Cicazzo et.al. [4] proposed an "elitist Immune Programming", or simply eIP. That method is inspired by clonal selection and uses the GP theory to generate the candidate solutions. The elitism is due to the fact that the best individual of the population and its hypermuted clone are

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 274–287, 2009.

always kept. The proposed technique was applied to circuit design optimization where it outperformed both the standard IP [16] and Koza's GP [13] in that test problem. A Clonal Selection Programming (CSP) algorithm which uses a symbolic string of fixed length was proposed in [9,8]. Symbolic regression [8] problems were used to evaluate and compare the algorithm with IP and gene expression programming (GEP) [7] as well fault detection system [9].

Another approach to evolve programs is Grammatical Evolution (GE) [22]. This method uses a Backus Naur Form (BNF) grammar to define the syntax of the language and a variable length string to represent the candidate solutions. This work was extended in [19] where it was shown to outperform GP when applied to symbolic integration and Santa Fe Ant Trail problems.

GE evolves valid programs from a user-specified grammar and can employ different meta-heuristics as the search engine, such as: genetic algorithms (GA) [19], particle swarm optimization (PSO) [18], differential evolution (DE) [17], and artificial immune systems (AISs) [15].

In the AISs field, Amarteifio and O'Neill [1] proposed a coevolutionary strategy where a simple GA evolved with the help of an immune inspired method aiming at promoting diversity. The approach was applied to symbolic regression and multiplexer problems. Another immune approach using GE can be found in [15] where a grammatical immunoglobulin hypermutation was proposed in which a nonterminal is mutated to another defined by the specified grammar. The method was used to infer a kinetic model for the oxidative metabolism of 17β-estradiol (E_2).

The present paper proposes a new Grammar-based Immune Programming (GIP) where candidate solutions are represented as in [19] but a new decoding process is performed. The decoding procedure creates a tree, based on the grammar definition, and repairs it when necessary, always generating valid programs.

The remaining of this paper is structured as follows. Before describing the repair method in Section 5, AIS concepts are introduced in Section 2, Backus-Naur Form and Formal Grammar are defined in Section 3, and the description of GE in Section 4. The GIP algorithm is describle in Section 6. Symbolic regression experiments, results, and comparisons can be found in Section 7. Finally, Section 8 concludes the paper.

2 Artificial Immune System

The natural immunity comprises innate and adaptive immunities [2]. The former is composed by cells and mechanisms that defend the host from attacks by other organisms in a non-specific manner while the later comprises highly specialized cells and processes that defend the organism from antigens. The main adaptive immunity feature is to distinguish between proteins produced by cells of the body (self) and the ones produced by intruders or by cells under virus control (non-self). According to the clonal selection theory, there is a selection mechanism which leads to the evolution of the immune system repertoire during the lifetime of the individual [3]. By this theory, on binding with a suitable antigen, activation of lymphocytes occurs. Once activated, clones of the lymphocyte are

produced expressing receptors identical to the original lymphocyte that encountered the antigen (clonal expansion). Any lymphocyte that has receptors specific to molecules of the host organism must be deleted (these lymphocytes will not be cloned). This ensures that only an antigen may cause a clonal expansion. In this way, the immune system can be viewed as a classifier of cells into either self or non-self cells. The non-self cells are called antigens, and are assumed as being from a pathogen and thus need to be removed from the body. The increase in the average of the affinity between the antibodies and antigens due to the somatic hypermutation and selection mechanisms of clonal expansion is known as affinity maturation.

Artificial Immune Systems are composed by intelligent methodologies, inspired by the biological immune system, to solve real world problems. The clonal selection algorithm (CLONALG) [6] is the immune method used by the approach proposed here. The algorithm evolves the antibodies inspired by the concept of clonal selection where each individual is cloned, hypermutated, and those with higher affinity are selected. The mutation rate is, normally, inversely proportional to the affinity of the antibody with respect to the antigens. Also, in each generation new candidate solutions can be randomly generated to improve the capacity of exploration of the algorithm.

3 Backus-Naur Form and Formal Grammars

John Backus and Peter Naur developed a context-free grammar to define the syntax of a programming language. The Backus-Naur Form (BNF) is a metasyntax widely used as a formal way to describe grammars of computer programming languages, instruction sets, and communication protocols, as well as for representing parts of natural language grammars.

A BNF specification is a set of derivation rules, written as

$$< nonterminal > ::= expression, \tag{1}$$

where $< nonterminal >$s are the production rules and *expression* consists of one or more sequences of symbols. If *expression* is composed by more than one sequence, the choices are delimited with the symbol "|". A nonterminal can be expanded into one or more terminals and nonterminals likewise may self reference to specify recursion.

A grammar G is a finite nonempty set of rules which specify the syntax of the language and is defined by

$$G = \{V, \Sigma, R, S\}, \tag{2}$$

where V is a finite set of nonterminals (or variables), Σ is a finite set of terminals or token symbols which are items that can appear in the language (such as x, $+$, $-$, and sin), R is a finite set of rules (or productions) in BNF, and S is the start symbol such that $S \in V$.

A context-free grammar (in which the left-hand side of each production rule consists of a single nonterminal symbol) to generate simple mathematical expressions is given by (2) where

$$V = \{< expr >, < op >, < pre >\}$$
$$\Sigma = \{sin, cos, log, exp, x, 1, +, -, *, /, (,)\}$$
$$S =< expr >$$

and R is defined as

$$\begin{aligned}
< expr > ::= \;& < expr >< op >< expr > && (0)\\
|\; & < pre > (< expr >) && (1)\\
|\; & < var > && (2)\\
< op > ::= \;& + && (0)\\
|\; & - && (1)\\
|\; & * && (2)\\
|\; & \backslash && (3)\\
< pre > ::= \;& sin && (0)\\
|\; & cos && (1)\\
|\; & log && (2)\\
|\; & exp && (3)\\
< var > ::= \;& x && (0)\\
|\; & 1 && (1)
\end{aligned}$$

4 Grammatical Evolution

Grammatical evolution (GE) is an evolutionary algorithm to evolve programs from a language via a defined grammar. The method was proposed by Ryan et.al. in [22] and extended in [19] by O'Neill and Ryan. In that approach the genotype is mapped onto terminals using the defined grammar. The genotype is a binary array representation of an integer one in which each value is generated by converting each 8-bit binary number. The integer value is used to select an appropriate rule from the grammar by the following expression

$$rule = (int) mod (nr)$$

where int is an integer number and nr is the number of rules for the current nonterminal.

A genotype and its integer array representation can be found in Figure 1.

The genotype-to-phenotype mapping process is shown in Table 1. For this example we consider the grammar defined in Section 3, in which the start nonterminal is $< expr >$, and consider the genotype shown in the Figure 1. In Table 1, the result of a given step is the input of the next one.

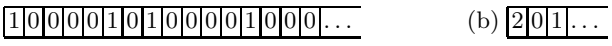


Fig. 1. (a) an example of genotype. (b) an example of 2 bits integer array generated by genotype in (a).

Table 1. Example of genotype-to-phenotype process

Step	Input	Rule	Result
1	$< expr >$	$(2)\ mod\ (3) = 2$	$< var >$
2	$< var >$	$(0)\ mod\ (2) = 0$	x

Evidently it is not necessary to use all integer values from the array and a genetic code degeneracy will occur.

Moreover, it is easy to see that during the genotype-to-phenotype mapping process it is possible for some candidate solutions to run out of integers in the array. In this case, the strategy of the original approach is to wrap the antibody and reuse (two or more times) the integer values. This wrapping process can be seen as inspired by the gene-overlapping phenomenon. However, that is not a good solution since it is possible that the genotype-to-phenotype mapping process enters an infinite loop as shown in Figure 2.

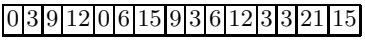


Fig. 2. Example of an integer array that causes an infinite loop in the genotype-to-phenotype mapping process

As seen, for all integers in the array from Figure 2, $(number)\ mod\ (3) = 0$, where *number* belongs to the array from Figure 2. Thus, the GE will generate $< expr >< op >< expr >$, $< expr >< op >< expr >< op >< expr >$, ..., indefinitely. When this happens, it is suggested [19] that the fitness (or affinity, in the AIS case) of this solution be set to the worst possible value. Therefore, the faulting solution is eliminated by the evolutionary process. To reduce the number of invalid individuals the same reference uses a steady-state replacement mechanism. However, it maintains such invalid individuals in the population. The approach proposed here repairs such invalid solutions into valid ones.

5 The Repair Method

In order to generate only valid candidate solutions, the approach proposed here produces a tree in the genotype-to-phenotype mapping process and repairs it when the array of integer numbers is exhausted and the program is not complete.

To do it, a new grammar is defined by a 5-tuple

$$G = \{V, \Sigma, R, R_f, S\}, \tag{3}$$

where V, Σ, R, and S are defined such as for equation (2). R_f is a set of rules, in BNF, to repair the generated tree. This new set of rules must contain a backward step to create the tree when there are no more integers available. Two restrictions must be highlighted: the new rule will use less subtrees than the original rule (reduced number of nonterminals), and given $w \in R_f$ then the inverse transformation of w is in (or can be reached by) R. Each R_f rule is similar to the equation (1). However, there are two or more nonterminals on the left side of the expression.

An example of R_f, applicable to the grammar given by the equation (2), can be found following

$$\begin{aligned} < expr >< op >< expr > ::= & \quad < expr > \quad | \quad < var > \\ < pre > (< exp >) ::= & \quad < var > \end{aligned}$$

Nonetheless, instead of using a rule as it is done when using R, a rule is chosen by the number and the type of available subtrees. Moreover, the first integer in the sequence is used to define the tree value and the other ones are employed to create the subtrees (if there is one).

The Figure 2 is an example of an individual which causes an infinite loop because, for all integer numbers in array, we have $(number)\ mod\ (3) = 0$, where $number$ belong to array. Thus, the GE will generate $< expr >< op >< expr >$, $< expr >< op >< expr >< op >< expr >$, ..., indefinitely, creating a tree only with $< expr >< op >< expr >$ nodes (first tree in Figure 3). Following the proposed procedure, the final leaf is replaced by a nonterminal $< var >$ which enters in a backward process up to the root node. Following R, and the next step being normal, then the phenotype is x. The Figure 3 demonstrates the repair method applied to the final tree from the standard make tree process from GE.

6 Grammar-Based Immune Programming

The proposed algorithm uses a binary string to represent each candidate solution and CLONALG as the search engine. A pseudo-code for the immune inspired algorithm can be found in Algorithm 1 where *antibodies* is a population of candidate solutions, *nGenerations* is the number of generations that the algorithm will perform (calculated by the total number of objective function evaluations and the number of evaluations of this function in each generation), β is used to define the number of clones each antibody will generate [6], *nSelection* is the number of antibodies selected to be cloned, *nReplaces* is the number of lower affinity antibodies which will be replaced by the new ones, and $bestAffinity$ is the best value found by the algorithm. Also, the function "calcAffinities" calculates the affinities between each antibody and all antigens, "select" selects the *nSelection* best individuals to be cloned, "clone" clones the selected antibodies; hypermutate, applies the somatic hypermutation in generated clones (such as defined in [6]), "update" replaces the *nReplaces* worst antibodies by other ones from *clones* and updates the $affinities$, and "getBestAffinity", returns the best affinity found by the algorithm.

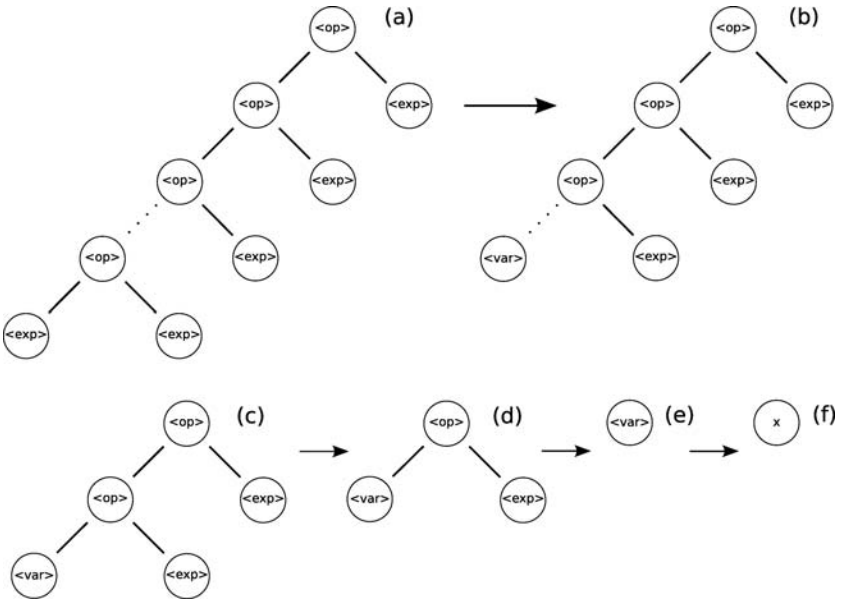


Fig. 3. The repair method

The method "calcAffinities" must be specially considered. This function is where each antibody is transformed from a binary string into a tree (i.e. a symbolic representation) and evaluated with respect to the data points in order to define its affinity level. In contrast to nature, the objective here is to minimize affinity. The process to decode an antibody into a tree is defined in Section 4 and 5.

7 Experiments

To illustrate that the proposed algorithm can find good solutions (programs) it is applied to three symbolic regression problems. This kind of problem aims at finding a mathematical expression that explains a given set of observations. A clonal selection algorithm (CLONALG) with binary encoding is used as the search engine. The grammar defined by equation (2) is applied together with the R_f presented in Section 5. The exclusion of some nonterminals and/or terminals from the grammar, in some experiments, was required in order to adjust our tests to the ones from the literature. To summarize the parameters used in each experiment, we adopt a style similar to that in [12]. The Table 2 shows the setup with the parameters used by all experiments and specific ones are presented in each subsection. The parameter are chosen from initial experiments where these values look make good results at most of the applied problems. A sensitivity analysis over the parameters is included in future works.

Algorithm 1. A CLONALG pseudo-code [2]

```
Data. antibodies, nGenerations, β, nSelection, nReplaces
Result. bestAffinity, antibodies
1  begin
2      affinities ⟵ calcAffinities(antibodies);
3      for generation = 0; generation < (nGenerations − 1) do
4          selectedAntibodies ⟵ select(antibodies, affinities, nSelection);
5          clones ⟵ clone(selectedAntibodies, affinities, β);
6          clones ⟵ hypermutate(clones, affinities);
7          cloneAffinities ⟵ calcAffinities(clones);
8          update(antibodies, affinities, clones, cloneAffinities, nReplaces);
9      bestAffinity ⟵ getBestAffinity(antibodies);
10 end
```

Table 2. GIP setup used in all experiments

Objective: Find a function in symbolic form that matches a given sample of data points.
Nonterminal unary symbols: *sin*, *cos*, *exp*, and *log*.
Nonterminal binary symbols: $+$, $-$, $*$, and $\backslash$.
Parameters: Population size=40, size of integer array=30, number of clones=1, $\rho = 3.5$ (used in hypermutation [19]), and percentage of new randomly generated antibodies=10%

7.1 Experiment 1

The first test is to apply the proposed algorithm to a simple regression problem and compare the results found with the ones obtained in [11] using an AIS as well as the standard GP. The objective is to find the function that matches a set of points generated by the function $x^4 + x^3 + x^2 + x$. The parameters in this experiment are given in Table 3. It is important to notice that the number of function evaluations in our tests is equal to that used by the reference.

The Figure 4 presents the comparison between GIP, AIS, and GP, when the best fitness in each generation is considered. The graphic shows that GIP found the best results for this problem, followed by AIS.

Considering the average of the fitness achieved in each generation, the Figure 5 shows the results obtained by the algorithms GIP, AIS, and GP. Similarly, the GIP found the best results for this case.

The same problem is used to compare the proposed algorithm with GE [19]. The table with the setup used can be found in Table 3.

The relation between the cumulative frequency (number of runs which match the solution) and the number of objective function evaluations was used in the comparisons. The Figure 6 presents the results found by the proposed algorithm, GE, and GP, where GP outperforms the other algorithms. For this problem, GIP produces the worst results showing, together the first comparison, that

Table 3. GIP setups used in experiment 1 – First and second comparisons

Terminal symbols: x.
Fitness cases: 100 data points, equally spaced, in the interval $[-1; 1]$.
Raw Fitness: 250 minus the sum of the error (maximization).
Standardized Fitness: Same as raw fitness.
Parameters: maximum number of objective function evaluations=80000 and number of independent runs=100.
Terminal symbols: x and 1.
Fitness cases: 20 data points, equally spaced, in the interval $[-1; 1]$.
Raw Fitness: Sum, taken over all fitness cases, of the error.
Standardized Fitness: Same as raw fitness.
Parameters: maximum number of objective function evaluations=25000 and number of independent runs=100.

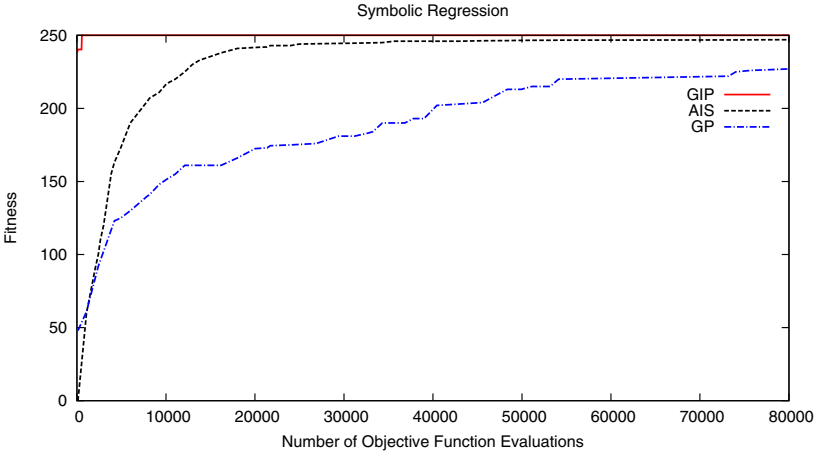


Fig. 4. Experiment 1 – Comparative fitness curves for GIP, AIS, and GP when the best fitness in each generation is considered. The AIS and GP curves were obtained from [11].

the proposed algorithm produced results close to the solution but not exactly matching it in most cases.

7.2 Experiment 2

The next experiment is a symbolic integration problem corresponding to finding a function f that is the integral of $g(x) = cos(x) + 2x + 1$ i.e. $f'(x) = g(x)$ [19]. Thus, $g(x)$ is numerically integrated by Simpson's rule in each interval (a, b) and this value is compared with the exact value $f(b) - f(a)$. The affinity of the candidate solution $f(x)$ is the sum of those errors. The setup for this problem can be found in Table 4.

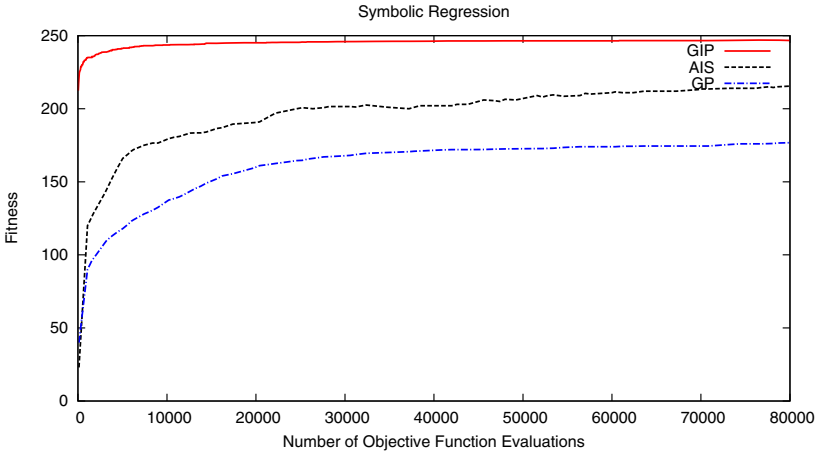


Fig. 5. Experiment 1 – Comparative fitness curves for GIP, AIS, and GP when the average of the fitness in each generation is considered. The AIS and GP curves were obtained from [11].

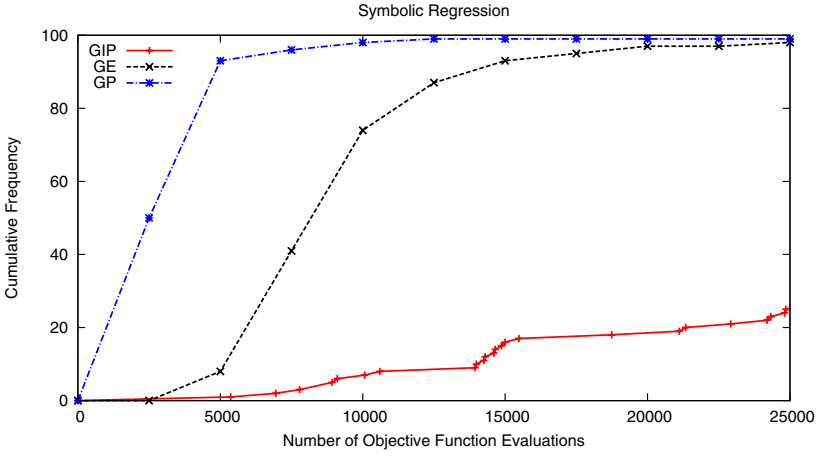


Fig. 6. Experiment 1 – Comparative cumulative frequency found by GIP, GE [19], and GP [19] algorithms

A comparison between GIP, GE, and GP can be seen in Figure 7 where GIP outperforms the other algorithms and converges to the integral of the given function, in all runs, with approximately 22000 objective function evaluations.

