New Constructions in Cellular Automata

Edited by
David Griffeath
Cristopher Moore

A VOLUME IN THE
SANTA FE INSTITUTE STUDIES IN THE SCIENCES OF COMPLEXITY

New Constructions in Cellular Automata

Santa Fe Institute
Studies in the Sciences of Complexity

Lecture Notes Volume

Author	*Title*
Eric Bonabeau, Marco Dorigo, and Guy Theraulaz	Swarm Intelligence: From Natural to Artificial Systems
M. E. J. Newman and R. G. Palmer	Modeling Extinction

Proceedings Volumes

Editor	*Title*
James H. Brown and Geoffrey B. West	Scaling in Biology
Timothy A. Kohler and George J. Gumerman	Dynamics in Human and Primate Societies
Lee A. Segel and Irun Cohen	Design Principles for the Immune System and Other Distributed Autonomous Systems
H. Randy Gimblett	Integrating Geographic Information Systems and Agent-Based Modeling Techniques
James P. Crutchfield and Peter Schuster	Evolutionary Dynamics: Exploring the Interplay of Selection, Accident, Neutrality, and Function
David Griffeath and Cristopher Moore	New Constructions in Cellular Automata

New Constructions in Cellular Automata

Editors

David Griffeath
University of Wisconsin
Madison, WI

Cristopher Moore
Santa Fe Institute
Santa Fe, NM
and
University of New Mexico
Albuquerque, NM

Santa Fe Institute
Studies in the Sciences of Complexity

OXFORD
UNIVERSITY PRESS
2003

OXFORD
UNIVERSITY PRESS

Oxford New York
Auckland Bangkok Buenos Aires Cape Town Chennai
Dar es Salaam Delhi Hong Kong Istanbul Karachi Kolkata
Kuala Lumpur Madrid Melbourne Mexico City Mumbai Nairobi
São Paulo Shanghai Taipei Tokyo Toronto

Published by Oxford University Press, Inc.
198 Madison Avenue, New York, New York 10016

www.oup.com

Library of Congress Cataloging-in-Publication Data

CIP is available from
the Library of Congress

ISBN 0-19-513717-5; ISBN 0-19-513718-3 (pbk.)

1 3 5 7 9 8 6 4 2

Printed in the United States of America
on acid-free paper

About the Santa Fe Institute

The *Santa Fe Institute* (SFI) is a private, independent, multidisciplinary research and education center, founded in 1984. Since its founding, SFI has devoted itself to creating a new kind of scientific research community, pursuing emerging science. Operating as a small, visiting institution, SFI seeks to catalyze new collaborative, multidisciplinary projects that break down the barriers between the traditional disciplines, to spread its ideas and methodologies to other individuals, and to encourage the practical applications of its results.

All titles from the *Santa Fe Institute Studies in the Sciences of Complexity* series will carry this imprint which is based on a Mimbres pottery design (circa A.D. 950–1150), drawn by Betsy Jones. The design was selected because the radiating feathers are evocative of the out-reach of the Santa Fe Institute Program to many disciplines and institutions.

Contributors List

Kellie M. Evans, *California State University, 18111 Nordhoff Street, Northridge, CA 91330; E-mail: kellie.m.evans@csun.edu*

Nick M. Gotts, *MLURI, Land Use Science Group, Aberdeen AB15 8QH, Scotland, United Kingdom; E-mail: n.gotts@mluri.sari.ac.uk*

Janko Gravner, *University of California, Mathematics Department, Davis, CA 95616; E-mail: gravner@math.ucdavis.edu*

David Griffeath, *University of Wisconsin, Department of Mathematics, Van Vleck Hall, 480 Lincoln Drive, Madison, WI 53706; E-mail: griffeat@math.wisc.edu*

Dean Hickerson, *Mathematics Department, University of California, Davis, CA 95616; E-mail: dean@math.ucdavis.edu*

George E. Homsy, *Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139; E-mail: ghomsy@ai.mit.edu*

Joy V. Hughes, *3954 Jarvis Road, Scotts Valley, CA 95066; E-mail: hughes@scruznet.com*

Norman H. Margolus, *Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 545 Technology Square, Cambridge, MA 02139; E-mail: nhm@mit.edu*