7.3 Experiment 3

The final test is a symbolic regression problem which can be found in [10]. In this experiment, the objective is to match the function $cos(2x)$ when the nonterminal

Table 4. GIP setup used in experiment 2

Objective: Find the analytical integral of a given function.
Terminal symbols: x.
Fitness cases: 20 data points in the interval $[0; 2\pi]$.
Raw Fitness: The sum of the differences between the produced function and the numerical integration over each sub-domain in the given interval.
Standardized Fitness: Same as raw fitness.
Parameters: maximum number of objective function evaluations=25000 and number of independent runs=100.

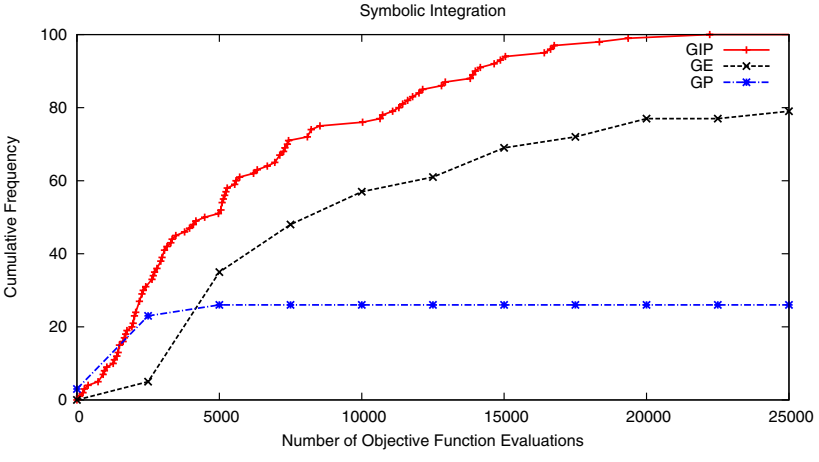


Fig. 7. Experiment 2 – Comparative cumulative frequency found by GIP, GE [19], and GP [19] algorithms

Table 5. GIP setup used in experiment 3

Terminal symbols: x and 1.0.
Nonterminal unary symbols: *sin*.
Fitness cases: 20 data points in the interval $[0; 2\pi]$.
Raw Fitness: The sum, taken over all fitness cases, of the error.
Standardized Fitness: Same as raw fitness.
Parameters: maximum number of objective function evaluations=100000 and number of independent runs=50

cos is not available. The GIP is compared with standard GP, grammar guided genetic programming (GGGP), and tree-adjunct grammar guided genetic programming (TAG3P). All the results from the literature are taken from [10]. The Table 5 presents the GIP setup for this test problem.

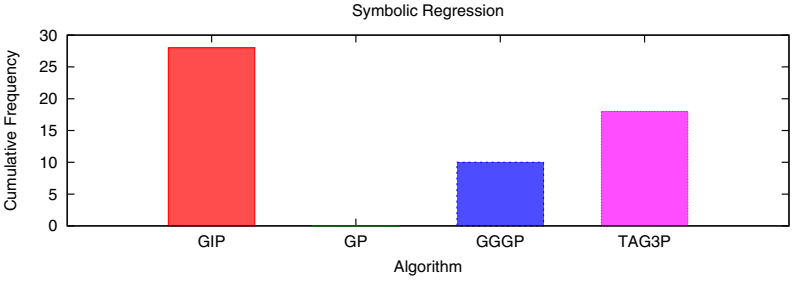


Fig. 8. Experiment 3 – Comparative cumulative frequency found by GIP, GP [10], GGGP [10], and TAG3P [10] algorithms

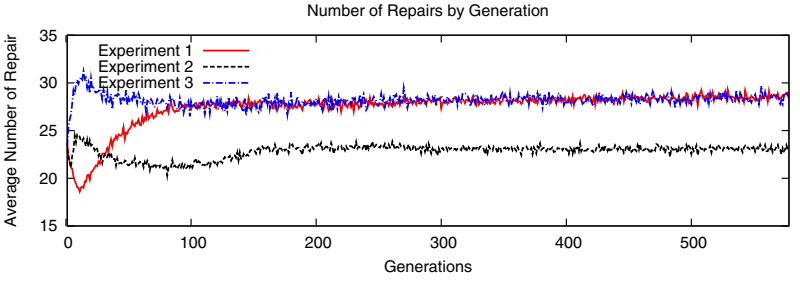


Fig. 9. The average number of repairs by generation for each experiment. For the experiment 3 the number of generations presented is equal to the other ones.

The Figure 8 presents a bar chart with the results found by GP and the algorithms tested in [10] showing that GIP outperforms the other algorithms in this test problem.

It is important to notice that the worst solution found by the GIP algorithm is a very good approximation of the target function:

$$\begin{aligned} sin\{2x + sin(1) + sin[sin(1)]\} &\simeq sin(2x + 1.5870951264734545) \\ &\simeq sin(2x + pi/2) = cos(2x) \end{aligned}$$

7.4 On the Use of the Repair Technique

Although O'Neil and Ryan [20] concluded that the use of the wrapping method is not an important issue because invalid individuals are removed from the population early in the run, we believe that the GAs' selection pression moves the population to the feasible space leading quickly to the production of valid solutions only. By using CLONALG as the search engine it is expected that this kind of convergence will not happen, and indeed, such trend can be observed in Figure 9 where the average number of repairs in each generation is presented for the three experiments performed here.

The Figure 9 shows that, for the three experiments presented here, the proposed algorithm tends to find better solutions in problems where the repair method is highly used in the first generations. However, the authors believe that more experiments need to be performed in order to better evaluate the repair process.

8 Conclusions

We proposed a Grammar-based Immune Programming (GIP) technique to evolve programs using a given grammar. Although based on grammatical evolution ideas, this AIS algorithm generates a tree in the genotype-to-phenotype mapping process and repairs it when the array of integers is exhausted and the program is incomplete (non-terminal symbols are still present).

The method was evaluated in three symbolic regression problems against several algorithms found in the literature (GP, AIS, GE, GGGP, and TAG3P) and the results obtained indicate that it is quite promising. The GIP outperforms the other methods in all presented experiments. An exception was the experiment 1 where the GP found better results when the cumulative frequency is considered. Of course, more experiments are necessary to demonstrate its efficiency and also study its sensitivity with respect to its parameters.

The authors believe that the application of AIS ideas to more complex search spaces is an important area of research, where more work is needed.

Acknowledgments. The authors thank the support from CNPq (140551/2008-5 and 311651/2006-2) and FAPERJ (E-26/ 102.825/2008) as well as the corrections and suggestions from the reviewers.

References

1. Amarteifio, S., O'Neill, M.: Coevolving antibodies with a rich representation of grammatical evolution. 1, pp. 904–911 (2005)
2. Bernardino, H.S., Barbosa, H.J.C.: Artificial Immune Systems for Optimization. In: Nature-Inspired Algorithms for Optimisation, pp. 389–411. Springer, Heidelberg (2009)
3. Burnet, F.M.: The Clonal Selection Theory of Acquired Immunity. Cambridge University Press, Cambridge (1959)
4. Ciccazzo, A., Conca, P., Nicosia, G., Stracquadanio, G.: An advanced clonal selection algorithm with ad-hoc network-based hypermutation operators for synthesis of topology and sizing of analog electrical circuits. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 60–70. Springer, Heidelberg (2008)
5. Cramer, N.L.: A representation for the adaptive generation of simple sequential programs. In: Proceedings of the 1st International Conference on Genetic Algorithms, Hillsdale, NJ, USA, pp. 183–187. L. Erlbaum Associates Inc., Mahwah (1985)
6. de Castro, L.N., Zuben, F.J.V.: Learning and optimization using the clonal selection principle. IEEE Trans. Evo. Comp. 6(3), 239–251 (2002)

7. Ferreira, C.: Gene expression programming: a new adaptive algorithm for solving problems. ArXiv Computer Science e-prints (February 2001)
8. Gan, Z., Chow, T.W., Chau, W.: Clone selection programming and its application to symbolic regression. Expert Sys. Appl. 36(2), 3996–4005 (2009)
9. Gan, Z., Zhao, M.-B., Chow, T.W.: Induction machine fault detection using clone selection programming. Expert Systems with Appl. 36(4), 8000–8012 (2009)
10. Hoai, N., McKay, R., Essam, D., Chau, R.: Solving the symbolic regression problem with tree-adjunct grammar guided genetic programming: the comparative results, vol. 2, pp. 1326–1331 (2002)
11. Johnson, C.G.: Artificial immune system programming for symbolic regression. In: Ryan, C., Soule, T., Keijzer, M., Tsang, E.P.K., Poli, R., Costa, E. (eds.) EuroGP 2003. LNCS, vol. 2610, pp. 345–353. Springer, Heidelberg (2003)
12. Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems). MIT Press, Cambridge (1992)
13. Koza, J.R., Bennett III, F.H., Andre, D., Keane, M.A.: Synthesis of topology and sizing of analog electrical circuits by means of genetic programming. Computer Methods in Applied Mechanics and Engineering 186(2-4), 459–482 (2000)
14. Lau, A., Musilek, P.: Immune programming models of cryptosporidium parvum inactivation by ozone and chlorine dioxide. Info. Sci. 179(10), 1469–1482 (2009)
15. McKinney, B., Tian, D.: Grammatical immune system evolution for reverse engineering nonlinear dynamic bayesian models. Cancer Inf. 6, 433–447 (2008)
16. Musilek, P., Lau, A., Reformat, M., Wyard-Scott, L.: Immune programming. Information Sciences 176(8), 972–1002 (2006)
17. O'Neill, M., Brabazon, A.: Grammatical differential evolution. In: Proceedings of the 2006 International Conference on Artificial Intelligence - ICAI 2006, Las Vegas, Nevada, USA, pp. 231–236. CSREA Press (2006)
18. O'Neill, M., Brabazon, A., Adley, C.: The automatic generation of programs for classification problems with grammatical swarm, vol. 1, pp. 104–110 (2004)
19. O'Neill, M., Ryan, C.: Grammatical evolution. IEEE Transactions on Evolutionary Computation 5(4), 349–358 (2001)
20. O'Neill, M., Ryan, C.: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language. Kluwer Academic Publishers, Dordrecht (2003)
21. Poli, R., Langdon, W.B., McPhee, N.F.: A Field Guide to Genetic Programming (2008)
22. Ryan, C., Collins, J., Neill, M.O.: Grammatical evolution: Evolving programs for an arbitrary language. In: Banzhaf, W., Poli, R., Schoenauer, M., Fogarty, T.C. (eds.) EuroGP 1998. LNCS, vol. 1391, pp. 83–95. Springer, Heidelberg (1998)
23. Smith, S.F.: A learning system based on genetic adaptive algorithms. Ph.D thesis, Pittsburgh, PA, USA (1980)
24. Smith, S.F.: Flexible learning of problem solving heuristics through adaptive search. In: Proc. 8th Int. Joint Conference on Artificial Intelligence (August 1983)

A New Algorithm Based on Negative Selection and Idiotypic Networks for Generating Parsimonious Detector Sets for Industrial Fault Detection Applications

Eduard Plett[1] and Sanjoy Das[2]

[1] Department of Engineering Technology, Kansas State University, Salina, KS, USA
[2] Department of Electrical and Computer Engineering, Kansas State University, Manhattan, KS, USA
{eplett,sdas}@k-state.edu

Abstract. Artificial Immune System (AIS) algorithms have been used for well over a decade to detect anomalies in computer security and data collection applications. They have also been used for real-time industrial fault detection applications, where they typically outperform traditional limit-checking algorithms in terms of anomaly detection ability. However, large computing overhead and large memory requirements make traditional AIS algorithms unsuitable for many applications. In this paper we propose a new AIS algorithm called Parsimonious Detector Set Algorithm which promises to greatly limit the effect of both constraints and make AIS algorithms suitable for a wider range of applications.

Keywords: Biologically Inspired Algorithm, Artificial Immune System, anomaly detection, negative selection, industrial applications, real-time applications.

1 Introduction

In many industrial fault detection applications, the goal is to monitor the system or process to ensure that the system or process is operating under normal conditions. Through testing or empirically, upper and lower limits for each variable are determined. The real-time measurements are then compared with the limits and the operator is alerted or the process is shut down automatically if any of the limits are violated.

A *limit-checking* system is able to detect sudden and pronounced changes (e.g. breakages), but it lacks the ability to detect gradual inching of multiple variables towards the abnormal area of operation indicating impending problems in the system. For example, a simultaneous rising of pressure and temperature in a storage tank, possibly indicating an uncontrolled reaction, can be easily detected by a human operator, but the limit-checking system typically lacks this type of intelligence and might not raise an alarm until it is too late to stop the reaction.

The problem with limit-checking systems is that each variable is treated as independent from all the others, which it usually is not. It essentially surrounds the normal

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 288–300, 2009.

region of operation with a rectangle (two dimensions), cube (three dimensions) or hyper-cube (more than 3 dimensions) and alerts the operator or stops the test automatically if any of the planes of the hyper-cube are violated. However, there is usually a large abnormal region, typically in the corners of the hypercube, which cannot be covered no matter how tightly the limits are set.

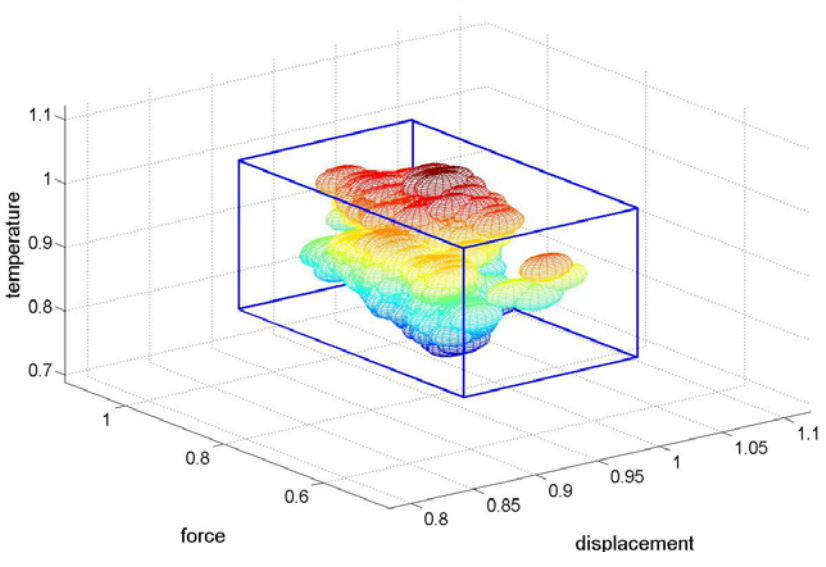


Fig. 1. Normal region of operation inside the limit box

There are alternative methods to detect anomalies. Some methods are based on statistical techniques, for example the one-class Support Vector Machines (SVM) [1], which, it can be argued, is a concept similar to the limit-checking algorithm, except that the hyper-planes are not limited to 90º.

In this paper we will mainly discuss methods based on the operation of the mammalian immune system. Artificial Immune Systems (AIS) algorithms do not treat individual variables as independent from each other, but as a complex "body," so they are able to detect multidimensional anomalous inputs long before individual limits in any particular direction are violated.

However, a common problem with using AIS algorithms for real-time industrial applications has been the large computing overhead and large memory requirements which eliminated many applications from consideration. In this paper we propose a new AIS algorithm called *Parsimonious Detector Set Algorithm* (PDSA) which, when tested with a particular application involving an automatic bearing testing machine, promises to improve anomaly detection ability in comparison to limit-checking algorithms while minimizing real-time computation and memory requirements compared to traditional AIS algorithms.

2 Anomaly Detection Using Artificial Immune Systems

Artificial Immune Systems (AIS) algorithms are based on the operation of the mammalian immune system, the primary objective of which is to recognize hostile organisms and destroy or otherwise neutralize them. The exact biological mechanism of the immune system is very complex, and several theories and mathematical models have been proposed to explain its operation, for example by Jerne [2] and Farmer et al. [3] AIS algorithms try to use the developed mathematical models of the immune system to solve various practical problems, for example detecting faults in squirrel cage induction motors [4], refrigeration systems [5], aircraft systems [6], and power systems [7]. AIS algorithms have also been used to detect anomalies in time series data [8] and to recognize patterns, for example the Indian Telugu characters [9], [10]. Most recently, AIS algorithms have been used to detect credit card fraud [11] and faults in rotating machinery [12].

There are numerous articles, books and tutorials written on the subject of Artificial Immune Systems, for example by Dasgupta [13], Aickelin and Dasgupta [14], de Castro and Timmis [15], Das et. al [16], Plett et. al [17]. This section only aims to give broad understanding of the subject necessary to follow the rest of the paper; therefore, in-depth immune system details not relevant to this paper have been omitted. A more thorough discussion is given on the new detection algorithm in section 2.5.

2.1 Negative Selection

In very general terms, one of the fundamental properties of the immune system is its ability to detect foreign cells called *antigens*, e.g. bacteria, viruses, etc. The antigens are detected if special detector cells called *antibodies* are able to bind to the antigens. Different antibodies will bind to different types of antigens, and this ability is called the *affinity* of the antibodies. The antibodies will not bind to the body's own cells, and the ability of the immune system to differentiate between the body's own cells and antigens is called the *Self-Nonself discrimination*.

In some artificial immune systems, for example in computer anti-virus programs, Self-Nonself discrimination is achieved by using positive characterization, i.e. by training the system to recognize known samples of malicious software. The virus and malware definitions have to be periodically updated to enable the system to identify new threats.

Some work [1], [18], [19], has also been done on using positive characterization, or *positive selection*, for anomaly detection. In this scheme, each normal or *Self* sample also functions as a detector, and inputs which lie outside a defined *Self-radius* are considered *Nonself* or anomalous. However, the huge size of the resulting detector set make it very difficult if not impossible to use for constrained industrial fault detection applications (more on the constraints later).

The mammalian immune system, in contrast, relies heavily on negative characterization or *negative selection*. The immune system continuously creates a large variety of antibodies. If an antibody binds to the body's own cells, it is eliminated to prevent autoimmune reactions. Otherwise, the antibody is released into the bloodstream. Once in the bloodstream, if an antibody binds to any cell, it is assumed to be foreign and is then destroyed by other cells of the immune system. Negative selection algorithms, in

particular the V-detector algorithms which will be described later, appear to better suit constrained industrial fault detection applications compared to positive selection algorithms and will be the main focus of this paper.

AIS algorithms based on negative selection typically work in the following way: a number of detectors are (usually randomly) generated to fill the entire detection region. Each detector is then presented to samples of normal data, to see if the detector erroneously classifies any as abnormal. If it does, the detector is discarded; otherwise it is inserted into the detector set until a large enough number of valid detectors is found. During the detection phase, if an input data point is detected by any detector, then it is classified as anomalous. Otherwise, an input data point that cannot be detected by any detector is considered normal.

2.2 Real-Valued Negative Selection Algorithms

Real-valued negative selection (RNS) is a special case of negative selection in which inputs and detectors are represented as multi-dimensional points with real-valued coordinates.

A basic RNS algorithm works as follows: during the training phase, samples of normal data are used to create a so called "Self" region, which is the region formed by the union of all hyper-spheres around each Self point. Then random detector points are generated within the detection region. The Euclidean distances between detectors and the points in the Self region are computed. If any of the distances is smaller than the radius around the Self-points, it means that the corresponding detector is inside the Self region and the detector is discarded. Each accepted detector has its own predetermined radius of coverage, and the union of all hyper-spheres around the detectors forms the so-called "Nonself" region. During the detection phase, the Euclidean distance between an input point and each detector point is calculated. If any of the distances is smaller than the detector radius, then the input lies in the Nonself region and is classified as anomalous or abnormal.

The major drawback of the basic RNS algorithm is that, since the detectors are random, some areas might be covered by multiple detectors while some areas might not be covered at all. To optimize the placement of the detectors to achieve better coverage, a *self-organizing* RNS algorithm [20] also begins with a set of detectors with randomly assigned coordinates. However, instead of immediately eliminating detectors that lie inside the Self region, it attempts to push those detectors to the Nonself region. Furthermore, it attempts to space the detectors, which are already outside the Self region, away from each other to prevent redundant detectors. Genetic algorithms have also been used [21], [22], [23] to find the optimal placement of detectors to maximize coverage. Nevertheless, the large number of detectors make the basic RNS algorithm unsuitable for constrained industrial fault detection applications.

2.3 The V-Detector Algorithm

The V-detector (Variable-sized detector) algorithm [24], [16] is another variation of the basic RNS algorithm, with one major difference: to maximize the coverage of each detector, the detector radii are not fixed, but each detector is now assigned a radius which is equal to its distance to the Self-region. Their locations do not have to

be adjusted as in the self-organizing RNS algorithm, which makes the algorithm less computationally complex, but it covers nearly the same region with a substantially reduced number of detectors.

The basic V-detector algorithm works as follows: in each iteration, a detector, whose location is represented by a multi-dimensional vector **x** with coordinates x_1, x_2, x_3, etc., is randomly picked within the detection region. The Euclidean distances between this detector and all points **S** in the Self region are then computed. If any of the distances is smaller than the radius r_S around the Self-points, then the detector is within the Self region and is discarded, otherwise it is accepted in the detector set **D**. Each accepted detector is associated with a radius r which is equal to the shortest distance to a Self-point minus the radius around that Self-point. All *r's* are combined to form a radius vector $\mathbf{r_D}$. A threshold parameter θ is usually included to create a gap between the Self region and the Nonself region, which will make the algorithm more robust to false anomalous classifications. The basic V-algorithm pseudo-code is included in section 2.5.

2.4 The Proliferating V-Detector Algorithm

This variation of the V-detector algorithm, introduced by Gui et. al [7] has two distinct stages: the generation stage and the proliferation stage. The generation stage is essentially the same as in the basic V-detector algorithm, except that typically fewer detectors are generated initially. During the proliferation stage each detector attempts to place offspring detectors on the circumference of its hyper-spheres by adding unit vectors of length r_D to the position of the detectors. Each detector in the detector set may be allowed to proliferate multiple times, or until the Nonself area is sufficiently covered. The threshold parameter θ can be lowered in discrete steps to improve the acceptance of the new detectors closest to the Self-Nonself boundary region. Steadily decreasing θ will result in increasingly smaller, but strategically placed offspring hyper-spheres around the Self-Nonself boundary region.

The proliferating V-detector algorithm deterministically fills potential gaps in the Self-Nonself border region, which improves the detection ability of the algorithm for complex Nonself spaces. However, it usually results in a very large number of detectors.

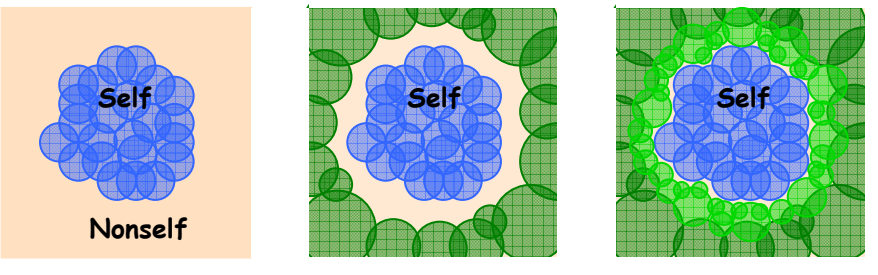


Fig. 2. (a) Self and Nonself regions. (b) Basic V-detectors (c) Basic & Proliferated V-detectors.

2.5 The Parsimonious Detector Set Algorithm (PSDA)

Multiple attempts, one of which is the region-restricted V-detector algorithm [25], have been made recently to exploit the coverage advantages of the V-detector algorithms while reducing the number of detectors necessary to achieve the maximum coverage. This section will introduce a new scheme to reduce the number of detectors while maximizing coverage.

We propose a new algorithm that alternates between two phases: a *generation* phase and a *suppression* phase. During the generation phase, the algorithm creates new antibodies using the traditional V-detector method. As before, antibodies that are found to be within the Self region are not allowed to enter the detector set to ensure that the detectors cover only the Nonself region.

Then, instead of optimally relocating the existing detectors to maximize coverage, as was done in earlier attempts [26], [27], the second phase of the proposed method attempts to do the opposite: it removes superfluous or nearly superfluous detectors whose regions of coverage overlap with other detectors. This is accomplished by mimicking the dynamic interaction of antibodies in idiotypic immune networks [28], [29], [30].

This dynamic interaction regulates the concentration of the antibodies. The concentration of antibodies decreases with time due to natural death. To offset this process, new antibodies are added to increase their concentrations in the bloodstream. Antibodies that provide a larger coverage – usually in terms of the antigens detected – are preferred in this process. Antibodies also tend to suppress and stimulate the concentrations of one another. Those that recognize other antibodies receive stimulations, increasing the rates of growth of their concentrations, whereas those that are recognized by others are suppressed. This interactive behavior can be expressed succinctly in the following manner,

$$\frac{d(\text{concentration}_i)}{dt} = -\text{death rate}_i + \text{coverage}_i + \sum_j i \text{ detects } j - \sum_j i \text{ is detect by } j \quad (1)$$

where i and j are any two antibodies[2].

For our algorithm, the third term in the equation above is dropped, leading to a more specialized form shown below,

$$\frac{dz_i}{dt} = -\tau z_i + cvr_i z_i - z_i \sum_{j \neq i} \frac{ovl_{i,j}}{cvr_i} z_j. \quad (2)$$

In the above equation, z_i and z_j are the concentrations of the i^{th} and j^{th} detectors, and τ is a constant associated with the death rate. The quantity cvr_i is the coverage of the i^{th} detector. For a detector located entirely within the Nonself region, this is simply the hypervolume, which can be readily calculated ($1.33\pi r_i^3$ in three dimensions, r_i being the detector radius). When the detectors are only partially within the detection region,

Monte Carlo estimates can be used for low-dimensional problems. The other quantity, $ovl_{i,j}$, is the amount of overlap between their covered regions. This is illustrated in the figure below.

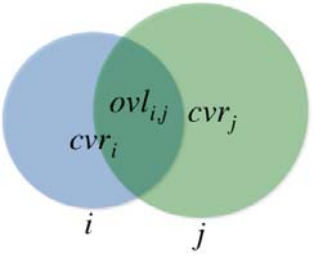


Fig. 3. Coverage and overlap of two detectors

The PDSA algorithm works as follows: first a preset number of random detectors are created, and the coverage of each detector and their overlap with each other are calculated. Then detectors are subject to idiotypic concentration adjustments according to equation 2. After several iterations, detectors whose concentration is below a threshold are removed. New detectors are then added to the detector set, and all are again subjected to idiotypic interaction. The process is repeated until the desired number of detectors is reached or the Nonself area is sufficiently covered.

Below are the pseudo-codes for the generation and suppression algorithms:

```
function D = generate_V_detectors (S, r_S, θ, N)
 begin
   for i = 1 to N      // N = #of attempts
     x = rand()        // random detector coordinates
     r = min_{s∈S} (||x, s||) - r_S - θ
     if r > 0
       D = D ∪ {(x, r)}
     endif
   endfor
 end

function D = reduce_detectors (D, N, threshold)
 begin
   for i = 1 to N      // N = #of detectors
     (x_i, r_i) = i^th element of D
     cvr_i = find_coverage(x_i, r_i)
     for j = 1 to N
       (x_j, r_j) = j^th element of D
       ovl_{i,j} = find_overlap(x_i, r_i, x_j, r_j)
     endfor
   endfor
   for t = 1 to max_time  //assume unit time increments
     for i = 1 to N
```

```
      dz_i= z_i(-τ + cvr_i - (Σ_{j≠i} ovl_{i,j} z_j)/cvr_i)
      z_i = z_i + dz_i
    endfor
  endfor
  for i = 1 to N
    if (z_i < threshold)
      D = D - {(x_i, r_i)}
    endif
  endfor
end
```

3 Anomaly Detection for an Automatic Bearing Testing Machine

3.1 Problem Description

An automatic bearing testing machine performs component design verification for heavy duty construction equipment. The machine simulates real-life operation by applying cyclical forces to a bearing which is subjected to rotation and contamination with construction debris and water. A test with a standard production bearing establishes the baseline life expectancy (measured in load cycles) of a bearing. New bearing designs are then tested under identical conditions and only bearings that show significant improvement in life expectancy over baseline bearings are then subjected to real-life tests for final acceptance.

A critical task of the test procedure is to determine when a bearing reaches the end of its useful life. This is done by analyzing real-time data and comparing it with baseline numbers. A total of 19 parameters for 4 different bearings are recorded at specific time intervals, but the three most significant parameters are: the play (axial displacement) of the bearing, the force to rotate the bearing, and the temperature of the bearing. These three parameters are a good indicator of the status of the bearing. As the test progresses, they collectively inch upward, indicating a worn bearing. An automatic shutdown is implemented if any of the variables exceed its limits.

As discussed in the introduction, the problem with this limit-checking algorithm is that it is able to detect sudden changes (e.g. breakage), and bearings which are far beyond their useful life. However, it cannot detect the collective migration towards the abnormal area of operation which indicates a worn bearing. Therefore, an operator - instead of the machine - usually makes the call to stop the test. In order to provide this machine with a similar type of intelligence, i.e. make it able to detect when the system leaves the normal region of operation, different AIS algorithms were tried.

However, when considering AIS algorithms for real-time control systems, two constraints have to be considered. One constraint is the typically small controller memory, which limits the ability to store a large number of multidimensional, real-valued, floating-point detectors. The second constraint is the maximum scan time of the controller, which again might be exceeded due to the significantly larger computation overhead required for AIS algorithms with a large number of detectors. Therefore, the size of the detector set has to be minimized while the coverage needs to be

maximized. The PDSA algorithm so far appears to show the greatest promise in achieving both objectives.

3.2 Results and Discussion

The testing of bearing designs has been performed for more than 5 years, and several million data points have been obtained. For this paper, a specific bearing test with about 1600 data points has been used to test the AIS algorithms. It has 344 anomalous data points, some of them resulting from erroneous readings, the rest resulting from worn bearings. As expected, with fairly aggressive limit settings, the simple-limit checking algorithm had little problem detecting erroneous readings and grossly anomalous data, detecting about 84% of all anomalies.

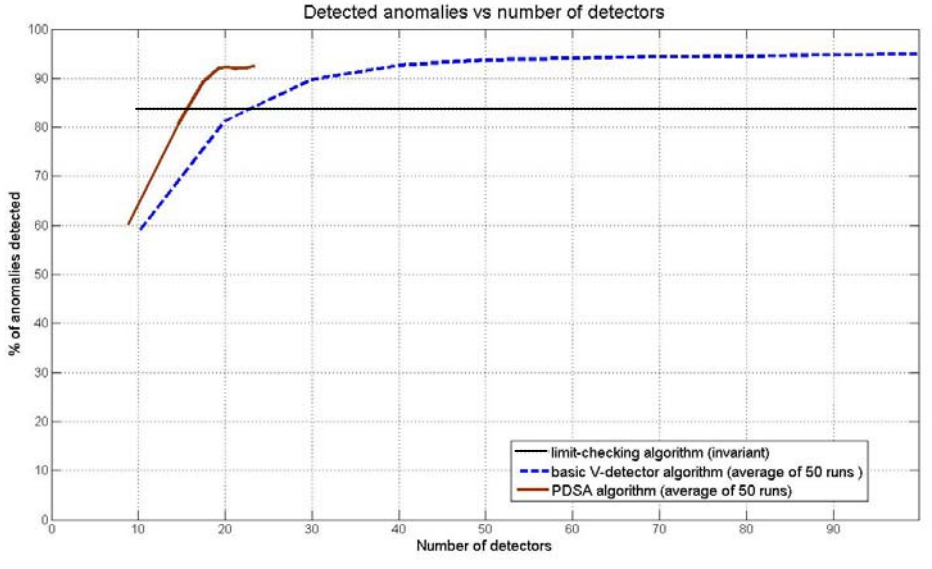


Fig. 4. Detected anomalies vs. number of detectors (average of 50 runs)

The simple V-detector algorithm was also able to detect erroneous readings, but also most anomalous data points in the corners of the limit hypercube, averaging about 95% of all anomalies detected with 100 detectors.

The PDSA algorithm averaged about 93% of all anomalies detected, but accomplished this feat utilizing less than ¼ of the detectors in comparison to the basic V-detector algorithm.

The following plot shows the percentage of anomalies detected, averaged over 50 runs. Note that both algorithms start with the same number of random detectors in each iteration, but the PDSA algorithm filters out superfluous or nearly superfluous detectors.

Below is the standard deviation plot for the same 50 runs. Note that with a low number of detectors, the randomness of placement results in a large standard

deviation. This means that, depending on the placement, 10 detectors can achieve as high as a 90% detection rate, but also as low as a 30% detection rate. To ensure a consistently high detection rate, the basic V-detector algorithm therefore relies on a large number of detectors. The advantage of the PDSA algorithm is obtaining a consistently high detection rate with a greatly reduced number of detectors.

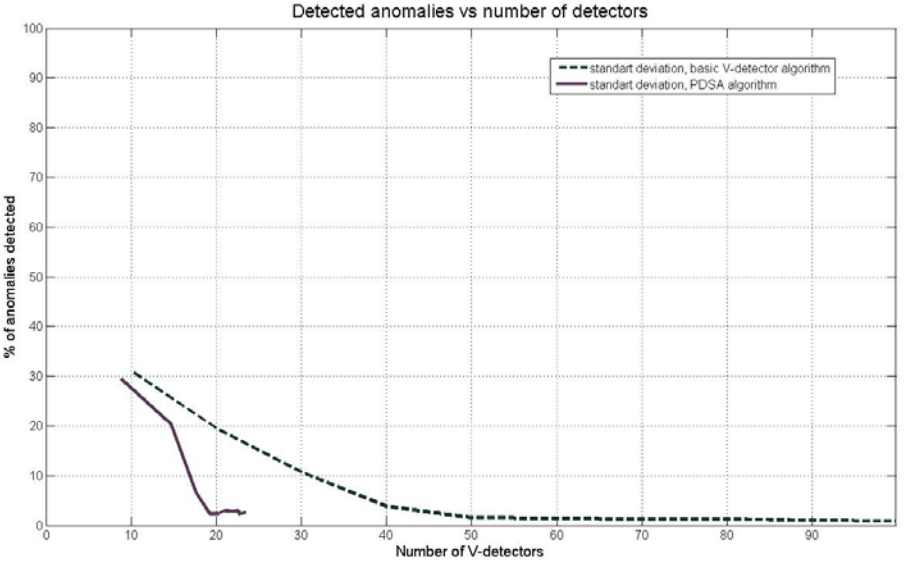


Fig. 5. Standard Deviation plot of 50 runs

In summary, the PDSA algorithm shows great promise for this application as it resulted in dramatically fewer detectors, while still attaining a comparable coverage area compared to the basic V-detector algorithm.

However, there is still much room for future research. For example, the proliferating V-detector algorithm in combination with the PDSA algorithm was also tried, but did not result in noticeable improvement, mainly because the smaller offspring detectors were mostly suppressed by the idiotypic interaction. The idiotypic interaction naturally favors detectors with large radii (large coverage), which also results in larger gaps between the hyper-spheres, in particular close to the Self-Nonself region. The coverage can be improved by allowing more detectors and by weighting the factors in equation 2 differently. In future research, we plan to optimize the factors and / or add additional factors in equation 2 to achieve the proper balance between rewarding *unique* coverage and penalizing overlap.

The final goal of this study is to implement an intelligent and efficient anomaly detection algorithm on actual hardware (a PLC controller) to monitor the device. The final algorithm will contain preprocessing steps necessary to minimize false alarms, and have a parsimonious, adaptable detector set to replace the current limit checking algorithm.

4 Conclusion

This paper introduced a new Artificial Immune System (AIS) algorithm based on negative selection and idiotypic networks to generate a parsimonious detector set for industrial fault detection applications. It demonstrates the advantage of AIS algorithms compared to traditional limit-checking algorithms in terms of detection ability. It also outlines the limitations of AIS algorithms for real-time industrial applications and shows how the new Parsimonious Detector Set Algorithm can overcome these limitations.

This paper is an *application* paper and focuses on a specific application involving an automatic bearing testing machine. The final goal of this study is an intelligent and efficient anomaly detection system implemented with hardware constraints for this particular application. AIS algorithms, specifically the V-detector algorithms, were showing the best results up to this point, but this paper introduces a method to make the basic V-detector algorithm even more efficient. Indeed it demonstrates that the PDSA algorithm outperforms the traditional limit-checking algorithm and shows comparable performance to the traditional V-detector algorithm with a drastically reduced number of detectors and the associated computing overhead and storage requirements for this application. We would also venture to say that the PDSA algorithm is suitable for a large number of industrial fault detection applications with similar hardware constraints.

References

1. Stibor, T., Mohr, P., Timmis, J., Eckert, C.: Is Negative Selection Appropriate for Anomaly Detection? In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). ACM Press, Washington (2005)
2. Jerne, N.K.: Towards a network theory of the immune system. Annals of Immunology 125(C), 373–389 (1973)
3. Farmer, J.D., Packard, N.H., Perelson, A.S.: The immune system, adaptation, and machine learning. Physica D 22, 187–204 (1986)
4. Branco, P.J.C., Dente, J.A., Mendes, R.V.: Using immunology principles for fault detection. IEEE Transactions on Industrial Electronics 50(2), 362–373 (2003)
5. Taylor, D.W., Corne, D.W.: An investigation of the negative selection algorithm for fault detection in refrigeration systems. In: Timmis, J., Bentley, P.J., Hart, E. (eds.) ICARIS 2003. LNCS, vol. 2787, pp. 34–45. Springer, Heidelberg (2003)
6. Dasgupta, D., KrishnaKumar, K., Wong, D., Berry, M.: Negative selection algorithm for aircraft fault detection. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 1–13. Springer, Heidelberg (2004)
7. Gui, M., Das, S., Pahwa, A.: Procreating V-detectors for Nonself Recognition: An Application to Anomaly Detection in Power Systems. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), London, UK, pp. 261–268 (2007)
8. Nunn, I., White, T.: The application of antigenic search techniques to time series forecasting. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), Washington, DC, pp. 353–360 (2005)

9. Ji, Z., Dasgupta, D.: Real valued negative selection algorithm using variable sized detectors. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 287–298. Springer, Heidelberg (2004)
10. Ji, Z., Dasgupta, D.: Applicability issues of the real-valued negative selection algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference, Seattle, WA, pp. 111–118 (2006)
11. Gadi, M.F., Wang, X., Lago, A.P.: Credit Card Fraud Detection with Artificial Immune System. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 119–131. Springer, Heidelberg (2008)
12. Strackeljan, J., Leiviskä, K.: Artificial Immune System approach for the Fault Detection in Rotating Machinery. In: Proceedings of the International Conference on Condition Monitoring & Machinery Failure Prevention Technologies, Edinburgh, UK (2008)
13. Dasgupta, D. (ed.): Artificial Immune Systems and their Applications. Springer, Heidelberg (1999)
14. Aickelin, U., Dasgupta, D.: Artificial Immune Systems, ch. 13, http://eprints.nottingham.ac.uk/336/1/05intros_ais_tutorial.pdf
15. Castro, L.N., De, T.J.: Artificial Immune Systems as a New Soft Computing Paradigm. Soft Computing - A Fusion of Foundations, Methodologies and Applications 7(8), 526–544 (2003)
16. Das, S., Gui, M., Pahwa, A.: Artificial Immune Systems for Self-Nonself Discrimination: Application to Anomaly Detection. Advances of Computational Intelligence in Industrial Systems 116, 231–248 (2008)
17. Plett, E., Das, S., Li, D., Panigrahi, B.K.: Artificial Immune Systems Approaches for Anomaly Detection. In: Soria, E., Martin, J.D., Magdalena, R., Martinez, M., Serrano, A.J. (eds.) Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques. IGI Global (in press, 2009)
18. Stibor, T., Timmis, J., Eckert, C.: A Comparative Study of Real-Valued Negative Selection to Statistical Anomaly Detection Techniques. In: Jacob, C., Pilat, M.L., Bentley, P.J., Timmis, J.I. (eds.) ICARIS 2005. LNCS, vol. 3627, pp. 262–275. Springer, Heidelberg (2005)
19. Stibor, T., Timmis, J.: Comments on Real-Valued Negative Selection vs. Real-Valued Positive Selection and One-Class SVM. In: Proceedings of the Congress on Evolutionary Computation (CEC). IEEE Press, Singapore (2007)
20. Gonzalez, F.A., Galeano, J.C., Rojas, D.A., Veloza-Suan, A.: Discriminating and visualizing anomalies using negative selection and self-organizing maps. In: Jacob, C., Pilat, M.L., Bentley, P.J. (eds.) GECCO 2005: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, vol. 1, pp. 297–304. ACM Press, Washington (2005)
21. Dasgupta, D., Gonzalez, F.: An immunity-based technique to characterize intrusions in computer networks. IEEE Transactions on Evolutionary Computation 6(3), 281–291 (2002)
22. Balachandran, S., Dasgupta, D., Nino, F., Garrett, D.: A Framework for Evolving Multi-Shaped Detectors in Negative Selection. In: IEEE Symposium on Foundations of Computational Intelligence, Honolulu, Hawaii, pp. 401–408 (2007)
23. Gao, X.-Z., Ovaska, S.J., Wang, X.: A GA Based Negative Selection Algorithm. International Journal of Innovative Computing, Information and Control 4(4), 971–979 (2008)
24. Ji, Z., Dasgupta, D.: Real valued negative selection algorithm using variable sized detectors. In: Deb, K., et al. (eds.) GECCO 2004. LNCS, vol. 3102, pp. 287–298. Springer, Heidelberg (2004)

25. Plett, E., Das, S.: A Region Restricted V-Detector Algorithm for Automatic Pin-Joint Testing Machines. In: Proceedings: International Conference on Artificial Intelligence, Las Vegas, Nevada (2008)
26. González, F.A., Dasgupta, D.: An immunity-based technique to characterize intrusions in computer networks. IEEE Trans. Evolutionary Computation 6(3), 281–291 (2002)
27. Gonzalez, F.A., Galeano, J.C., Rojas, D.A., Veloza-Suan, A.: Discriminating and visualizing anomalies using negative selection and self-organizing maps. In: Jacob, C., Pilat, M.L., Bentley, P.J. (eds.) GECCO 2005: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, vol. 1, pp. 297–304. ACM Press, Washington (2005)
28. Cayzer, S., Aickelin, U.: A Recommender System based on Idiotypic Artificial Immune Networks. Journal of Mathematical Modelling and Algorithms 4(2) (2005)
29. Aickelin, U., Chen, Q.: On Affinity Measures for Artificial Immune System Movie Recommenders. In: Proceedings RASC 2004: The 5th International Conference on Recent Advances in Soft Computing, Nottingham, UK (2004)
30. Whitbrook, A.M., Aickelin, U., Garibaldi, J.M.: Idiotypic Immune Networks in Mobile-Robot Control. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 37(6), 1581–1598 (2007)

Parametric Modelling of a Flexible Plate Structure Using Artificial Immune System Algorithm

S.M. Salleh and M.O. Tokhi

Department of Automatic Control and Systems Engineering,
The University of Sheffield, United Kingdom
cop07sbm@sheffield.ac.uk

Abstract. Parametric modelling of dynamic systems may benefit from advantages of biologically-inspired immune system, which maintains its own system against dynamically changing environments through the interaction between lymphocytes and/or antibodies. In this paper, artificial immune system with clonal selection algorithm and artificial immune network are used to identify the unknown parameters characterising a flexible plate system. The identification is performed on basis of minimising the mean-squared output error and is assessed with correlation tests and in time and frequency domains. The approach is tested with three different disturbance signals. Simulation results demonstrate the potential of artificial immune system as promising technique with fast convergence and less computational time in comparison to binary-coded genetic algorithm.