Bernd Mayer, *Institute for Theoretical Chemistry, and Radiation Chemistry, University of Vienna, UZAII, Althanstraße 14, A-1090 Vienna, Austria; E-mail: bernd@asterix.msp.univie.ac.at*

Cristopher Moore, *Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, NM 87501 and University of New Mexico, Department of Computer Science and Department of Physics and Astronomy, Albuquerque, NM 87131; E-mail: moore@santafe.edu*

Gadi Moran, *Department of Mathematics, University of Haifa, Haifa 31905, Isreal; E-mail: gadi@mathcs2.haifa.ac.il*

Mark D. Niemiec, *2260 Par Lane PH7, Willoughby Hills, OH 44094; E-mail: mniemiec@interserv.com*

Martin Nilsson, *Los Alamos National Laboratory, EES-5 & T-CNLS, Mail Stop D450, Los Alamos, New Mexico 87545 ; E-mail: nilsson@lanl.gov*

Nienke A. Oomes, *University of Wisconsin, Department of Economics, 1180 Observatory Drive, Madison, WI 53706; E-mail noomes@ssc.wisc.edu*

Steen Rasmussen, *Los Alamos National Laboratory, EES-5 & T-CNLS, Mail Stop D450, Los Alamos, NM 87545; E-mail: steen@lanl.gov*

Rudy Rucker, *Department of Mathematics and Computer Science, San Jose State University, San Jose, CA 95192; E-mail: rucker@mathcs.sjsu.edu*

Raissa D'Souza, *Department of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139; E-mail: raissa@im.lcs.mit.edu*

David Whitten, *CST-1 MS J565, Los Alamos National Laboratory, Los Alamos, NM 87545; E-mail: whitten@lanl.gov*

This page intentionally left blank

Contents

Preface

This book is the long-awaited proceedings of a conference, held at the Santa Fe Institute in December, 1998, and sponsored by the National Science Foundation. "New Constructions in Cellular Automata" brought people together to discuss topics ranging from modeling physics and economics, to reversible computation, to the latest discoveries of bugs, puffers, and all the flora and fauna of the cellular automaton world.

The first part of the book focuses on the best-loved CA rule, Conway's Life, and its variants. In the first chapter, Nick Gotts answers the cosmological question of what happens in a random low-density initial condition, showing that a surprising amount can be learned about what structures self-organize in the early Life universe. In the next chapter, Mark Niemiec shows us the latest methods of constructing complex objects from collisions of gliders, an essential engineering skill for Life devotees. David Griffeath and Dean Hickerson answer one of Life's open questions: whether an initial seed exists that populates the universe with an irrational density. Matthew Cook shows that telling when a "still life," a configuration which is stable under the Life rule, can be divided into

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

separate pieces is an NP-complete problem. Moving on to Life's generalizations, Kellie Evans introduces Larger than Life and HighLife, and finds many families of replicators in these rules.

To bring the book alive and to help the reader explore the many open questions remaining in the field, many of the Life patterns discussed in these chapters can be downloaded from the book's companion web page, ⟨http://psoup.math.wisc.edu/NewConstructions⟩.

In the next chapters, we put cellular automata to work as platforms for simulating phenomena in physics and economics. Janko Gravner introduces us to the mathematics of growth phenomena and studies the asymptotic shapes of various rules. Martin Nilsson, Steen Rasmussen, Bernd Mayer, and David Whitten discuss how to use lattice gases to simulate hydrophobic and hydrophilic polymers. (In recent work, they have achieved the formation of micelles with this method, and shown that CAs can reach time-scales several orders of magnitude longer than standard molecular dynamics (MD) simulations.)

Raissa D'Souza, George Homsy and Norman Margolus then use reversible CAs to model how an aggregating cluster reaches equilibrium with its environment, and show that their reversible aggregation (RA) rule can simulate universal reversible logic. Margolus shows that a soft-sphere model also has this degree of computational power, and Nienke Oomes rounds out this section by using CAs to model how economic inequality can persist in emerging markets.

In the concluding chapters, Joy Hughes gives us beautiful examples of how CAs can be used in art and video, Rudy Rucker extols the virtues of CAs whose states are continuous rather than discrete, and Gadi Moran shows a phase transition in majority-voting rules on graphs.