Keywords: Artificial immune system, parametric modelling, flexible plate.

1 Introduction

A number of techniques have been devised by researchers to determine the input-output behaviour of a system (optimization problem) utilizing artificial intelligence approaches. These include genetic algorithm (GA), particle swarm optimisation (PSO), artificial neural networks and artificial immune system (AIS). Even though there are tradeoffs between solution quality and computational cost, intelligent techniques are probably desired in many cases where it is difficult to obtain a good model structure using traditional system identification techniques. Among the algorithms, GA, and PSO have been used in time varying situations specifically in parametric modeling [1], [2]. However, the use of AIS has been scarce for parametric modelling. Attempting to use AIS in parametric modelling is a challenge to confront with immense generations and longer computational times in comparisons to other intelligent approaches.

Parametric modelling is defined as the process of estimating parameters of a model characterising a plant. The technique can be regarded as an optimization problem; minimizing the error between the predicted (model) output and the measured (plant) output. Parametric modelling can include both the parameter estimates and the model structure. Statistical validation procedures, based on correlation analysis, are utilised to validate parametric models.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 301–314, 2009.

This modelling approach has been implemented in identifying the parameters of flexible structures such as fixed-free beam and plate with all edges clamped [3]. Several interests in flexible structures are due to light weight, lower energy consumption, smaller actuator requirement, low rigidity requirement and less bulky design. These advantages lead to extensive usage of flexible plates in various applications such as space vehicles, automotive industries, and the construction industry. Modelling is the first step in a model-based control development of a system. Accordingly, the accuracy of the model is crucial for the desired performance of the control system.

The utilisation of bio-inspired AIS using clonal selection and immune network models in optimization problems has been reported in various applications such as combinatorial and multimodal optimization [4], generalized geometric programming [5] and pattern recognition. However, AIS models are not limited to function optimization, and are applicable to areas like data mining (classification and clustering) and search for optimal in control using artificial immune network (AiNet) models, and computer security using negative selection model. In this case, representation of AIS needs to be carefully considered, appropriate to the particular problem.

The clonal selection algorithm and AiNet were proposed by de Castro and Von Zuben [4]. They firstly adopted the clonal selection principle to optimization tasks after using in machine-learning and pattern recognition problems. The principle has been simplified in terms of similarity between the cell and its receptor, and this has made the editing and cell replacement process equivalent [4]. They described that standard GA and evolutionary strategies are different in individual's fitness evaluation through genetic operators towards the best candidate solution and self-adapting parameters under mathematical framework respectively. In contrast, CLONALG takes into account the cell affinity akin to individual's fitness, in order to define the mutation rate to be applied to each member of the population. The tasks involved maximization of two unidimensional functions in multimodal optimization and travelling salemans problem (TSP) in combinatorial optimization.

Malim et, al. [6] applied clonal selection algorithm and other two models, namely AiNet and negative selection to solve the optimization problem in university timetabling. Clonal selection has been successfully applied to solve the problem in examination timetables; nevertheless, clonal selection was less effective in solving course timetabling. Meanwhile, negative selection was effective in both problems. In particular, AIS representation has potential in tackling highly constrained optimization tasks.

In this work, AIS with CLONALG and AiNet are adopted for parametric modeling of a flexible plate structure in comparison to a binary-coded GA. The rest of the paper is structured as follows: Section 2 describes the flexible plate system and formulates the problem. Section 3 presents AIS algorithm and its parameters used in this work. Section 4 presents implementation of the algorithms in modeling the system. Section 5 discusses the result of the model validity through input/output mapping, mean square of output error and frequency domain response. Parametric modelling is also confirmed with convergence of fitness values and time run. Finally, the paper is concluded in Section 6.

2 System Description and Problem Formulation

Consider a thin square plate of length *a*, along x-axis, width, *b*, along y-axis and thickness, *h* along z-axis, undergo small lateral motion. This condition is illustrated in Fig. 1.

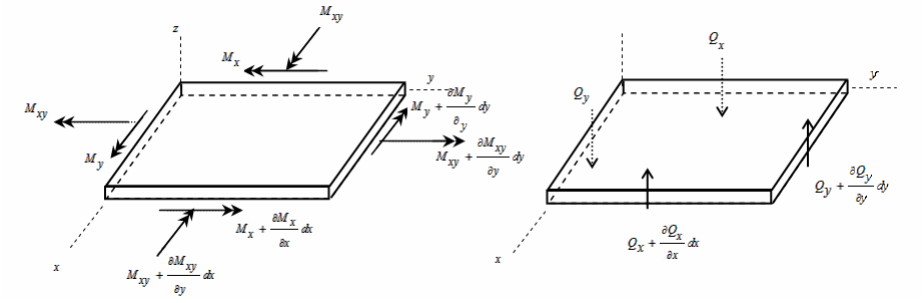


Fig. 1. Moments and shear forces of the plate element

From the bending and twisting moments acting on the plate due to the intensity of the load q, the plate is assumed to experience a small deflection, w. Summing all the forces in the z direction and considering the effect of shear forces Q_x and Q_y in terms of bending moments, M_x, M_y and M_{xy} yield [7] :

$$\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4} + \frac{\rho}{D}\frac{\partial^2 w}{\partial t^2} = \frac{q}{D} \tag{1}$$

where w represents lateral deflection in the z direction, ρ is mass density per unit area, $q=q(x,y)$ is transverse external force per unit area, $\partial^2 w/\partial^2 t$ is acceleration in the z direction, $D = Eh^3/(12x(1-v^2))$ is the flextural rigidity, v is Poisson ratio, h is thickness of the plate and E is Young Modulus.

Eq. (1) represents the dynamic equation characterising the behaviour of the flexible plate. Solution of this equation with (FD) will result in a representation in a matrix form by considering the boundary conditions. The central finite FD method is used to discretise the governing dynamic equation of the plate with no damping. The resulting relation as transformed into a state space equation is given as

$$\boldsymbol{W}_{i,j,k+1} = (\mathbf{A} + 2_{ijk})\boldsymbol{W}_{i,j,k} + \boldsymbol{B}\boldsymbol{W}_{i,j,k-1} + \boldsymbol{CF} \tag{2}$$

where 2_{ijk} represents the diagonal elements of $(2/c)$, $C=(\Delta t^2/\rho)$, $c = -DC$, and $\boldsymbol{W}_{i,j,k+1}$ is the deflection of grid points $i = 1, 2, \ldots\ldots, n+1$ and $j = 1, 2, .., m+1$ at time step k+1. $\boldsymbol{W}_{i,j,k}$ and $\boldsymbol{W}_{i,j,k-1}$ are the corresponding deflections at time steps k and k-1 respectively. $\mathbf{A}$ is constant $(n+1)(m+1)\times(n+1)(m+1)$ matrix whose entries depend on physical dimensions and characteristics of the plate, $\boldsymbol{B}$ is an identity matrix and C is a scalar related to the given input and $\boldsymbol{F}$ is an $(n+1)(m+1)\times 1$ matrix known as the forcing

matrix. The algorithm is implemented in Matlab/SIMULINK with applied external force or disturbance to the plate. The boundary condition is reconstructed to form deflection in a matrix form[8]. This is transformed into discrete state-space equation. A state space formulation can be generally constructed by referring to the matrix formulation with state dimension N, m inputs and p outputs as:

$$\begin{aligned} \boldsymbol{x}(n+1) &= \boldsymbol{P}\boldsymbol{x}(n) + \boldsymbol{Q}\boldsymbol{u}(n) \\ \boldsymbol{y}(n) &= \boldsymbol{R}\boldsymbol{x}(n) + \boldsymbol{S}\boldsymbol{u}(n) \end{aligned} \tag{2}$$

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ I_{NxN} & 0_{NxN} \end{bmatrix}, \quad \boldsymbol{Q} = \begin{bmatrix} C \\ 0_{Nx1} \end{bmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} I_N & 0_N \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} 0_{2N} \end{bmatrix}$$

$$\boldsymbol{u}(n) = [....., f, ...]^T, \quad \boldsymbol{y}(n) = [x(1,n)...x(N,n), x(1,n-1)...x(N,n-1)]$$

where u(n) is the input to the system, $\boldsymbol{x}(n)$ is the state of the system and $\boldsymbol{y}(n)$ is the output of the system, and $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ matrices are calculated in FD formulae as above.

3 Immune System and AIS Algorithm

The immune system is a system that protects the body from foreign substances and pathogenic organisms by producing the immune response. The immune system has two types of response: primary and secondary. The primary response is responsible for recognition and destruction of intruder like pathogens (which produce antigens such as bacteria, viruses, etc.) and attack the organisms for the first time. At this stage, the immune system learns about the antigen, thus preparing the body for any further invasion from that antigen. The secondary response is based on memory and ability to remember previously recognized pathogens and in this way it reacts more rapidly than the primary one. Both mechanisms harmonize each other with powerful recognition and elimination tools of different microbes, viruses, etc. AIS is known as 'computational immunology' [9] and defined as 'adaptive systems inspired by theoretical immunology and observed immune functions, principles and models, which are applied to complex problem domains' [10]. The algorithms inspired by the immune system include clonal selection, immune network and negative selection.

3.1 CLONALG Algorithm

Clonal selection is the process of adaptation of cell of the human system to the new patterns that are based on the lyhmpocytes which are generated in the bone marrow (called B-cell) to the matching pathogens [11]. The B-cells receptors called antibodies match the pathogens when they intrude in the organism. The *antibodies* which match the pathogens bind to them and cause the pathogens to be destroyed. Each element in AIS is called antibody, where antibody is referred to B-cell and its receptor. With the aim of parametric modelling, the algorithm used in this work is based on the CLONALG for solving multimodal optimisation, [4]. However, some modifications to the scheme are presented in this work. Clonal-selection probability is in direct proportion to a set of antibody affinity. Each set of antibody encloses feasible a and b variables of the parametric model. The antibodies with higher affinity have higher

chance to be selected and define its clonal rate. In this work, higher affinity means lower error between the actual output and the predicted output generated by feasible antibodies. The best antibodies with higher affinity remain in the population to improve recognition of minimum error during future generations.

The number of clones produced is proportional to the affinity. The higher the value of the affinity function the larger the number of clones. The total number of clones is generated for all *n* antibodies according to

$$N_C = \sum_{i=1}^{n} round\left(\frac{\beta * n}{i}\right) \tag{3}$$

where: i=1,2..,N, β– multiplying factor of the clones' number, n – total number of individuals chosen for cloning, N – size of population, and n = N. However, the hypermutation rate is inversely proportional to their affinity, where higher affinity gives lower mutation rate and vice versa. Applying the Hamming shape-space, the chosen bitstring length was L = 22, for a precision of six decimal places. After few best individuals are chosen from the population, the individuals with small change in population of antibodies will be replaced. Size of population is set by 50 antibodies as more antibodies chosen will slower the computational time. Other parameters of CLONALG is chosen as follows; Clones multiplying factor, β = 0.1, Hypermutation control parameter, pmr = 0.9, Hypermutation probability, pm = 0.01, size of population, N = 50 and maximum number of generation = 100.

3.2 AiNet Algorithm

AiNet was inspired by the idiotypic network formerly introduced by Jerne's. In AiNet algorithm, Jerne's theory described the discrimination between antibody and antigen cells. The interaction between immune cells includes both positive and negative recognitions. Positive, when an antibody recognizes an antigen or another antibody and negative, when an antibody is recognized by another immune cell. Positive recognition results in cell proliferation, activation and antibody secretion. Meanwhile negative recognition would lead to suppression. The structure of network models can be presented as in Fig. 2 and described as ;

$$C_{new} = C_{st} - C_{sup} + C_A \tag{4}$$

where C_{new} is the network variation, C_{st} is the network stimulation, C_{sup} is the network suppression, C_A is Ab-Ag recognition. The application of AiNet was firstly proposed in performing data mining [12], followed by optimizing multimodal function by [13]. AiNet has some interesting features namely with adjustable population/network sizes, searching for local and global optima and has the ability to maintain the optimum. However due to high computational cost during objective function assessments the use of AiNet may be bounded. Ab-Ag recognition is not considered in this work, only antibodies are used in modeling the system. The antibodies are in the same range as applied in CLONALG where $\Re \in [-1,1]$. The AiNet properties applied include: Suppression Threshold, σ_s = 0.001, Initial number of network cells = 20, Percentage of new cells =10 %, Range of variables = [-1,1], Population size = 100 and maximum number of generation = 100.

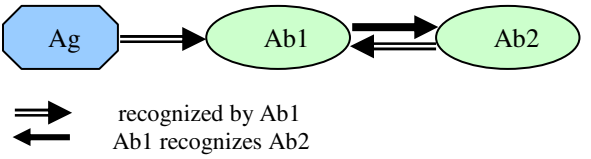


Fig. 2. The immune network idea, [10]

3.3 The Affinity Measures

The affinity measures specify the best solution of antibodies obtained from the parametric modelling problem. It can be opted as termination criteria, however, this usually increases the computational time. In this study, a fitness function is used as affinity measure of the algorithm, whereas number of generations is used as stopping criterion. The fitness function, X, or mean square error (MSE) is set to minimize

$$X = \min \left(\frac{1}{n} \sum_{i=1}^{n} \left(\left| y(i) - \hat{y}(i) \right| \right)^2 \right) \tag{5}$$

where *y(i)* is the actual system output subjected to a disturbance signal, $\hat{y}(i)$ is the response of the estimated system under the same disturbance, and i=1,2,…,n ; n is total number of input/output sample pairs. Predefined a maximum number of generations is used as stopping criterion for all algorithms.

4 Parametric Modelling and Simulation Model

The transfer function of the model used corresponds to the ARMA model structure by neglecting the noise, η term;

$$\hat{y}(k) = -a_1 y(k-1) - \ldots - a_4 y(k-4) + b_0 u(k-1) + \ldots + b_3 u(k-4) \tag{6}$$

Once the model is determined, the model needs to be verified whether it is well enough to represent the system. The correlations tests including autocorrelation of the error, cross correlation of input-error, input*input –error are carried out to test the model validity. The first four variables are assigned to $b_0,\ldots,b_3$ and the next four to $a_1,\ldots,a_4$ as indicated in Eq. (6) and taking the variables of best particle with actual input and output data. Each simulation is observed over 7000 samples of data for each set.

The physical parameters of the plate considered comprise are detailed in Table 1. With all edges of the plate structure are clamped, the sampling time of $\Delta t = 0.001$s is used to satisfy the stability requirement of the FD algorithm and is sufficient to cover a broad range of dynamics of the flexible plate. The response of the plate is considered over 7s. A finite duration step input force with amplitude q = 5.8945 x 10^{-8}N, is applied to the centre-point of the plate from t = 0.2 to 0.5 seconds, and the response of the plate is measured at an observer point. Random and pseudo random binary sequence (PRBS) signals are also used as exciting signals (see Fig.4). The input-output data extracted from the plate at detector and observer points respectively (see Fig. 3),

will be used for identification of dynamic behaviour of the flexible plate with AIS techniques. From observation, the first five resonance modes of vibration of the plate found from the spectral density of the predicted output of the AIS model are 9.971 rad/sec, 34.51 rad/sec, 56.76 rad/sec, 78.23 rad/sec and 99.71 rad/sec.

Table 1. Parameters of the flexible plate

Parameter	Value
Length, b	1.000 m
Width, a	1.000 m
Thickness, h	0.0032004 m
Mass density per area, ρ	2700 kg/m2
Young's modulus, E	7.11× 1010N/m2
Second moment of inertia, I	5.1924× 10-11m2
Poisson's ratio, υ	0.3

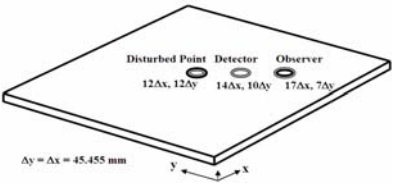
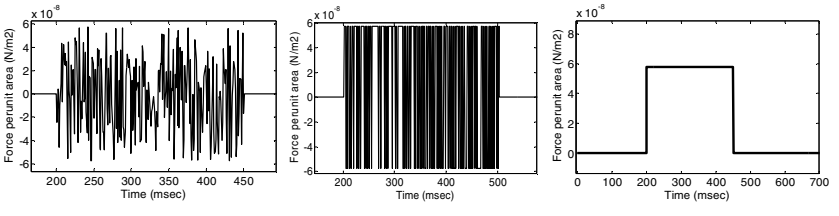


Fig. 3. Flexible Plate System

The framework development for the flexible plate simulation using Mat-lab/SIMULINK is shown in Fig.5. Meanwhile, the block named flexible plate is masked block diagram based on the block diagram of the flexible plate with state-space formulation as shown in the Fig.6.



(a) Random (b) PRBS (c) Finite Duration Step

Fig. 4. Disturbance Signals

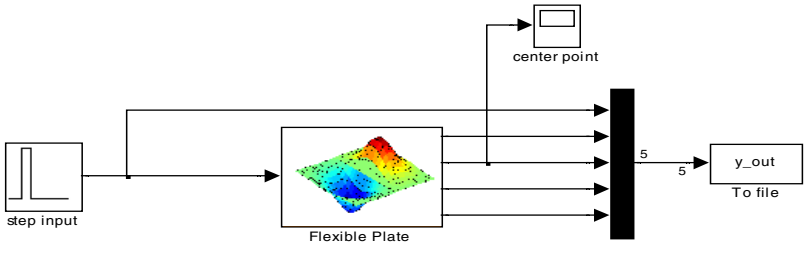


Fig. 5. Flexible plate simulation with step input

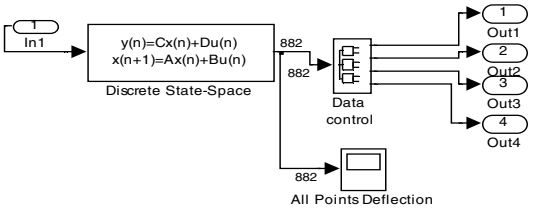


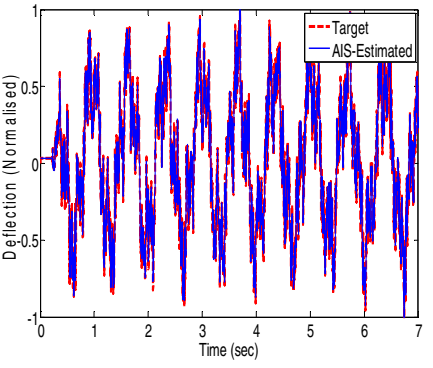
Fig. 6. Simulation block diagram using SIMULINK with state-space formulation

5 Results and Discussion

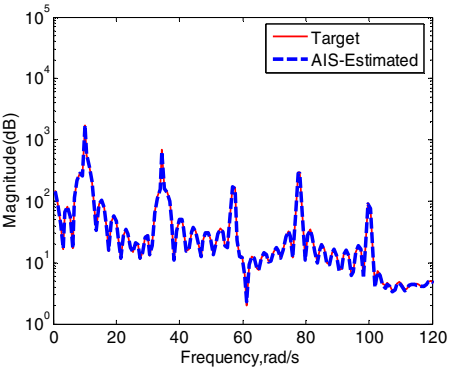
The output and the input of the plate system are measured from the observer point at (17Δx, 7Δy) and the detector point at (14Δx, 10Δy) respectively. Both the data vectors are then loaded into Matlab program to calculate predicted AIS output. Next, using the previous input, output, and randomly feasible variables with AIS antibodies, the predicted AIS output is determined as in Eq. (5). Therefore, the affinity measures can be generated using the difference between predicted AIS output and previous target output. AIS algorithm was thus tested with a fourth order model (8 variables) and subjected to random, PRBS and finite duration step disturbances. After 10 runs, the best solution of the models was recorded in terms of time histories, frequency domain and mean square of the errors. Simulations were performed over 7000 samples of data for each test set. In comparison to AIS, the execution of GA employed the same model order of four (4) with the following parameter: N_{ind}=100, generation gap 0.67, mutation rate 0.001, single point point crossover and fitness value similar to affinity measures of function.

The first five resonance frequencies of vibration of the plate found from spectral density of the predicted output of the AIS model were 9.971 rad/sec, 34.51 rad/sec, 56.76 rad/sec, 78.23 rad/sec and 99.71 rad/sec. In order to acquire a good model of the flexible plate, the predicted AIS output needs to fit the target output. Figure 7(a) shows time histories of predicted AIS model superimposed on the target output. This confirmed that the predicted AIS model was in agreement with the plant. Notably, power spectral density of predicted AIS characterized the plate behaviour in first five modes as shown in Figure 7(b). With minimum MSE of 0.0013829, the AIS predicted model shows good potential to be used in model-based control development in subsequent work. The corresponding correlation test results are shown in Figures 7(d)-7(f) using random signals for AIS, which are within the 95% confidence levels. Thus, this demonstrated the accuracy of the model in representing the actual plant.

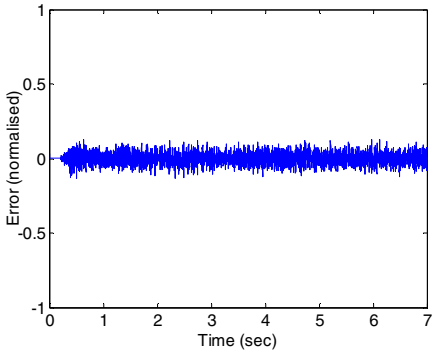
Figure 8 shows the changes of set of antibodies in affinity measures estimation when subjected to random disturbance. The process of searching the best affinity measures among antibodies set in CLONALG takes about 20 generations for the parameters a and b to steadily converge to optimal values. Meanwhile in AiNet-searching process may take more than 100 generations. The resulting transfer functions for all tested signals of best MSE are given as:



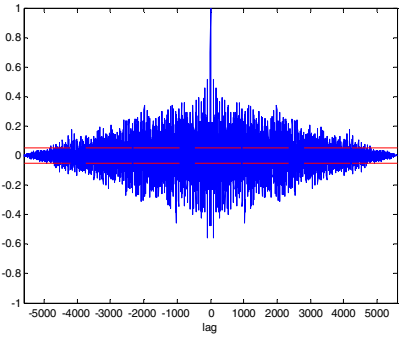
(a) Time histories of target-estimated output



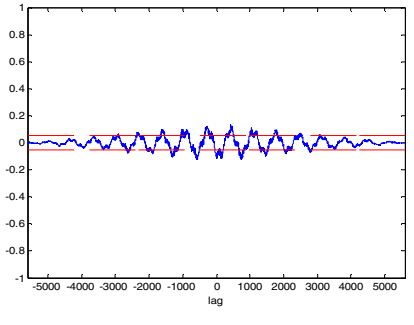
(b) Power spectral density



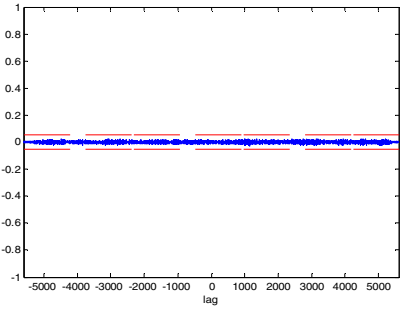
(c) Mean squared error



(d) Autocorrelation of residuals



(e) Cross-correlation of input-residual



(f) Cross-correlation of input square-residual

Fig. 7. Results of parametric flexible plate modeling with random disturbance signal

Transfer function for random signal :

$$F_r(z) = \frac{0.875z^3 - 0.1251z^2 + 0.2147z - 2.122x10^{-5}}{z^4 - 0.3205z^3 + 0.3233z^2 - 0.7031z + 0.7268}$$
$$f_r(s) = \frac{616.5s^3 + 1.411x10^6 s^2 + 2.077x10^9 s + 1.079x10^{12}}{s^4 + 319.1s^3 + 4.823x10^6 s^2 + 1.598x10^9 s + 1.819x10^{12}} \quad (7)$$

Transfer function for PRBS signal :

$$F_p(z) = \frac{0.7895z^3 + 0.9137z^2 - 0.9077z + 0.2323}{z^4 - 0.5742z^3 + 0.7623z^2 - 0.2113z - 0.001819}$$
$$f_p(s) = \frac{-4135s^4 - 2.926e007s^3 + 1.079x10^{10}s2 + 8.802x10^{13}s + 8.001x10^{16}}{s^5 + 1.109x10^4 s^4 + 4.982x10^7 s^3 + 7.583x10^{10}s^2 + 1.028x10^{14}s + 7.59x10^{16}} \quad (8)$$

Transfer function for finite duration step signal :

$$F_s(z) = \frac{0.8691z^3 + 0.2148z^2 - 0.5089z + 0.3906}{z^4 - 0.03335z^3 + 0.983z^2 - 0.1043z - 0.8124}$$
$$f_s(s) = \frac{760.8s^4 + 4.49x10^5 s^3 + 9.489x10^9 s^2 + 7.674x10^{12}s + 6.474x10^{15}}{s^5 + 561.4s^4 + 1.232x10^7 s^3 + 9.541x10^8 s^2 + 2.478x10^{13}s + 6.927x10^{15}} \quad (9)$$

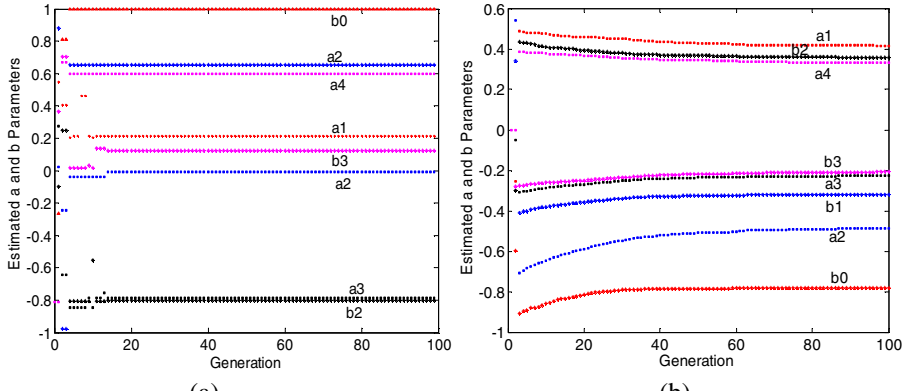


Fig. 8. Parameter changes of a's and b's for random disturbance signal using (a) CLONALG and (b) AiNet

All algorithms gave different mean squared error at each time run due to their stochastic search nature. The mean value and average time per generation for 10 runs is summarized in Table 2, where MSE values after mean squared error 100 generations of AIS, with 100 and 500 generations of binary-coded GA are also presented. It can be observed that, AiNet needed fewer generations compared to

Table 2. Normalised MSE and average time execution

Algorithm| Generations/ Disturbance	Normalised MSE (Mean)		
	Random	PRBS	Step
CLONALG100	$(4.296\pm1.8793)\times10^{-3}$	$(9.374\pm8.122)\times10^{-4}$	$(3.241\pm1.569)\times10^{-4}$
AiNet100	$(2.084\pm0.5633)\times10^{-3}$	$(5.68\pm1.75)\times10^{-4}$	$(4.40\pm1.78)\times10^{-5}$
BCGA100	$(2.780\pm0.758)\times10^{-3}$	$(6.285\pm2.385)\times10^{-4}$	$(2.419\pm1.067)\times10^{-4}$
BCGA500	$(2.079\pm0.767)\times10^{-3}$	$(5.438\pm2.218)\times10^{-4}$	$(3.527\pm1.725)\times10^{-4}$
Algorithm (100 Generations)	Average Time per generation (s)		
	Random	PRBS	Step
CLONALG	(14.997±0.313)	(15.26724±0.534)	(13.4455±0.488)
AiNet	(4.079±0.555)	(3.6863±0.466)	(3.5981±0.398)
BCGA	(4.686±0.148)	(4.75195±0.127)	(4.66328±0.118)

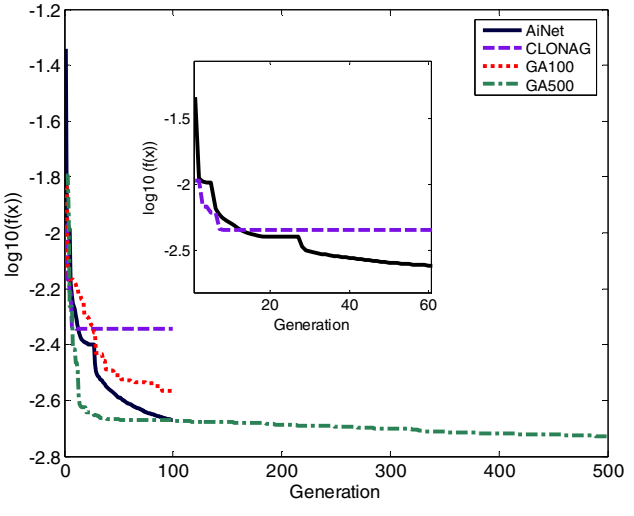


Fig. 9. Convergence of AIS vs GA for random disturbance

standard GA and CLONALG in estimating model of the system with all the disturbance signals used. For achieving similar value of MSE, GA required about 500 generations compared to 100 generation in AIS. It was noted that a larger number of generations in AIS model did not improve the convergence rate, but took more time to compute. Finite duration step input is the simplest signal that shows minimum error in representing the estimated model. In the case of average time per generation, AiNet can perform three times faster than CLONALG and at slightly similar

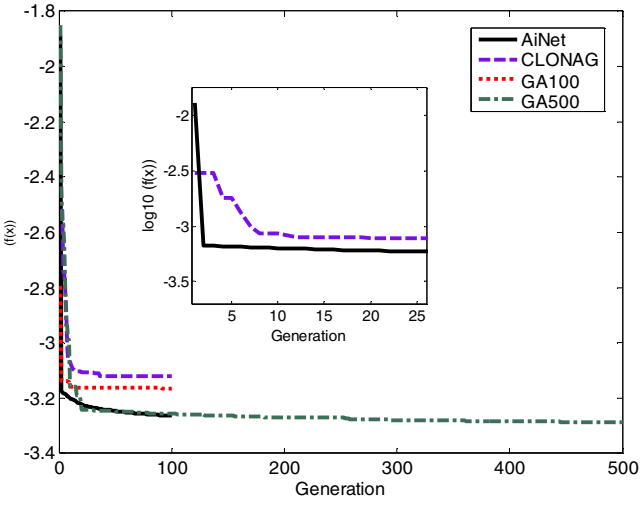


Fig. 10. Convergence of AIS vs. GA for PRBS disturbance

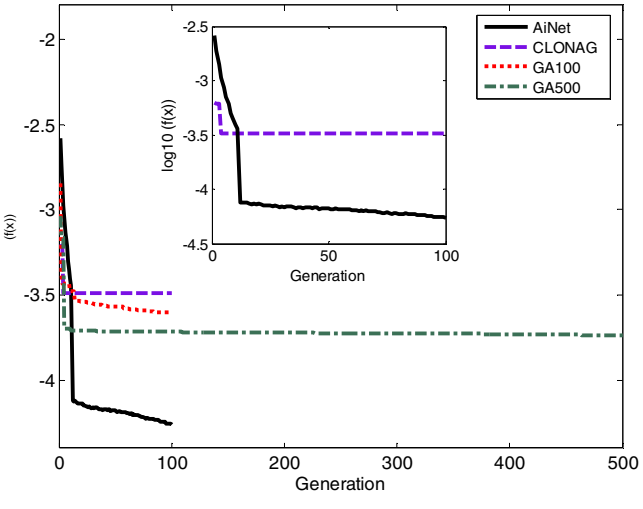


Fig. 11. Convergence of AIS vs. GA for step disturbance

rate compared to binary coded GA (BCGA). Nevertheless, AiNet is still capable to replicate thesystem better with minimum value of normalized MSE. The expected result is mainly due to real value operation in AiNet, where CLONALG and GA operations in coding and decoding features are more computationally complex.

The convergence curves for all tested signals are shown in Fig. 9 to Fig.11. The curves show that the AIS model has the ability to approximate solutions very fast and produce quite reasonable solutions. The optimal affinities for 100 generatons in terms

of transfer function in z- and s-domain presented in Eq.(7) to Eq.(9), described the best a and b parameters produced by AIS algorithm to characterize the plate model.

6 Conclusion

Parametric modelling of a flexible plate system has been carried out. Artificial immune system has been used for estimation of parameters of the model characterising the dynamic behaviour of a plate system. The approach has been evaluated in comparison to binary-coded GA with three different test signals. It is noted that the models obtained with AIS have performed satisfactory with reasonable solution compared to those obtained with binary-coded GA. This work has demonstrated that AIS has the potential to be applied in parametric modeling and subsequently in model-based control with reduced computational time.

Acknowledgements

S.M.Salleh is supported by the Ministry of Higher Education Malaysia and University Tun Hussein Onn Malaysia (UTHM).

References

1. Mohd Hashim, S.Z., Tokhi, M.O., Mat Darus, I.Z.: Active vibration control of flexible structures using genetic optimisation. Journal of low frequency noise, vibration and active control 3(25), 195–207 (2006)
2. Alam, M.S., Tokhi, M.O.: Modelling and vibration control of a twin rotor system: a particle swarm optimisation approach. In: 13th International Congress on Sound and Vibration, Vienna, Austria (2006)
3. Mat Darus, I.Z., Tokhi, M.O.: Parametric modeling of a flexible 2D structure. Journal of low frequency noise, vibration and active control 23(2), 115–131 (2004)
4. de Castro, L., Von Zuben, F.J.: The Clonal Selection Algorithm with Engineering Applications. In: Proceedings of GECCO 2000, Workshop on Artificial Immune Systems and Their Applications (2000)
5. Jui-Yu, W., Yun-Kung, C.: Artificial immune system for solving generalized geometric problems: a preliminary results. In: Proceedings of the 2005 conference on Genetic and evolutionary computation. ACM, Washington (2005)
6. Malim, M.R., Khader, A.T., Mustafa, A.: Artificial Immune Algorithms for University Timetabling. In: The 6th International Conference on the Practice and Theory of Automated Timetabling. Masaryk University, Brno (2006)
7. Timoshenko, S., Woinowsky-Krieger, S.: Theory of plates and shells. McGraw-Hill Book Company, New York (1959)
8. Salleh, S.M., Tokhi, M.O.: Discrete simulation of flexible plate structure using state-space formulation. Journal of System Simulation 20(19), 5141–5146 (2008)
9. Simon, M.G.: How Do We Evaluate Artificial Immune Systems? Evolution Compution 13(2), 145–177 (2005)
10. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, Heidelberg (2002)

11. Burnet, F.M.: The Clonal Selection Theory of Acquired Immunity. Cambridge University Press, Cambridge (1959)
12. de Castro, L.N., Von Zuben, F.J.: aiNet: An Artificial Immune Network for Data Analysis. In: Abbass, H.A., Sarker, R.A., Newton, C.S. (eds.) Data Mining: A Heuristic Approach. Idea Group Publishing, USA (2001)
13. de Castro, L.N., Timmis, J.: An artificial immune network for multimodal function optimization. In: Proceedings of Evolutionary Computation Congress, CEC 2002 (2002)

A Hybrid Approach for Learning Concept Hierarchy from Malay Text Using GAHC and Immune Network

Mohd Zakree Ahmad Nazri[1,2], Siti Mariyam Shamsuddin[2], Azuraliza Abu Bakar[1], and Salwani Abdullah[1]

[1] Data Mining & Optimization Research Group, Faculty of Information Science & Technology, Universiti Kebangsaan Malaysia, 43650 Bangi, Selangor, Malaysia
[2] Soft Computing Research Group, Faculty of Computer Science & Information System, Universiti Teknologi Malaysia, 81100, Skudai, Johor, Malaysia
{mzan,aab,salwani}@ftsm.ukm.my, mariyam@utm.my

Abstract. The human immune system provides inspiration in the attempt of solving the knowledge acquisition bottleneck in developing ontology for semantic web application. In this paper, we proposed an extension to the Guided Agglomerative Hierarchical Clustering (GAHC) method that uses an Artificial Immune Network (AIN) algorithm to improve the process of automatically building and expanding the concept hierarchy. A small collection of Malay text is used from three different domains which are IT, Biochemistry and Fiqh to test the effectiveness of the proposed approach and also by comparing it with GAHC. The proposed approach consists of three stages: pre-processing, concept hierarchy induction using GAHC and concept hierarchy learning using AIN. To validate our approach, the automatically learned concept hierarchy is compared to a reference ontology developed by human experts. Thus it can be concluded that the proposed approach has greater ability to be used in learning concept hierarchy.

Keywords: Artificial Immune System, Ontology Learning, Immune Network, Ontology Engineering, Machine Learning, Semantic Web.

1 Introduction

Ontology is one of the main components of Semantic Web. Ontology provides a common vocabulary for a specific domain of interest. It describes properties of a term or word so that machines, applications or services can effectively share information and knowledge, thus ensuring interoperability among machines. Alesso in [1] defined ontology as: Ontology = <taxonomy, inference rules> ; Taxonomy = <{classes},{relations}>. Taxonomy is also known as concept hierarchy. However, the traditional and labour-intensive ontology modelling, as described in [2-4] is a process that is tedious, complex, time consuming and expensive. Ontology engineers have been struggling with the *knowledge acquisition bottleneck* problem in developing the ontology. Thus, different learning approaches and methods from diverse spectrum of fields such as statistical analysis, machine learning and natural language

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 315–328, 2009.

processing have been proposed to semi or fully automatic construction of ontology. Cimiano [5] introduced an algorithm called guided agglomerative hierarchical clustering (GAHC) which combined linguistic analysis and clustering algorithm to construct concept hierarchy. His work produces better results compared to other unsupervised techniques such as Formal Concept Analysis or Bi-Section K-Mean. However, this approach has much room for improvement, and researches are still on-going to develop more effective ontology learning approaches. In order to improve GAHC, we propose the hybridized GAHC with idiotypic artificial immune network (AIN).

The approaches described in this paper are inspired from the vertebrate immune system in order to overcome some of the GAHC limitations. The aim of this paper is mainly to investigate whether there are distinct advantages to hybridized AIN with GAHC strategies to improve the effectiveness of GAHC in learning concept hierarchy from text. Moreover, the results of this study may inspire a solution or research plan in developing new techniques or methods for modelling ontologies automatically "*in a period as limited as possible with quality as high as possible* [6]".

The paper is structured as the following: Section 2 briefly discusses the idiotypic immune network theory. Section 3 discusses the problems in ontology learning and motivations for hybridization of GAHC with AIN. Section 4 outlines the implementation of our proposed hybrid approach and Section 5 presents the evaluations of the approach. Whereas the experimental results of the evaluation are discussed in Section 6. The conclusions are found in Section 7.

2 Idiotypic Immune Network

In this section, we briefly discuss the immune principles pertinent to the function of the proposed approach. For a comprehensive review of the biology and inspiration behind artificial immune systems, the reader is direct towards literature such as [7] and [8].

The purpose of the immune system is to protect the body against infection. This refers to a population of circulating white blood cells called lymphocytes comprise of B-cells, which grow in the bone marrow and T-cells, which mature in the thymus. The immune network algorithms (AIN) are based on the idiotypic immune network theory introduced by Jerne [11]. Jerne emphasizes the existence of idiotypic network, where antibodies can match other antibodies as well as antigens.In his theory, an antigen generates antibodies whose serologically unique structure, called an idiotype, results in the production of anti-idiotypic antibodies [12]. This means thatbinding not only exist between antibody and antigen, but also between antibody and antibody. Any node of the network can bind and be bound by any other node. This occurs due to the unique structure (idiotype) of the antigen-binding site of an antibody. An antibody (Ab), such Ab1 may stimulate the immune system to generate Ab2, which mimics the structure of the antigen. Eventually, a similar mechanism generates Ab3. This activation can continue to spread through the population and could result in the suppression of similar antibodies, thus encouraging diversity in the antibody population. The end result can be the production of a chain of antibodies that recognize each other.

The idiotypic network theory stresses that a specific immune response to foreign antigens is offered by the entire immune system in a collective manner although a single clone may play the dominant role in the whole process [9, 10]. Unlike idiotypic

immune network theory, clonal selection theory emphasizes that only cell that capable of recognizing an antigenic stimulus will proliferate, thus being selected against those that do not [7].

The features that are particularly relevant to our research are pattern recognition, diversity, distributed control, noise tolerant, learning and memory. This is discussed further in section 3.

3 Immune Network for Learning Concept Hierarchy

Ontology learning is defined by [13] as the methods and techniques used for constructing an ontology from scratch, or by enriching or adapting an existing ontology in a semiautomatic fashion using several sources. Much actually has been done towards answering this quest and a thorough overview on ontology learning research and comparison can be found in [3, 14-16].

The inspiration for the application of AIN in GAHC arises from the desirable features and properties of AIN as described by [17] and [18]. Features that are particularly relevant to our research are:

1. **Unsupervised pattern recognition:** The ability to recognize patterns of data without training examples is a common characteristic found in ontology engineering tools. It refers to the binding between antibodies and antigens or between antibodies itselves. In our work, pattern recognition is a process of data classification. The main goal is to detect relationships between data (i.e. concepts) and to determine how the data should be organized (e.g. create new cluster) without any prior knowledge about it.
2. **Diversity:** Diversity refers to the fact that, in order to achieve optimal antigen space coverage, antibody diversity must be encouraged [18]. The diversity principle is very important in solving our problem as Cimiano has stated in [3] that there are three main weaknesses in learning concept hierarchy from text: 1) the output of the Natural Language Processor tools can be erroneous, i.e. not all derived verb-argument dependencies are correct; 2) not all the derived dependencies will help to discriminate between different objects, and 3) the assumption of completeness of information will never be fulfilled. However, the immune system has the possibilities to solve these issues due to the antibody diversity coming from clone and mutation process which 'empower' AIN with the self-adaptive capability.
3. **Distributed control:** The immune system is governed by local interactions between cells and antibodies which means there is no need for a central controller.
4. **Noise tolerance:** The natural immune system is tolerant towards noise. An AIN has the potential to filter noisy data and uncover an underlying concept [17].
5. **Learning and Memory:** An immune system can learn the structure of pathogens and remember those structures. Thus, future responses to the same pathogens can be much faster.

Therefore, based on the metaphor of concepts and instances as *lymphocytes* that need to be 'classified' as either a hyponym or hypernym of a concept, this work investigates the potential use of AIN as a learning algorithm to improve GAHC performance. The literature describes a vast array of systems for learning concept hierarchy.