We are deeply indebted to the Santa Fe Institute and Oxford University Press for making this book possible, and especially to Della Ulibarri and Ronda K. Butler-Villa for their tireless work and extraordinary patience. We also thank the University of Wisconsin, Madison, for hosting the Primordial Soup web page and the book's companion page, ⟨http://psoup.math.wisc.edu/NewConstructions⟩, where many patterns and simulations relevant to these chapters can be downloaded. Finally, we dedicate this book to Oscar, Rascal, Scurry, and Spootie the Cat.

Cristopher Moore
Santa Fe Institute and University of New Mexico

David Griffeath
University of Wisconsin

Self-Organized Construction in Sparse Random Arrays of Conway's Game of Life

Nicholas M. Gotts

1 INTRODUCTION

The construction problems and techniques described in this chapter arose out of a single problem:

> What happens in very low density infinite random arrays of Conway's *Game of Life*?

However, the work reported has wider implications, briefly discussed in the final section.

Conway's Game of Life (henceforth GoL) is a deterministic cellular automaton (CA), which is binary (a cell has two possible states: 0 and 1) and runs on an infinite two-dimensional grid of cells. A deterministic CA cell's state at

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

time step t is determined, according to a *transition rule*,[1] by those of a set of *in-neighbors* at step $t-1$, and its own state at step $t-1$ can affect the state of its *out-neighbors* at t. In GoL, in-neighbors and out-neighbors coincide, and include the cell itself. The neighborhood is a 3×3 square of cells. GoL's transition rule specifies that a cell is in state 1 at step t if and only if either of the following held at $t-1$.

1. The cell and either two or three other cells in its neighborhood were in state 1.
2. The cell was in state 0, and exactly three other cells in its neighborhood were in state 1.

By a *random array*, I mean one in which the initial probability p of each cell being in state 1 is the same for all cells, and the initial state is determined independently for each cell. Of course, we cannot actually construct such an array, but we can reason about it. Toward the end of the chapter, large finite random arrays will be considered, but it is simpler to start with the infinite case. In fact, none of the reasoning used in the infinite case depends upon the distribution of state 1 cells being strictly random, provided the frequency of all finite arrangements of cell-states is as expected in a random array with the same density of state 1 cells. A sparse random array is one in which p is very low (a more precise definition is given below).

In a popular book on GoL, Poundstone [20] says:

> Speculation about "living" Life patterns focuses on infinite, low-density random fields.... If there are self-reproducing Life patterns, they would have room to grow in such a field [pp. 175–176].

Poundstone may have drawn on material published in Berlekamp et al. [3], which claims that self-replicating patterns can be shown to exist in GoL. (Such patterns are finite arrangements of state 1 cells that produce multiple disjoint copies of themselves in an otherwise empty—state 0—array.) Berlekamp et al. [3], then say:

> Inside any sufficiently large random broth, we expect *just by chance*, that there will be some of these self-replicating creatures.... It's probable...that after a long time, intelligent self-reproducing animals will emerge and populate some parts of the space [p. 849] (emphasis in original).

[1] The chapter uses a good deal of CA and Game of Life terminology, some of it novel. Terms of this kind are italicized when first used, and explained unless their meaning is clear from context.

Notice that *low-density* random arrays are not specified. Most studies of the development of random arrays have been based on simulating finite random arrays of moderate density; for example, see Bagnoli et al. [1], Garcia et al. [10], Gibbs and Stauffer [11], and Sales et al. [23]. One simulation study Malarz et al. [17] has looked at low (and high) as well as moderate densities. However, *very* low density random arrays, unlike those of higher density, appear to offer scope to analytical approaches, for reasons that will appear in the course of the chapter.

The investigations of GoL reported here are the most developed aspect of an attempt to answer a broader question:

> Are there CA in which self-reproducing entities will emerge from initially structureless configurations, and evolve to arbitrary levels of behavioral complexity? If so, what are the simplest transition rules and global network topologies permitting this?

Work by others most obviously relevant to this question includes Chou and Reggia [5], Langton [16], and Reggia et al. [21].

2 CHUNKS, PATTERNS, AND CLUSTERS IN RANDOM GAME-OF-LIFE ARRAYS

The term *global configuration* will be used to refer to a complete assignment of cell states to the cells of a cellular automaton (CA); *array* will be used for the complete set of cells and their neighborhood links itself, or with the same meaning as "global configuration" when no confusion will arise. Similarly, *chunk* will be used either for a finite rectangle of cells within an array of GoL or any other CA that runs on a two-dimensional square lattice, or for such a set of cells along with a specific assignment of cell states to all its cells. Considering chunks with $n \times m$ cells in an infinite array of GoL or any other binary square lattice CA, there will be 2^{nm} distinct kinds of chunk with those dimensions (i.e., that many possible arrangements of cell states within such a chunk).