Most of the existing studies focus either on unsupervised machine learning techniques as stated in [3, 16] or rely on linguistic analysis as reported in [4,19]. However, OL approaches that are based on unsupervised learning paradigms share two major problems. The first problem is data sparseness and the second one is the inability to label the produced cluster appropriately [5]. A particularly interesting application is proposed by Cimiano [5], who introduced an algorithm called guided agglomerative hierarchical clustering (GAHC). What makes GAHC interesting to be explored and enhanced is it produces better results than other hierarchical clustering techniques [3, 5]. However, there are some limitations in GAHC. First is the problem of "low counts" (i.e. linguistic patterns that rarely found) [19]. Most of the taxonomic relations do not appear in the suggested linguistic patterns. Therefore, a sufficiently enormous corpus is needed in order to find a significant number of matches (i.e. linguistic pattern), but large annotated Malay corpuses are scarce and unavailable. Second, unclassified terms still have to rely on the distributional similarity of terms to locate its best position in the hierarchy. If a term still could not be classified by the algorithm, GAHC halted the process by classifying the term directly under the root. We believe that immune system features can improve GAHC performance.

A literature search revealed that the areas of ontology learning from text using immune approaches are unexplored. To date, no study has been conducted on ontology learning from Malay text in the literature. Thus, Malay text is used in the study to pioneering research on ontology learning from Malay text. Malay language shares the same Subject-Verb-Object sentence structure as English. However, Nazri et al [20] has shown that syntactic features which have been effectively used on English text for inducing concept hierarchy are not effective on Malay text.

4 The GCAIN System

The proposed hybridization approach employed here is called GCAIN (Guided Clustering and Artificial Immune Network). The overall process is divided into 3 steps which are pre-processing, GAHC and AIN. An algorithm flow is shown in Figure 1.

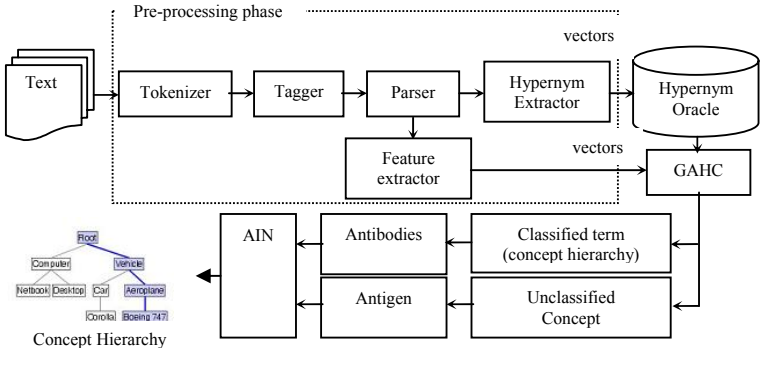


Fig. 1. The overall process of GCAIN

4.1 Pre-processing, Hypernym Oracle and Feature Extraction

The process starts with tokenizing the text. The purpose of tokenization step is to break up a text into its constituent tokens. The constituent tokens are then tagged with part-of-speech (POS). Each token is marked or labelled according to a particular part of speech, e.g. noun or verb. Next, the POS tagged texts are parsed. A parser is an application that analyse a sequence of tokens to determine grammatical structure with respect to the given Malay grammatical rules.

Since the GAHC algorithm relies on an oracle returning possible hypernyms for a given term, the oracle needs to be constructed before applying the GAHC algorithm. To construct the Malay Hypernym Oracle, we used Malay translated Hearst linguistic patterns to find hypernyms in the text. The Hearst's pattern have been used by Cimiano in [5] and [20] to identify is_a relationships between the concepts referred by two terms in the text. We adopted Cimiano's technique in formulating the patterns using the variable '<INSTANCE>' to refer to a candidate noun phrase, as the name of an ontology instance, and '<CONCEPT>' to refer to the name of a concept [19]. For example :

Hearst Pattern 1: <CONCEPT>s such as <INSTANCE>
Hearst Pattern 2: such <CONCEPT>s as <INSTANCE>

The above patterns match the following expressions: *President such as Obama* or *such university as UKM*. GAHC will recognize that *Obama* and *President* and *university* and *UKM* has a taxonomic relationship as such *Obama is_a President* and *UKM is_a university*. Thus, *President* is a hypernym of an instance called *Obama* while *UKM* is a hyponym of a concept named *university*.

The hypernyms are stemmed before storing it in the Hypernym Oracle (HO). The quality of the learned concept hierarchy also depends on the features extracted from the given corpus as proved by [21]. Syntactic features are extracted from the parsed sentences. In this study, we used the same syntactic features used by Cimiano in [3]. However, as reported by [21], the available Malay NLP tools produce erroneous outputs. Therefore, syntactically parsed text is subject to human inspection and syntactic features had to be extracted manually to ensure the extracted features are acceptable.

Before GCAIN can be applied to induced concept hierarchy, the extracted terms must be represented as a group of feature vectors. Bits in the vector represent the presence of features which are the corresponding verb to a noun in a sentence. The features are extracted between the verbs appearing in a sentence and the heads of the subject, object and prepositional phrase-complements. For each noun appearing as head of the argument positions, the corresponding verbs are used as attributes that results in a pair of object-attribute [3]. Each noun is a candidate for concept or instance. Therefore, each noun is a feature vector of which each feature (i.e. verb) is binary representing whether or not the corresponding verb appears along the noun in a sentence. Thus, 1 represents presence and 0 absences of the feature.

After the features have been extracted, i.e. a set of object (noun) and its attributes, each word in the set is stemmed, i.e. assigned to their base form. The feature vector is a high dimensional space. All terms are the sparse point in the high dimensional space. The more features extracted, the sparser the space is.

4.2 Concept Hierarchy Induction Using GAHC

Cimiano [5] proposed a novel GAHC algorithm for the automatic induction of concept hierarchies from text by exploiting the hypernyms stored in database called hypernym oracle (HO) to guide the clustering process. We redevelop GAHC based on algorithm presented in [5] with two changes made to the algorithm in order to meet AIN requirement. The changes are briefly summarised below:

1. The original GAHC does not filter the extracted features. In our approach, before applying the GAHC, we determine the similarities of each terms within the list of extracted features ($O(n^2)$ by using the cosine distance measure and only pairs with higher degree of similarities than the pre-determined threshold t_g will be clustered by GAHC. Pairs of terms with lower similarities than t_g are clustered in set C, i.e. C:= {};
2. The step in the original GAHC where unclassified terms are automatically added as sub-concepts to the most frequent hypernym in HO(t) is removed. In our algorithm, unclassified terms are clustered in set C, i.e. $C := C \cup (t_1,t_2)$.
3. The Hypernym Oracle (HO) is constructed by looking for hypernym matched with a Malay translated Hearst pattern from the text only while Cimiano used three sources, i.e. Wordnet, Google and the corpus.

The generic pseudo code for GAHC algorithm is shown in Fig. 2. A detailed and thorough discussion on GAHC algorithm and its classification rules can be found in [5] and [3].

```
Input: HO(t), list T and set C
FOREACH pair (t1,t2) in the ordered list T
        Search GAHC classification rules
        IF matched
                Insert new hypernym into HO(t)
        ELSE
                Mark t1 and t2 as clustered, i.e. C := C ∪ (t1,t2)
ENDFOR
Output: HO(t) and unclassified terms in C.
```

Fig. 2. GAHC algorithm adapted from Cimiano's work in [3]

The output of GAHC is a pair of hypernyms (i.e. $is\_a(t_1,t_2) \in HO(t)$) and unclassified terms in set *C*. Both HO and set C will be fed into AINs.

4.3 AIN Algorithm in GCAIN

In this phase, AIN act as a classifier in order to classify terms (i.e. an instance or concept) which GAHC has failed to classify as either hypernym or hyponym of a term. A diagrammatic depiction of the algorithm flow is shown in Fig. 3. We refer B-cells to antibodies (i.e. classified terms), which mean the concept hierarchy is composed of a

number of antibodies. AIN evolves a population of antibodies based on the immune network theory, clonal selection and affinity maturation.

The procedure of evolving antibodies (Ab) can be explained as follows. **(1)** *Antibodies Generation:* GAHC initializes a set of Abs and put them into an empty memory matrix called M. The antibodies are not created or copied from the antigens as usual. The antibodies are considered correctly classified concept/instance in three different steps as follows:

1. Ab type 1: antibodies found during the construction of the hypernym oracle;
2. Ab type 2: antibodies created through GAHC hierarchical clustering technique;
3. Ab type 3: antibodies created by learning process using AIN.

The antigens (pathogens) are formed in two ways; First, terms that GAHC has failed to classify as a hypernyms or hyponyms into the established concept hierarchy, and terms which have lower degree of similarities compared to other terms. While the termination condition is not true, follow the steps below:

(2) Affinity calculation: Calculate the affinity between the current Ag and each Ab from M. **(3) Clonal Selection:** Select the *n* highest affinity antibodies (Abs) from the concept hierarchy. For each of these highest affinity Abs, if an Ab's affinity > threshold $\sigma_{d,}$, clone the Ab. The clone size is proportional to the affinities of Abs. That is, the higher the affinity is, the more Abs are cloned. **(4) Affinity maturation:** Mutate each cloned Ab and increase the fitness of those Abs. Each clone undergoes a mutation inversely proportional to its affinity and the fitness of the Abs will be increased through a process call *replication*. **(5) Reselection:** Calculate the affinity between each cloned Abs and the current Ag. Reselect Abs with highest affinity and classified the Ag as hyponym of the Abs. Insert the resulting Abs into M. The Ag would eventually recognized as Ab (type 3).

If none of the highest affinity Abs could bind the Ag (i.e. classified as hypernyms), the Ag will be classified as sub-class of the root/top concept and recognized as Ab (type 3). **(6) Repeat (2) – (5) for each Ag** in the population. **(7) Clonal suppression:** Calculate affinity between the antibodies (type 3 only) within each cluster and do suppression. This process will remove redundant Abs with affinity < threshold σ_s. **(8) Network suppression:** Calculate affinity between clusters and do suppression. Remove Ab type 3 in the concept hierarchy whose fitness $< \sigma_f$.

GCAIN will not be effected in case the pre-processing phase and GAHC failed to induce any concept hierarchy (i.e. antibody type 1 and 2) since there will always be one seed which is the 'root' or top concept in the matrix M. All antigens in the population will be presented to the 'root' one by one as depicted in Figure 3. This enables GCAIN to automatically detect new hierarchical clusters under the 'root'.

In our model, we define two types of suppressions: clonal suppression and network suppression (i.e., concept hierarchy compaction). The clonal suppression occurs when the affinity between Ab-Ab within a cluster is greater than a given threshold σ_s. Such suppression will cause antibodies with low capability in the competition of recognising antigens to be deleted from the hierarchy. The suppression would also cause the elimination of similar immune cells, which is important in maintaining the diversity

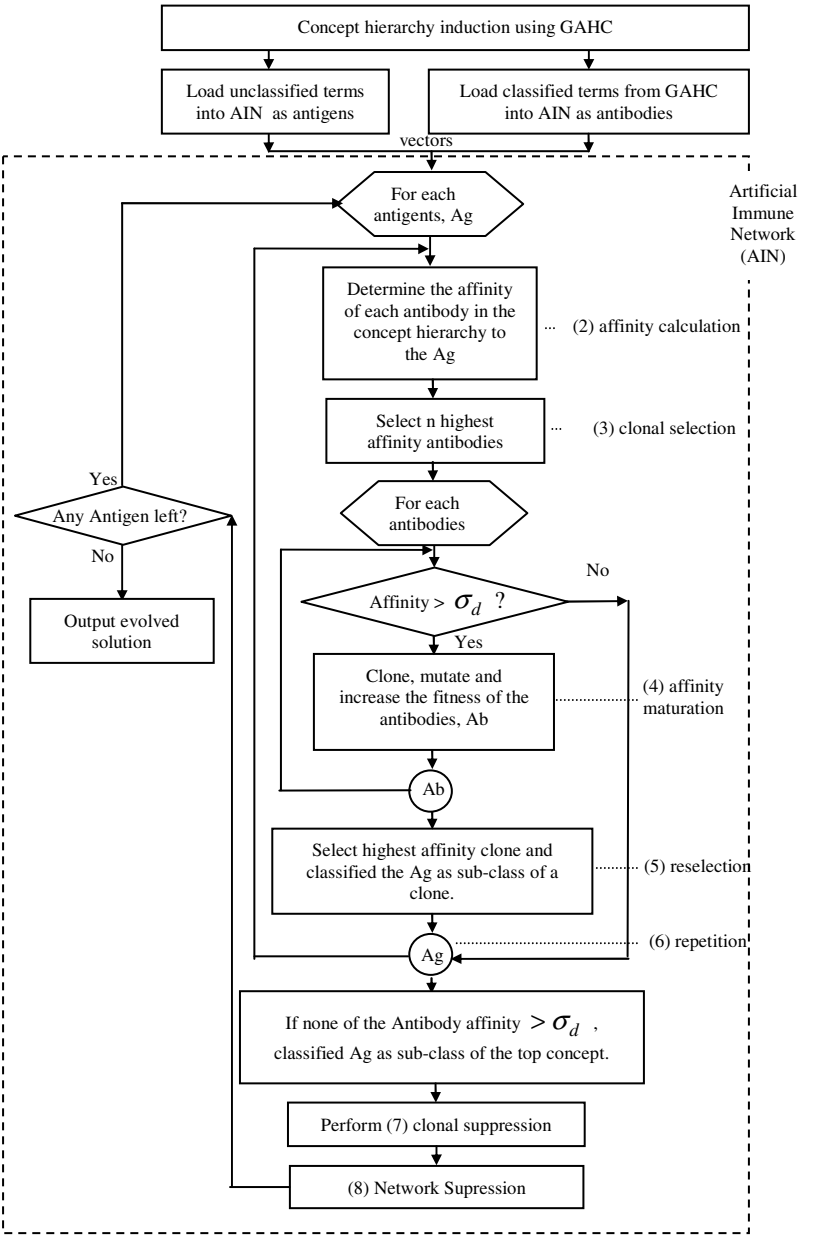


Fig. 3. Overview of the AIN algorithm

of antibody pool. However, only the antibody type 3 is suppressed in CGAIN. Type 1and Type 2 antibodies are established by previous technique, thus these antibodies will not be removed in any such way.

The distance metric presented in [9] is used to measure the affinity between antibodies and antigens as described in Eq. (1). In this work, an antigen is a term represented as a feature vector. The affinity between an antigen and an antibody is the ratio of the number of features (i.e verb) that an antigen has and can be recognised by an antibody to the total number of features extracted from the text. The affinity between two antibodies is defined as the reversal of their similarity.

$$\text{Affinity(Ab,Ag)}=\sum_{i=1}^{n}\phi(\text{Ab}_i, \text{Ag}_i)\div|\text{Ag}| \quad (1)$$

where

$$\phi(\text{Ab}_i, \text{Ag}_i)= 1, \text{ if Ab = Ag = 1 OR 0, otherwise} \quad (2)$$

The network suppression occurs when the affinity between two clusters of concept hierarchy is inferior to the specified threshold and such suppression will result in a merging of two cluster of the concept hierarchy. The affinity between two clusters is defined by [9] as the affinity between the two highest affinity antibodies in the two clusters.

If affinity of Ab-Ag is greater than the affinity threshold σ_d, the antibody will reproduce and mutate itself. Number of clones to be produced (i.e., *nc*) and number of genes need to be 'mutated' (i.e., *M*) are determined by two formula as shown in (3) and (4) respectively [9]. η and ω is a coefficient determined by users.

$$nc = round(\eta \times Ab.affinity) \quad (3)$$

$$M = round(\omega \times Ab.affinity) \quad (4)$$

The learning process in GCAIN may repeat a number of specified iterations. The termination condition for the *while* loop can be determined by specifying a constant or a number of iteration. However, in this explorative research, the number of iteration (*n*) is limited to 1. In our observation, if $n > 1$, the suppression process will fail to remove redundant concepts or repetition which is not allowed in a concept hierarchy. Gomez et al in [21] stated that ***redundancies of Subclass-Of relations*** occur between classes that have more than one SubClass-Of (i.e., $is\_a(t_1,t_2)$ relation).

5 Evaluation Measures

In this experiment, the performance of GCAIN is evaluated by means of comparing the automatically produced taxonomy against a handcrafted "reference taxonomy". The evaluation is conducted based on Cimiano's evaluation method in [5]. In the context of this study, we evaluated the concept hierarchy produced by GCAIN on three reference ontologies which are information technology (IT), Biochemistry and Fiqh (i.e., Islamic Jurisprudence) ontology. The reference ontologies were developed by a graduate research assistant and assisted by three experts in the domain. They were asked to construct the ontology based on concepts or terms that they could find in the text. In order to formally define our evaluation measures, the *core ontology* has been introduced by [3] as the following:

> *A core ontology is a structure* $O := (C, \leq c)$ *consisting of (i) a set C of concept identifiers and (ii) partial order* $\leq c$ *on C called concept hierarchy of taxonomy.*

In this section, we briefly present measures to compare the lexical and taxonomic overlap (TO) between two ontologies. For comprehensive discussion of the measures presented in this paper, the reader will be directed towards [3]. The lexical recall (Eq. 5) and lexical precision (Eq. 6) of two ontologies O1 and O2 is measured as follows:

$$LR(C_1, C_2) = | C_1 \cap C_2) \div | C_2 | \tag{5}$$

$$LP(C_1, C_2) = | C_1 \cap C_2) \div | C_1 | \tag{6}$$

In order to compare the taxonomies of two ontologies, we use the common semantic cotopy (SC) introduced by Cimiano in [3] as Eq.(7):

$$SC''(c, O_1, O_2) := \{c_j \in C_1 \cap C_2 \mid c_j c_i \vee c_i c_j\} \tag{7}$$

We calculate the Precision of Taxonomic Overlap (P_{TO}) as Eq (8) Recall of Taxonomic Overlap (R_{TO}) as Eq. (9) between two ontologies as follows [3]:

$$P_{TO}(O_1, O_2) = \overline{TO'}(O_1, O_2) \tag{8}$$

$$R_{TO}(O_1, O_2) = \overline{TO'}(O_2, O_1) \tag{9}$$

We also calculate the harmonic mean of both P_{TO} and R_{TO} as Eq. (10):

$$F_{TO}(O_1, O_2) = (2.P_{TO}(O_1, O_2).R_{TO}(O_1, O_2) \div P_{TO}(O_1, O_2) + R_{TO}(O_1, O_2)) \tag{10}$$

where

$$\overline{TO'}(O_1, O_2) = 1 \div | C_2, C_1 | \sum_{C \in C_1 / C_2} \max_{c' \in C_2 \cup \{root\}} TO''(c, c', O_1, O_2) \tag{11}$$

and equation (12) **:**

$$TO(c, c', O_1, O_2) = | SC''(c, O_1, O_2) \cap SC''(c', O_2, O_1) | \div | SC''(c, O_1, O_2) \cup SC''(c', O_2, O_1) |$$

6 Experiments and Results

6.1 Problem Instances

The proposed algorithm was conducted on a relatively small collection of text together with a Malay-language ontology which reflects a given corpus. A collection of documents for this experiment was obtained from the Center for Artificial Intelligence Technology (CAIT) of Universiti Kebangsaan Malaysia (UKM). The Malay

developed corpus, contains documents from three different domains i.e. Information Technology, Biochemistry and Fiqh (i.e., Islamic Jurisprudence).

The Malay corpus are initially compiled from text books written by researchers at CAIT. The corpus consists of about 90,000 tokens. The parameters used in the algorithm are chosen after preliminary experiments as shown in Table 1.

Table 1. Parameters used in GCAIN

		Information Technology	Bio-Chemistry	Fiqh
Number of terms in set C	:A	265	748	114
Length of attributes (in vector)		137	211	162
Similarity threshold for (t1,t2)*	: t_g	0.01	0.6	0.4
Affinity threshold for Ab-Ag	: σ_d	0.15	0.15	0.15
Affinity threshold for Ab-Ab	: σ_s	0.1	0.1	0.1
Affinity threshold for Cluster-Cluster	: σ_f	0.3	0.3	0.3
Clone coefficient	: η	10	10	10
Mutation coefficient	: ω	10	10	10
Iterations for while loop	: n	1	1	1

*Note that (t1,t2) is not yet treated as an antigen or antibodies

As shown in Table 1, population *A* is the number of terms in set C which has been excluded from being processed by GAHC and also terms that have not 'triggered' the GAHC rules to be classified as a sub-concept or super-concept. During parameter setting, the authors have to check for redundancy before measuring its taxonomy overlaps. Thus, the parameter values in Table 1 are chosen to avoid any potentially repeated concept names.

6.2 Comparison Results

The aim of the experiment carried out here is to support our hypothesis that the AIN algorithm can further improve the effectiveness of GAHC in learning concept hierarchy. Thus, we conducted experiments on three Malay texts and compare the performance of GCAIN against GAHC algorithm in hierarchical clustering. Table 2 shows the results of comparing the concept hierarchy produced by GCAIN and GAHC with the reference ontology in terms of the taxonomic overlap measures.

Table 2. Comparison results between GAHC and GCAIN

Text	Information Technology		Biochemistry		Fiqh	
	GAHC	GCAIN	GAHC	GCAIN	GAHC	GCAIN
Precision_{TO}	22.15%	36.89%	19.27%	17.88%	16.87%	18.06%
Recall_{TO}	23.72%	28.32%	29.39%	33.21%	33.12%	33.91%
F_{TO}	22.91%	32.04%	22.28%	23.25%	22.36%	23.57%
TO Differences	9.13%		0.98%		1.21%	

Based on the results, the proposed GCAIN increases F_{TO} from 22.91% to 32.04% over the Information Technology documents. The obtained results show significant improvement in F_{TO} for IT text. GCAIN performance on IT text is 9.13% higher than GAHC. However, GCAIN's performances over Biochemistry and Fiqh texts are comparable to GAHC as the GCAIN's produced F_{TO} over both texts are slightly higher than GAHC (i.e., 0.98% and 1.21% higher).

The results in Table 2 show that both GAHC and GCAIN performance over Biochemistry and Fiqh corpora is significantly lower than their performance over IT corpora. Table 1 show that the length of features extracted from Biochemistry and Fiqh natural text are larger than IT. Thus, we believe that the main reason for this result is that features extracted from Biochemistry and Fiqh are not enough o represents the terms extracted from the text. The feature selection method which used corresponding verbs of a noun seen in a sentence leads to data sparseness due to the large number of different verbs (as features) encountered in natural language texts.

Based on our experimental results, it is found that the performance of GCAIN over the three texts is much affected by t_g rather than the three affinity thresholds: σ_d , σ_s and σ_f . As an example, table 3 show the result of using different parameter t_g on GCAIN's F_{TO} over Fiqh text.

Table 3. The effect of parameter t_g on GCAIN's F_{TO} over Fiqh text

	0.01	0.1	0.15	0.4	0.5	0.6
F_{TO}	18.77%	19.45%	20.22%	21.86%	22.66%	23.57%

Further evaluations are carried out to measure the accuracy of the lexical precision, lexical recall and lexical F-measure metrics as in Eq. (4) and Eq. (5). Note that, lexical precision is defined as a measure of the proportion of concepts or instances that the learning algorithm got right no matter where a concept may located in the concept hierarchy; recall stands for the number of correct terms per number of terms in the reference ontology; and F-measure is the combination of these two previous metrics as a single measure of overall performance.

Table 4 shows that GCAIN yields slightly better result against the GAHC. GCAIN generates 1.86% higher over IT corpora, 1.58% higher over Biochemistry corpora and 0.95% higher over Fiqh corpora. Thus it can be concluded that the proposed approach performance is comparable or slightly better than GAHC in learning concept hierarchy that we tested for Lexical F-Measure metric.

Table 4. Comparison Results between GAHC and GCAIN on the lexical

Corpus	Information Technology		Biochemistry		Fiqh	
	GAHC	GCAIN	GAHC	GCAIN	GAHC	GCAIN
Lex.Recall	65.15%	69.28%	50.39%	50.95%	41.07%	42.86%
Lex.Precision	61.00%	61.09%	42.49%	44.80%	57.82%	57.14%
Lex.F-Measure	63.07%	64.93%	46.10%	47.68%	48.03%	48.98%
F-Measure Diffrences	1.86%		1.58%		0.95%	

7 Conclusions

In this article, we proposed a hybrid approach that combines guided agglomerative hierarchical clustering (GAHC) and artificial immune network (AIN) for learning concept hierarchy from Malay texts. An ontology learning system has been built based on GAHC and GCAIN and tested on three different corpuses domains which are IT, Biochemistry and Fiqh (i.e, Islamic Jurisprudence). Results have shown that GCAIN perform better than GAHC in all three domains. Among the three domains, GCAIN perform significantly better in IT but shows comparable results to GAHC in Fiqh and Biochemistry. The performance of GCAIN is much affected by t_g instead of the three affinity thresholds: σ_d , σ_s and σ_f . Our future work will incorporate concept hierarchy induction technique and approach in order to compare GCAIN performance against different learning paradigms. This work is an exploration of AIN application in new research domain such as ontology engineering. The experiment results indicate a modest improvement on the GAHC by applying AIN to classify the antigens (i.e. unclassified terms). However further study and larger trials on huge corpuses are much needed in order to examine and confirm the efficacy of AIN in the domain of ontology learning from text.

References

1. Alesso, H.P., Smith, C.F.: Developing Semantic Web Services. AK Peters, Natick (2005)
2. Gulla, J.A., Brasethvik, T.: A Hybrid Approach to Onology Relationship Learning. In: Kapetanios, E., Sugumaran, V., Spiliopoulou, M. (eds.) NLDB 2008. LNCS, vol. 5039, pp. 79–90. Springer, Heidelberg (2008)
3. Cimiano, P.: Ontology Learning and population from Text. Springer, Berlin (2006)
4. Hearst, M.: Automatic Acquisition of Hyponyms from large Text Corpora. In: Proc. of 14th COLING, Nantes, France (1992)
5. Cimiano, P., Staab, S.: Learning Concept Hierarchies from Text with a Guided Agglomerative Clustering Algorithm. In: International Conference on Machine Learning 2005 (ICML 2005), Bonn, Germany (2005)
6. Reinberger, M.-L., Spyns, P.: Unsupervised Text Mining for the learning of DOGMA-inspired Ontologies. In: Buitelaar, P., Cimiano, P., Magnini, B. (eds.) Ontology Learning from Text: Methods, Applications and Evaluation, pp. 29–42. IOS Press, Amsterdam (2005)
7. Timmis, J., Andrews, P.: A Beginners Guide to Artificial Immune Systems. In: Flower, D., Timmis, J. (eds.) Book In Silico Immunology, vol. I, pp. 47–62. Springer, US (2007)
8. Castro, L.N.d., Timmis, J.I.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, London (2002)
9. Hang, X., Dai, H.: An Immune Network Approach for Web Document Clustering. In: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence. IEEE Computer Society, Los Alamitos (2004)
10. Chowdhury, D.: Immune Network: An Example of Complex Adaptive Systems (1998)
11. Jerne, N.K.: Towards a network theory of the immune system. Annals of Immunology (Inst Past) 125, 373–389 (1974)
12. Shoenfeld, Y.: The idiotypic network in autoimmunity: antibodies that bind antibodies that bind antibodies. Nature Medicine 10, 17–18 (2004)

13. Stojanovic, L., Stojanovic, N., Ma, J.: An Approach for Combining Ontology Learning and Semantic Tagging in the Ontology Development Process: eGovernment Use Case. In: Benatallah, B., Casati, F., Georgakopoulos, D., Bartolini, C., Sadiq, W., Godart, C. (eds.) WISE 2007. LNCS, vol. 4831, pp. 249–260. Springer, Heidelberg (2007)
14. Gómez-Pérez, A., Manzano-Macho, D., Alfonseca, E., Núñez, R., Blacoe, I., Staab, S., Corcho, O., Ding, Y., Paralic, J., Troncy, R.: Deliverable 1.5: A survey of ontology learning methods and techniques. In: Gómez-Pérez, A., Manzano-Macho, D. (eds.) Next Web Generation, Leopold Franzens University of Innsbruck, Insitute of Computer Science, Innsbruck (2003)
15. Biemann, C.: Ontology Learning from Text - a Survey of Methods. Journal for Computational Linguistics and Language Technology 20, 75–93 (2005)
16. Gomez-Perez, A., Manzano-Macho, D.: An overview of methods and tools for ontology learning from texts. The Knowledge Engineering Review 19, 187–212 (2004)
17. Secker, A., Freitas, A.A., Timmis, J.: AISEC: an artificial immune system for e-mail classification. IEEE Congress on Evolutionary Computation, 131–138 (2003)
18. Cayzer, S., Aickelin, U.: A Recommender System based on Idiotypic Artificial Immune Networks. Journal of Mathematical Modelling and Algorithm 4, 181–198 (2005)
19. Cimiano, P., Handschuh, S., Staab, S.: Towards the Self Annotating Web. In: 13th International World Wide Web Conference (WWW 2004). ACM, New York (2004)
20. Nazri, M.Z.A., Shamsudin, S.M., Abu Bakar, A., Abd Ghani, T.: Using linguistic patterns in FCA-based approach for automatic acquisition of taxonomies from Malay text. In: International Symposium on Information Technology. ITSim 2008, vol. 2, pp. 1–7 (2008)
21. Gomez-Perez, A., Corcho-Garcia, O., Fernandez-Lopez, M.: Ontological Engineering. Springer, New York (2005)

An Immune Inspired Algorithm for Solving Dynamic Vehicle Dispatching Problem in a Port Container Terminal

N.M.Y. Lee, H.Y.K. Lau, and A.W.Y. Ko

Department of Industrial and Manufacturing Systems Engineering,
The Univerisity of Hong Kong, Pokfulam Road, Hong Kong
myleenicole@graduate.hku.hk, hyklau@hkucc.hku.hk,
aux1496@gmail.com

Abstract. A typical Vehicle Dispatching Problem (VDP) for a port container terminal often involves offline resource allocation and is often successfully solved by heuristics algorithms. In this research, an autonomous and decentralized vehicle dispatching algorithm is proposed in which the algorithm is inspired by the human immune system. Specifically, the proposed algorithm is inspired by the cell-mediate immune response of T-cells that possess the capability of exploring the environment and providing an adaptive and specific immune response to the invading antigens. We conduct extensive simulation studies to study the performance of the algorithm in solving a typical vehicle dispatch problem derived from realistic terminal configurations and operational constraints. The results show good vehicle utilization and low computational cost when comparing with a GA-based algorithm.

Keywords: Cell-mediated Immune Response, Stimulation and Suppression Model, Vehicle Dispatching Problem.

1 Introduction

In the biological immune system, immune molecules and immune cells work cooperatively to eliminate the invading pathogens. Immune mechanisms such as the Clonal Selection Principle and Immune Network Theory that provide profound functions to characterize the immune system through a series of procedures including immune cells recognizing the antigens, regulating the capacity of the immune cells and controlling the immune response. By adopting these mechanisms, a number of engineering applications have been solved, more significantly, in the fields of optimization, anomaly detection and clustering [1] [2]. Amongst these studies, there are few addressing dynamic vehicle planning and scheduling. On the other hand, VDPs have been resolved by classical evolutionary computing and heuristics approaches, nevertheless they are ineffective in handling communication between the involved entities and managing the uncertainties concurrently at a dynamical changing environment. In view of this, our research proposed an immune inspired algorithm for solving general vehicle dispatch problems at a typical seaport container terminal.

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 329–342, 2009.

In the context of a port container terminal, Vehicle Dispatching Problem (VDP) is a class of resource allocation problem by which a fleet of vehicles is allocated to handle the tasks of loading and discharging of containers onto/from a vessel at both quay side and the yard. In general, the performance of such operation is quantified by the dwell time, which is the time required for a vehicle to complete the handling a container. For typical terminal operation, containers are handled by various terminal equipments and transportation devices such as quay cranes, yard cranes and tractors that are coordinated and collaborated synchronously and interactively. Conventionally, job allocation and dispatching of tractors in a container terminal is handled manually by operators. We proposed an immune inspired algorithm for dealing with this problem in a decentralized and autonomous manner. While the proposed algorithm inherits the adaptive, distributive and cooperative properties of the human immune system, cell-mediated immunity performed by T-cells is specifically adopted by the proposed algorithm. Our study aims at minimizing the dwell time of vehicles in a simulation studies that is developed based on real terminal configurations and data logged for the actual terminal operations. More importantly, each vehicle is capable to provide an immediate response to uncertain requests and cooperate with other vehicles, which is in parallel to the behavior of the immune cells.

Following a review of related work, Section 3 discusses the theoretical underpinning of the algorithm. A detailed explanation of the immune functions including affinity measure, stimulation and suppression functions are given in Section 4. In Section 5, a Genetic Algorithm (GA) approach is taken as a control to compare the performance of the proposed algorithm in solving a real vehicle dispatch problem in a port container terminal in terms of the total dwell time of vehicles, the utilization of the vehicles, and the computation cost of the algorithm. We conclude the study in Section 6.

2 Related Works

In the literature, there is little describing immune inspired frameworks or algorithms developed for solving vehicle dispatching or routing problems in contrast to the traditional rule-based [3], heuristics approaches [4] [5] and evolutionary computing [6]. In the work of Gendreau and his colleagues [6], tabu search was deployed for solving dynamic VDPs with pickup and deliveries, yet in the study, the vehicles were incapable of communicating with other vehicles and handling new requests in the vicinity of a vehicle current location. In contrast, the algorithm proposed in our study is novel in using cell-mediated immune response to solve VDPs in a typical container terminal and attempt to overcome the problems raised by Gendreau et al.

With regard to computing techniques, Potvin [7] conducted a review on bio-inspired vehicle routing algorithm and found that only a small number of problems were solved using Artificial Immune Systems (AIS) inspired algorithms. The performance of evolutionary computing and meta-heuristics approaches is often claimed to outperform other computation methods with respects to the quality and convergence of solutions, and the required computational complexity. Each of these algorithms exhibits its strength and weaknesses in various domains, e.g., studies of finding near optimal solutions using Ant Colony Optimization (ACO) and GA [8]. Alternatively, ACO was claimed to outperform GA by Gambardella et al. [9]. Nonetheless,

GA often consumes less computation resources [10] and provides better convergence [11]. In view of this, it is fair to compare the performance of the proposed algorithm with either of these approaches. As such, a GA based approach is selected as a control in the context of the study.

Immune inspired vehicle routing and planning algorithms are summarized here though the numbers are few. Representative works include an immunity-based behavioral model was proposed by Lau et al. [12] that demonstrated the cooperation of the Automated Guided Vehicles (AGVs) through controlling the movement and actions of AGVs to the given tasks in a warehousing operation. Experimental studies also showed that proper communication and information exchange improved the efficiency of AGVs. Recently, Masutti & de Castro [13] developed a hybrid algorithm with Artificial Neural Network and Artificial Immune Systems to solve capacitated vehicle routing problems. Regarding the performance of the algorithm, good quality solutions were resulted with small computational effort.

3 Liaisons with the Cell-Mediated Immune Response

In our study, an immune inspired dispatching algorithm is developed based on the paradigm of the biological immune system to optimize the dwell time of vehicles traveling between the quay side and the yard to perform typical berth operations in a port container terminal.

In the Cell-mediated immune system, the invaded pathogen is eliminated by T-cells through a series of immune mechanisms, namely, antigen specific recognition, proliferation and cytokine signaling. For antigen recognition, the antigen receptors on the lymphocytes recognize the antigens that are complementary to the structure of the receptors. Furthermore, the interactions (stimulation and suppression) of the immune cells and molecules produce a self-organizing and attain a state of equilibrium of the immune system as proposed by Jerne's immune network theory [14]. These mechanisms provide an adaptive, dynamic and explicit immune response to all invading antigens.

Vehicles dispatching in a typical port container terminal is dynamic in a changing operating environment (e.g. with real-time container loading and unloading requests) with interactions occur between various equipment, in particular, the cranes, tractors and the containers. In general, job orders (also known as the Containers Work Program (CWP)) refer to a set of containers to be loaded onto the vessel or to be discharged from the vessel to the tractor associated with each quay crane. Specifically, the handling of each job is a two-fold operation. For loading a container, the yard crane (YC) picks up the container from the yard and transports it to the quay side by a tractor where the container is loaded onto the vessel by a quay crane (QC). For the discharging operation, a container is discharged onto a tractor by QC. The discharged container is then transported to the yard where an YC unloads the discharged container to the yard. For container allocation, jobs are executed according to the sequence specified in the CWP. These are either specified as "Non-presentable containers" and "Presentable containers". The former describes a container that cannot be assigned to a tractor as previous containers have yet not been assigned to the deployed

tractors. Alternatively, if no previous containers are pending, a container is said to be a "Presentable Container" that can be assigned to a tractor according to the proposed dispatching algorithm. These characteristics are parallel to the immune cells behavior in the biological immune system, and terminal operation therefore has similar dynamics of the biological immune system. Explicitly, the entities taking part in the terminal operations can be mapped onto the immune cells and molecules (Table 1) whilst the operation strategy is analogous to the immune functions that are discussed in the next section.

Table 1. The analogies between components of the biological immune system and the terminal model

Biological Immune System	AIS-based Vehicle Dispatching Framework
Antigen	Jobs (also called Container Work Program)
Peptide	Non-presentable containers ("Non-presentable Jobs)
Peptide- MHC Complex	Presentable Containers ("Presentable Jobs)
MHC molecules	The available crane for handling a two-fold operation (i.e. QC for discharging containers and YC for the loading containers)
T-cells	Tractors
Receptors of T-cells and surface receptors of immune cells/ molecules	Attributes of the corresponding entities

4 An AIS-Based Vehicle Dispatching Framework

As aforementioned, cell-mediated immune response is adopted by the proposed algorithm for solving vehicle dispatching problem. The main objective of the algorithm is to minimize the total dwell time of the vehicles in cooperation with other interacting. This is defined by the objective function of Equation 1.

$$\min \sum_{c=1}^{\delta_c} \sum_{x=1}^{\delta_{J_x^c}} f(WT_{cx}, TT_{cx}) \tag{1}$$

with

WT_{cx} Waiting time of the x^{th} container that is handled by *vehicle c* at the corresponding crane (on-line data)

TT_{cx} Transportation time to handle the x^{th} container by *vehicle c*, this refers to the transportation time between (i) the destination of the $(x\text{-}1)^{th}$ container and the origin of the x^{th} container (ii) the origin and the destination of the x^{th} container. This is represented by a probability density function with a mean 120 sec. and standard deviation 40 sec.

The total dwell time (in terms of unit second) of deployed vehicles is computed by the time consumed for the assigned container that is handled by *vehicle c*. Here, a set of containers (denoted by *x*) that is assigned to *vehicle c* is defined by:

$$J_x = \{ j_1^c, j_2^c, j_3^c, \ldots j_x^c \} \quad where\, x \in N,\ x = 1,2,3 \ldots \delta_{j_x^c} \tag{2}$$

These time components are computed by the stimulation and suppression function that is discussed in Section 4.3. Based on the proposed analogy of the biological immune system and the proposed algorithm (Table 1), the immune mechanisms adopted by the proposed framework are incorporated as depicted in Fig. 1.

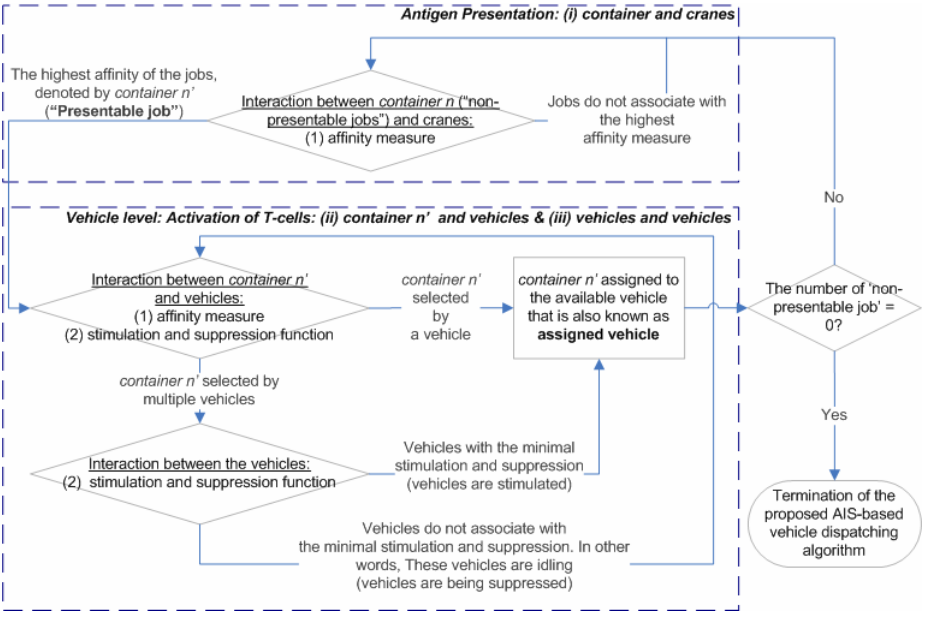


Fig. 1. The mechanisms in the cell-mediated immune response of T-cells are (i) Antigen Presentation (Section 4.2) and (ii) Activation of T-cells (Section 4.3) that are incorporated in the proposed algorithm. These mechanisms are control the actions of Job Presentation and Job Assignment.

4.1 Entities Representation

In the proposed algorithm, each entity is represented by a chain of attributes (Equations 2 - 5); this is also known as the shape space as depicted in Fig. 2. For example, the attributes of a job include job number, movetype (Loading or Discharge), the equipment number of the required yard crane and quay crane. On the other hand, the interaction between the entities is measured by an affinity measure that quantify the compatibility between the entities, such as the interaction between (i) jobs and cranes, (ii) jobs and vehicles, and (iii) vehicles and vehicles as depicted in Fig. 1. The details of each pair of interactions are given in the next session.

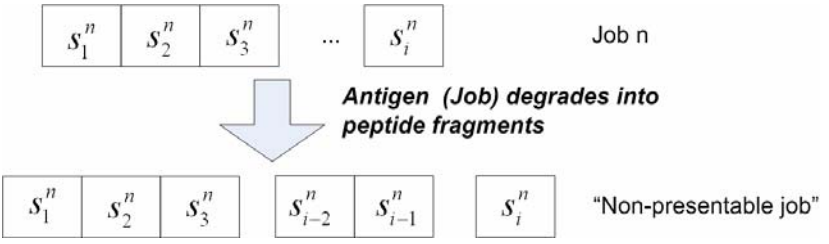


Fig. 2. Process of the degradation of antigen: entity (e.g. Job) is represented by an *I*-dimensional shape-spaces in the proposed algorithm, where $I = 1,2,3, \ldots i$

For the notation of the employed entities, vehicles (denoted by *c*), cranes (denoted by *m*) and containers (denoted by *n*) are defined in the abstract terminal model as follows:

$$c \in C \text{ where } c = 1,2,3...\delta_c \tag{3}$$

$$m \in M \text{ where } m = 1,2,3...\delta_m \tag{4}$$

$$n \in N \text{ where } n = 1,2,3,...\delta_n \tag{5}$$

For the above formulation of the involved entities, *c*, *m* and *n* are the index of the corresponding entities whereas δ_c, δ_m and δ_n are the total number of deployed vehicles, cranes and containers respectively.