It will be useful to restrict the definition of *pattern* more than is implied by the quote from Poundstone [20] in the introduction. Henceforth, the term will mean a specific arrangement of a finite number of state 1 cells, on an infinite array otherwise consisting of state 0 cells. The successor of a pattern is always a pattern, if we include the *null pattern*: an array consisting only of state 0 cells. Patterns may, however, have predecessor arrays which contain an infinite number of state 1 cells. Three definitions of what counts as the same pattern can be given, although it will not always be important to distinguish these:

- If the cells of a GoL array are numbered, we could regard every finite set of cells as specifying a distinct *location-specific pattern.*

- Next, we could regard translations of a location-specific pattern as instances of the same *translation-defined pattern.*
- Third, we could count rotations and reflections of a translation-defined pattern as producing instances of the same *automorphism-defined pattern* (translations, rotations, reflections, and compositions of these operations constitute the automorphisms of the GoL network of cells, and GoL's transition rule is symmetric under them).

Beyond these three possibilities, if one automorphism-invariant pattern develops into another, they can be described as instances of the same *development class* of patterns. It follows that if two automorphism-defined patterns develop into the same pattern, they are also members of the same development class.

A collection of state 1 cells which may not be a pattern—because the array is not otherwise empty and/or because the group of state 1 cells is infinite in number—but which is isolated to some degree from any other state 1 cells in the array—will be called a *cluster.* Terminology defined for patterns generally transfers to clusters.

More precisely, a *0-cluster* is a maximal set of state 1 cells such that each distinct pair of cells (c_x, c_y) in the set is either a pair of immediate neighbors, or is linked by a sequence of members of the set $c_{i_1} \ldots c_{i_n}$, such that c_x is a neighbor of c_{i_1}, c_{i_1} of c_{i_2}, ... and c_{i_n} of c_y. To put it another way, there is a path from any cell in the set to any other, along links between pairs of neighboring cells (a "neighborhood link path"), that never goes through a state 0 cell. A *1-cluster* is a maximal set of state 1 cells such that there is neighborhood link path between any two members of the set that never goes through two successive state 0 cells, and a *d-cluster* is a maximal set of state 1 cells such that there is neighborhood link path between any two members of the set that never goes through $d + 1$ successive state 0 cells. A *cluster* is then simply a set of state 1 cells forming a d-cluster for some d. Notice that a d-cluster may also be a $(d + 1)$-cluster and, indeed, a pattern on an infinite array, as defined above, will be a d-cluster for all d exceeding some minimum value. The state 1 cells in a 1-cluster will not share a neighbor with any state 1 cells outside the cluster, so the states of the cluster's cells and their neighbors can be calculated for one step without considering anything outside the cluster. However, that step may split a 1-cluster into two or more parts, and/or merge previously distinct 1-clusters.

Some GoL global configurations have no predecessor. There are *orphan* or "Garden of Eden" chunks of cell-states which cannot be part of a global configuration with a predecessor [3, p. 829]. An $n \times m$ chunk can be shown to be an orphan by considering all possible $(n + 2) \times (m + 2)$ chunks: if none of these generates the $n \times m$ chunk in its $n \times m$ cell interior in a single step, it is an orphan. It is not known whether there are any chunks which can appear after a single step (at $t = 1$), but not after more than one.

Patterns which have no predecessor *patterns* will be called *nonconstructable*; any pattern which is not nonconstructable is 1-constructable; any 1-constructable

pattern with a 1-constructable predecessor is 2-constructable; and so on. Any pattern with at least one infinite sequence of predecessor patterns is ω-constructable. Any orphan chunk defines a corresponding nonconstructable pattern: just place the chunk in an otherwise empty array (note that since chunks of different dimensions may differ only in whether certain cells are specified as state 0, or are unspecified—i.e., left out of the chunk—an infinite number of orphan chunks correspond to the same nonconstructable pattern). The smallest known nonconstructable pattern (where "smallest" means, as it generally does here, having the fewest state 1 cells) is of size 143 [25].

Conversely, proof that a pattern is i-constructable implies that all chunks containing it and otherwise blank will exist at time $t = i$. However, there may be nonconstructable GoL patterns without the corresponding chunks being orphans. For a chunk to be an orphan, it must be impossible to produce that chunk from any predecessor configuration, irrespective of what the predecessor configuration produces outside the chunk boundaries, whereas a pattern is nonconstructable unless there is a configuration (specifically, a pattern) that can produce that pattern in an otherwise empty array.

In an infinite random array of any binary square lattice CA, with initial density p of state 1 cells, an $n \times m$ chunk of cells including a state 1 cells and $nm - a$ state 0 cells would occur with a density of $p^a(1-p)^{nm-a}$—that is, one in that number of cells would be (say) the top left corner of a chunk of cells with that arrangement of cell states. The exact density of any type of $n \times m$ chunk can be calculated for any values of p and t: simply enumerate all the $(n + 2t) \times (m + 2t)$ chunks that give rise to it in t steps, and calculate the density of each of these at $t = 0$ using the formula given above. However, the number of chunks to consider increases $\propto 2^{t^2}$. Moreover, no finite number of such calculations would show whether the chunk has a limiting density as $t \to \infty$ and if so, what that density is. In the case of infinite CA, it is reasonable to refer to the situation after any specific number of steps as belonging to the "short term," and to restrict "long term" to what happens as $t \to \infty$. Anything we want to know about "short-term" events is then calculable in principle, but in practice exact calculation cannot in general take us far. So far as the long term is concerned, we can say that any chunk containing an ω-constructable pattern and no other state 1 cells will always be present (i.e., belongs to the *limit set* of chunks).

The following patterns are known to be ω-constructable:

1. The null pattern is its own predecessor, and hence is ω-constructable, so any chunk consisting entirely of state 0 cells belongs to the limit set.
2. Any *still-life* (a pattern in which exactly the same cells are in state 1 at step $t+1$ as at step t) is ω-constructable, being its own n-step predecessor for any n.

3. Any *oscillator* (a pattern which is the same at $t + m$ as at t for some m) is ω-constructable: for any n, some phase of the oscillator is the oscillator's n-step predecessor.
4. Any *repeater* (a pattern which can be mapped onto its mth successor by a translation, for some m, the repeater's *translation period*) is ω-constructable: for any n, some translation of some phase of the repeater is the repeater's n-step predecessor. As defined here, repeaters include oscillators, oscillators include still-lifes, and still-lifes include the null pattern. Repeaters which are *not* oscillators are *spaceships*: they move across the array with a characteristic period and velocity. Any set of repeaters (oscillators) with the same velocity can be combined (in infinitely many ways) to constitute a *compound* repeater (oscillator), provided they are distant enough not to interfere with each other.
5. Anything that can be produced in a collision between two or more repeaters of different velocities, where those repeaters can initially be placed arbitrarily far apart, is ω-constructable. This case is discussed in more detail below.

Self-replicating patterns are not necessarily ω-constructable: there are CA [20, pp. 136–137] in which all (non-null) patterns self-replicate. All non-null patterns thus increase their number of state 1 cells monotonically, so no such pattern is ω-constructable. Nevertheless, we can say that if a CA supports self-replicating patterns (a proof that GoL does so, constructed by Conway and others soon after the discovery of GoL remains unpublished [6]), copies of all the arrangements of cell-states that define such patterns will occur in a random array of any density $0 < p < 1$. Moreover, they will continue to exist, and replicate, indefinitely, as there will initially be examples of such arrangements surrounded by arbitrarily large areas devoid of state 1 cells. What happens to the density of these and other types of arrangement in the long term, however, is unknown. No progress has been made in determining limiting densities for any particular type of chunk or cluster in random GoL fields, with the exception of the few chunks known to be orphans: their density falls to zero at $t = 1$ and remains zero thereafter.

In sum, both short-term and long-term properties of infinite random GoL arrays currently appear very resistant to analysis. When we consider arrays of very low density, however, the short-term situation improves, and it also turns out that a rather naturally defined "medium term" appears, about which a reasonable amount can be discovered.