4.2 Job Presentation

As aforementioned, containers are handled by specific quay cranes that are assigned according to particular working sequences. To ensure that the loading/unloading of containers are proceeded according to these pre-defined sequences, jobs have to be activated (as presentable jobs) prior to notify the deployed vehicles. As such, the mechanism of antigen presentation in the cell-mediated immune response is adopted to control this process. For job assignment between containers and corresponding cranes, affinity measure between the 'non-presentable' jobs and the cranes is formulated as follows:

The affinity of each pair of attributes (s_i^n, s_i^m) and the affinity of *container n* (ε_{nm}) to the available crane is computed according to Equations (6) and (7):

$$aff(s_i^n, s_i^m) = \begin{cases} 1 & if\ s_i^n \neq s_i^m \\ 0 & \text{othewise} \end{cases} \tag{6}$$

$$\varepsilon_{nm} = \sum_{i=1}^{\delta_i} aff(s_i^n, s_i^m) \tag{7}$$

In the control action of *Job Presentation*, two pairs of attributes are encountered: (i) the expected job number of the crane and, (ii) the index of the crane that is compatible with the number assigned to each job and the crane specified by the job respectively. As a result, a container having the highest affinity with a corresponding crane will be activated. This is also termed as the container is 'presentable' to the deployed vehicle and is denoted by *container n'* (as shown in Fig. 1), where $n' \in N$.

4.3 Vehicle Cooperation in Job Assignment

As aforementioned, the proposed algorithm inherits the characteristics of the immune cells, in which the deployed vehicles exhibit effective communication with minimum overhead, and cooperation among other terminal equipment based on the information collected from the environment. The mechanism of (a) antigen recognition and (b) stimulation and suppression of the immune cells are introduced to attain a highly responsive system.

Affinity between *container n'* and *vehicle c.* The matching between the 'presentable' jobs and the deployed vehicles is the first milestone of achieving a cooperative working environment. Similar to the affinity measure defined in Equations (6) and (7), the affinity of *container n'* with respect to the vehicle ($\varepsilon_{n'c}$) is computed according to Equations 8 and 9:

$$aff(s_i^{n'}, s_i^c) = \begin{cases} 1 & if\ s_i^{n'} = s_i^c \\ 0 & \text{othewise} \end{cases} \tag{8}$$

$$\varepsilon_{n'c} = \sum_{i=1}^{\delta_i} aff\ (s_i^{n'}, s_i^c) \tag{9}$$

In order to minimize the communication overhead with other cooperating vehicles, a vehicle communicates with a 'presentable' job within its 'vicinity' at a distance denoted by *d* (depicted in Fig. 3) measured from the vehicle's current location.

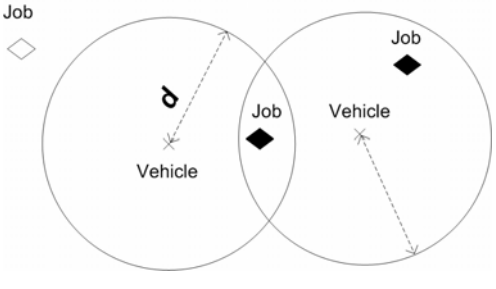


Fig. 3. A vehicles (drawn as black crosses) and its vicinity with distance *d*

In Fig. 3, the jobs (drawn as a black square) are detectable by the vehicles, whereas the job (drawn as a white square), is out of the detectable range or not in the vicinity of the vehicles.

Stimulation and suppression for *vehicle c*. As defined by Equation 10, the proposed algorithm is to optimize the total dwell time of the deployed tractors. In other words, the traveling time and waiting time of vehicles should be minimized. In order to minimize the objective defined by Equation 1, four essential time components for quantifying terminal operations are considered for each pair of *vehicle c* and *container n'*, namely, (i) urgency, (ii) traveling time of *container n'*, (iii) remaining processing time and (iv) pending time at the cranes specified by *container n'*. These essential time components are incorporated by Equation (10), which is derived based on the inspiration from the dynamics of the concentration of the immune cells [15] [16]. Equation 10 provides an alternative means to measure the concentration of the immune cells based on the stimulation and suppression of *vehicle c* with respect to *container n'*.

$$f(t_{n'}^{c}) = [t_1 - t_{CurrentTime}] - [t_2 + t_{2'}] + [\sum (t_3 - t_{CurrenTime}) + \sum (t_{3'} - t_{CurrenTime})] - \sum [t_4 + t_{4'}] \quad (10)$$

The definitions of these time components are given below:

i) Urgency of *container n',* in which t_1 denotes the time of being an activated container *n'*. The most urgent container is associated with the highest $(t_1 - t_{CurrentTime})$. This implies that the most urgent container should be handled in the highest priority, such that the efficiency of cranes and the terminal operations is maximized.

ii) Traveling time between (i) the current location of *vehicle c* and the origin of *container n'* (denoted by t_2), and (ii) the origin and the destination of *container n'* (denoted by $t_{2'}$). *Vehicle c* is suppressed if it spends a long time on dispatching.

iii) The remaining processing time is given by $(t_3 - t_{CurrentTime,})$ and $(t_{3'} - t_{CurrentTime,})$, *where* t_3 *and* $t_{3'}$ refer to the commence time of the container at *crane m* (origin of the *container n'*) and *m'* (destination of the *container n'*) of *container n'* respectively. The shorter the remaining processing time, *vehicle c* will be stimulated in the proposed *stimulation and suppression function*.

iv) Pending time of the containers queuing up at *crane m* (origin of the *container n'*) and *m'* (destination of the *container n'*), that are denoted by t_4 and $t_{4'}$.

Specifically, time component (ii) aimed at minimizing the traveling time of *vehicle c* whilst time components (iii) and (iv) are for reducing the waiting time of *vehicle c* given in Equation 1. The container with the highest $f(t_{n'}^{c})$ will be the next activated job, which will be allocated to *vehicle c* and denoted by *x*. This is also referred as the x^{th} container to be handled by *vehicle c* (Equation 2).

5 Simulation Studies

Taking reference from the discussion of Section 2, the effectiveness of the proposed AIS-based algorithm is studied and compared with a GA-based dispatching algorithm developed based on the operations as aforementioned. We deployed the two algorithms to investigate the stochastic behavior of terminal operation; it is worth

noting that these two algorithms are somehow different in nature. In particular, the former is a renounce planning approach; this means that the immediate requests are handled by evaluating the jobs that are locating at the vicinity of the vehicle. On the other hand, the latter is a re-optimizing planning strategy that performs iterative search to obtain optimal solutions. Nonetheless, the comparison justifies the effectiveness of the immune inspired algorithm in solving an engineering problem with distributed and dynamic nature.

In the context of the study, the quality of the obtained solutions (in terms of the total dwell time and the utilization of vehicles) and the computation complexity is studied based on a simplified terminal model consists of 15 containers, 4 QCs, 5 YCs and the number of vehicles deployed is varied from two to twelve (data extracted from the actual CWP obtain from the terminal operator). The results presented represent the average of ten experiments for each instance.

5.1 Experimental Setup

In the experiment study, the terminal operation presented is based on real data representing typical port container terminal operation. In addition, the configuration of the port container terminal and the operation constraints are also addressed by the proposed algorithm as summarized in the table below:

Table 2. The key parameters of the AIS-based and GA-based algorithms as defined based on the configuration and set up of a real container terminal

Configurations of terminal operations	Experimental setups
Distance between the employed cranes	Ranging from 0.6km to 2.0km
Speed of vehicles	30km/hr
Productivity of Quay Cranes (QCs)	Normally distributed with a mean of 45 sec. and standard deviation 15 sec.
Productivity of Yard Cranes (YCs)	Normally distributed with a mean of 100 sec. and standard deviation 16 sec.
No. of containers handled by a vehicle	1
No. of containers handled by a crane	1

A GA-based Dispatching Algorithm is also developed for evaluating the performance of the proposed algorithm. The operation of the GA-based algorithm is depicted in Fig. 4, with the parameters and constraints defined in Table 2. The strings of population of the GA-based continue to evolve until an optimal solution is reached. In our implementation of the GA-based algorithm, the strings of population represent the assignments of vehicles to the associated containers that are arranged in the sequence as specified by the CWP. The permutations are reproduced through the operations of crossover and mutation that inject randomness into the new permutations to retain the diversity of population. In addition, the fitness function is computed to eliminate those poor permutations within the population. As such, the population evolves iteratively to maximize the fitness function.

Based on the results of a large number of experimentations with the GA-based algorithm, the optimal solutions are obtained for the instances given above (not explicitly

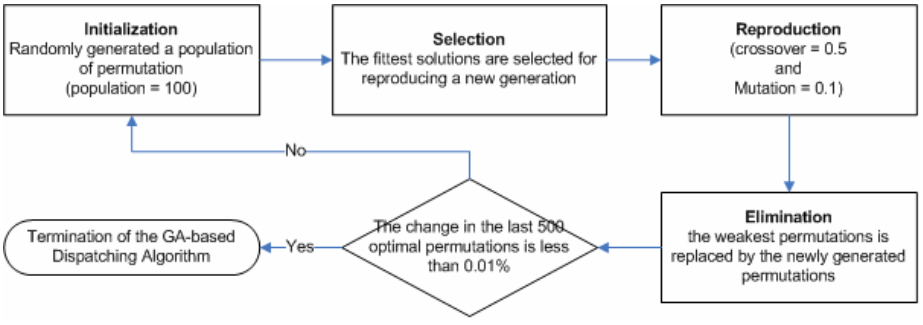


Fig. 4. The operation of the GA-based Dispatching Algorithm

presented), and suitable crossover and mutation rates are determined (set as 0.5 and 0.1 respectively) and these values are used throughout the study.

5.2 Quality of Solutions

According to the results given in Fig. 5, the total dwell time obtained by the AIS-based algorithm increases as the number of deployed vehicles increases. This is because the waiting time of the terminal equipments, namely, QCs and YCs, is lengthened due to the long queue created by the large number of deployed vehicles. The dwell time obtained from the AIS-based algorithm varies from 15-45% more than the dwell time obtained from the GA-based algorithm (16-80 minutes), especially when the number of vehicles deployed is large.

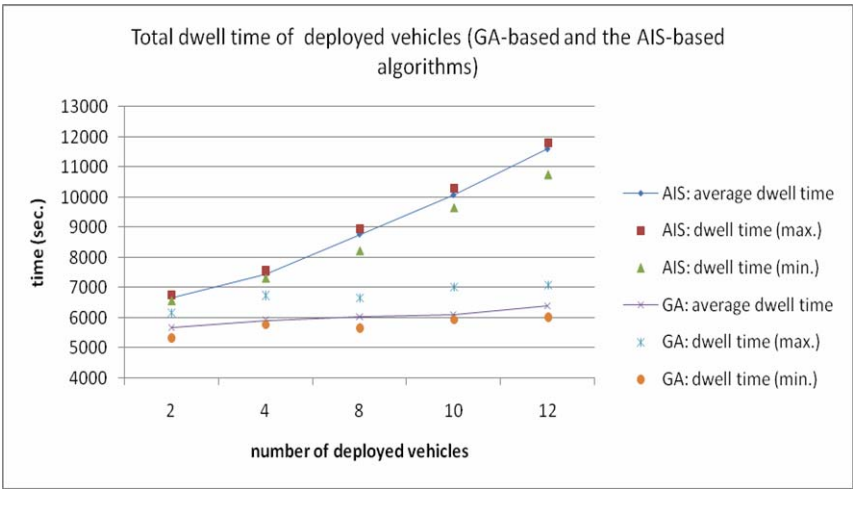


Fig. 5. The total dwell time of all employed vehicles. For the AIS-based algorithm, the total dwell time is directly proportional to the number of vehicles employed whereas the recorded total dwell time is similar in all scenarios according to the GA-based algorithm.

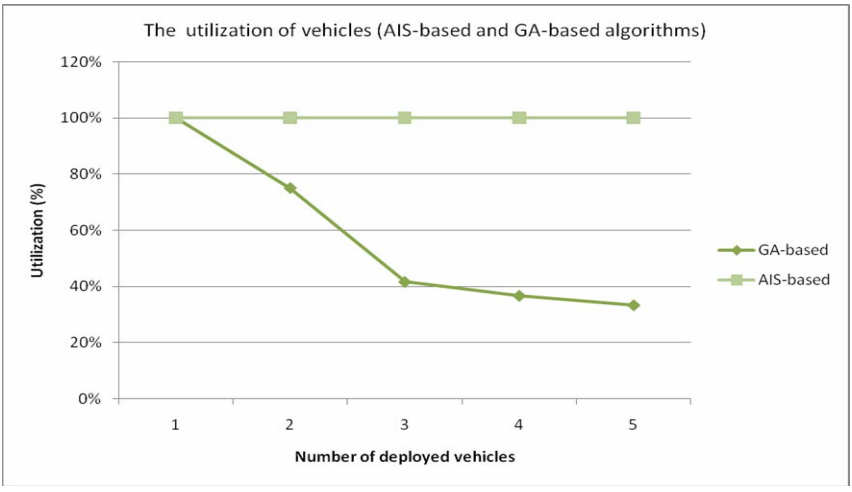


Fig. 6. The low utilization of vehicles in GA-based algorithm is due to the presence of idle vehicles, compared to the AIS-based algorithm. (Remarks: utilization of vehicles is defined as the number of operated vehicles out of the fleet of deployed vehicles. "vehicles deployed" is defined as the vehciles assigned initially for each simulation runs; "vehicles operated" is refered to the actural number of vehicles executing the jobs in each simulation runs).

Despite of the longer dwell time of vehicles obtained from the AIS-based approach, the utilization of the vehicles is higher than the GA-based algorithm as depicted in Fig. 6. In the GA-based algorithm, only three to four vehicles are in operation whilst the others are idling when the number of deployed vehicles is greater than eight. As such, the total dwell time obtained by the GA-based algorithm is similar. In other words, four vehicles are sufficient to handle fifteen containers.

Though the optimal number of vehicles cannot be directly evaluated by the AIS-based algorithm due to the stochastic nature of the problem, the AIS-based algorithm provides an efficient means for evaluating the time required for the employed vehicles and for the scheduling other resources such as crane and crane operators in real-time at a lower computational cost as discussed in Section 5.3.

5.3 Computation Complexity

The time required to obtain the solutions upon termination of the simulations is used to quantify the computation complexity of the algorithms. Our study shown that the computation complexity of the AIS-based algorithm is less than the GA-based algortihm, in particular, for a large number of vehicles deployed (Fig. 7). For the proposed AIS-based algorithm, the majority of computational effort is spent in data retrieval, for instance, in the matching of the 'non-presentable'containers and the employed termianl equipment; and the 'presentable'containers and the deployed vehicles using the stimulation and suppression function mentioned in Section 4.2. In Fig. 7, a reduction of computation cost is observed when the simplified stimulation and suppression function is deployed. This is due to the great reduction of data being

processed as described by Equation 11 (modified based on Equation 10, with the notations described above).

$$f'(t_{n'}^{c}) = -[t_2 + t_{2'}] - \sum [t_4 + t_{4'}] \tag{11}$$

On the other hand, the GA-based algorithm performs a large number of evaluations of the permutation in an iterative manner. This is largely due to the crossover and mutation operations for reproducing the new permutations (children) that inherit the characterisitcs of different combination of permutations (parents); in the mean time, the genetic sequence is changed from its original state by mutation. As a result, a large number of permutations irrespective of their validity is generated. These invalid permutations also require compuational effort on verification. In the context of this study, some of the experiments required more than 3000 interations in order to generate the solutions (Fig. 5). The AIS-based algorithm on the other hand required much reduced computation effort as it only requires to consider the vehicles in its vicinity. In this case, there are at most eleven vehicles in the vicinity, and this number is continuously reduced as the jobs have been completed by other vehicles.

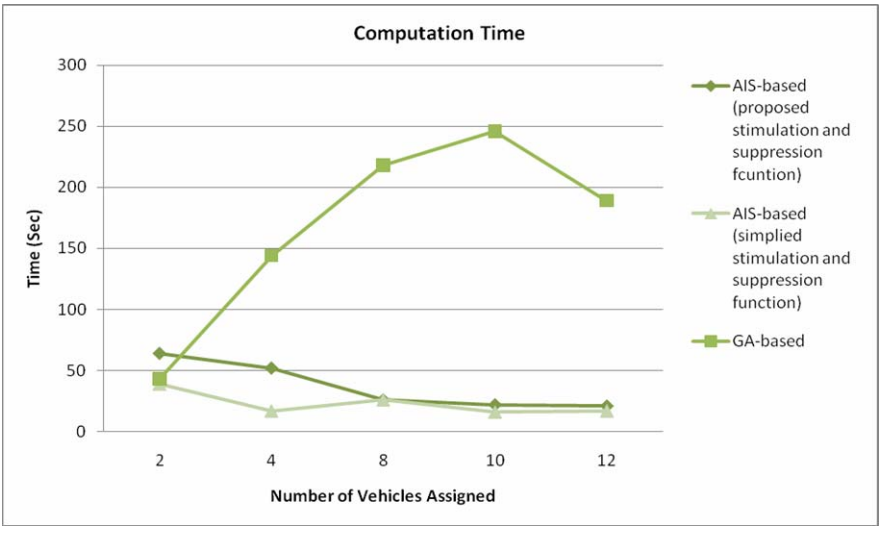


Fig. 7. The computation cost required for the AIS-based and GA-based algorithms. This refers to the time required to obtain the ultimate solutions given in Fig. 5.

6 Conclusion and Future Work

In this study, an AIS-based algorithm for solving vehicle dispatch problem is proposed. The algorithm is demonstrated to tackle a typical vehicle dispatch problem in a port container terminal that involves the handling of a sequence of dynamic and distributed operations. According to the experimental studies, the proposed AIS-based algorithm provides satisfactory performance in terms of (1) the utilization of vehicles and (2) the computation cost incurred. The total dwell time obtained from AIS-based

algorithm is close to the dwell time obtained from the GA-based algorithm, which is widely accepted as a standard tool for evaluating optimal resources utilization in terminal operation with good performance.

In the context of our study, the small-scale yet representative problem (12 vehicles, 15 containers, 9 cranes) provides promising results for further investigation. We are looking to further our research in (i) deploying the proposed AIS-based algorithm in a large scale terminal configuration, and (ii) evaluating the effectiveness of the AIS-based algorithm with respect to its robustness and the adaptability in the near future.

Acknowledgement

The work described in this paper was partly supported by the Research Grant Council of the Hong Kong Special Administrative Region, PRC under the GRF Project No. HKU713707E.

References

1. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer, Heidelberg (2002)
2. Hart, E., Timmis, J.: Application areas of AIS: the past, the present and the future. Applied Soft Computing 8, 191–201 (2008)
3. Grunow, M., Gunter, H.O., Lehmann, M.: Dispatching multi-load AGVs in highly automated seaport container terminals. OR Spectrum 26, 211–235 (2004)
4. Kim, K.H., Bae, J.W.: A look-ahead dispatching method for automated guided vehicles in automated port container terminals. Informs(Transportation Science) 38(2), 224–234 (2004)
5. Bish, E.K., Chen, F.Y., Leong, Y.T., Nelson, B.L., Ng, J.W.C., Simchi-Levi, D.: Dispatching vehicles in a mega container terminal. OR Spectrum 27, 491–506 (2005)
6. Gendreau, M., Guertin, F., Potvin, J.Y., Seguin, R.: Neighborhood search heuristics for a dynamic vehicle dispatching problem with pick-ups and delivers. Transportation Research Part C 14, 157–174 (2006)
7. Potvin, J.V.: A review of bio-inspired algorithms for vehicle routing. Bio-inspired Algorithm for the Vehicle Routing Problem, SCI 161, 1–34 (2009)
8. Silva, C.A., Sousa, J.M.C., Runkler, T.A.: Rescheduling and optimizing of logistics processes using GA and ACO. Engineering Applications of Artificial Intelligence 21(3), 343–352 (2008)
9. Gambardealla, L.M., Taillard, E., Agazzim, G.: MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows, New Ideas in Optimization, pp. 63–76. McGraw-Hill, New York (1999)
10. Skrlec, D., Filipec, M., Krajcar, S.: A heuristic modification of genetic algorithm used for solving the single depot capacited vehicle routing problem. In: IEEE Proceedings on Intelligent Information Systems, pp. 184–188 (1997)
11. Zhao, F.G., Sun, J.S., Liu, W.M.: A hybrid genetic algorithm for the traveling salesman problem with pickup and delivery. International Journal of Automation and Computing, 97–102 (2009)
12. Lau, H.Y.K., Wong, V.W.K., Lee, I.S.K.: An immunity approach to strategic behavioral control. Engineering Applications of Artificial Intelligence 20(3), 289–306 (2007)

13. Masutti, T.A.S., de Castro, L.N.: A Neuro-Immune Algorithm to solve the capacitated vehicle routing problem. In: Bentley, P.J., Lee, D., Jung, S. (eds.) ICARIS 2008. LNCS, vol. 5132, pp. 210–219. Springer, Heidelberg (2008)
14. Jerne, N.: Towards a network theory of the immune system. Ann. Immunology 125C, 373–389 (1974)
15. Na, D., Lee, D.: Mathematical modeling of immune suppression. In: Jacob, C., Pilat, M.L., Bentley, P.J., Timmis, J.I. (eds.) ICARIS 2005. LNCS, vol. 3627, pp. 182–192. Springer, Heidelberg (2005)
16. Wierzchon, S.T.: Idiotypic networks as a metaphor for data analysis algorithm. In: Saeed, K., Pejas, J. (eds.) Information Proceeding and Security Systems, pp. 389–400. Springer, US (2005)

Author Index