3 SPARSE RANDOM ARRAYS IN CONWAY'S GAME OF LIFE

3.1 INITIAL CONSIDERATIONS

In the analysis of sparse infinite random GoL arrays, the value of p will be taken to be nonzero but *arbitrarily low*: whenever the analysis would differ according to whether $p < x$ for some positive x, it will always be assumed that indeed

$p < x$. An alternative way of expressing this is to say that the analysis centers on how infinite random GoL arrays behave as $p \to 0$.

If the value of p is very close to zero, most state 1 cells in the initial array will be isolated. If we consider translation-defined d-clusters in the initial array for any sufficiently low value of d, the relative frequency with which two types of d-cluster of the same size (with the same number of state 1 cells) occur, approaches 1 as $p \to 0$. Those consisting of a single cell will be the most common, those of size 2 will occur at approximately p times their density, and in general those of size s will be approximately p^s times as dense as those of size 1. To calculate the *exact* density of a particular type of cluster in the initial array for some specific p, it would be necessary to take account of the number of cells which must be in state 0 as well as the number that must be in state 1 to produce such a cluster, but for any given value of d, the approximations given can be made as close as required by setting p sufficiently low.

For any value of d, there will be a value of p below which only finite d-clusters will form in an infinite random array. This follows from a result in percolation theory [24]: given an infinite but locally finite vertex-transitive graph in which each node has z nonself neighbors, (the GoL array of cells is such a graph, with $z = 8$), no infinite 0-clusters will form below a critical value of p, where $p \leq 1/(z-1)$. As Stephen Silver has pointed out [26], it follows that there is some such critical p for every d in an infinite random GoL array, since adding links between every pair of nodes that are joined by a path of n or fewer links in a locally finite vertex-transitive graph, produces another locally finite vertex-transitive graph. In this new graph 0-clusters will correspond to $(n-1)$-clusters in the original: as $p \to 0$, the maximum value of d for which only finite d-clusters exist $\to \infty$.

Thus as $p \to 0$, clusters corresponding to the smallest patterns with particular dynamic properties become significant. These minimum-size patterns determine how common examples of clusters with that property are in the short term, and have a crucial influence on medium-term events.

3.2 SMALL GAME-OF-LIFE PATTERNS

This subsection reviews some of what is known about the smallest GoL patterns with particular dynamic properties.

The smallest GoL patterns that do not disappear in a single step consist of three state 1 cells. Among these, patterns consisting of three in an orthogonal row (the *blinker*), and of an orthogonally oriented "L" shape (the *preblock*) are the only ones that do not disappear on the second step. The blinker is a period 2 oscillator, switching from vertical to horizontal orientation and back; the preblock becomes a 2×2 *block* on the first step, and remains unchanged thereafter.

Four-cell patterns fall into seven development classes. As with three-cell patterns, the final result may be the null pattern, a block, or a blinker, but it may also be a *tub*, a *pond*, a *beehive*, or a symmetrical grouping of four blinkers termed

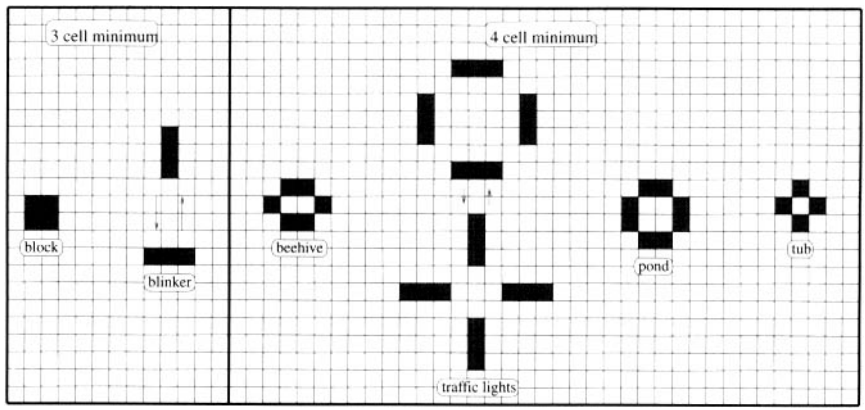

FIGURE 1 Oscillators derived from four or fewer cells.

traffic lights. These are shown in figure 1, while the developmental paths that can produce the traffic lights from initial four-cell patterns are shown in figure 2. Notice that there are twelve translation-defined and three automorphism-defined four-cell patterns that can lead to the traffic lights.

Five-cell patterns give rise to further kinds of oscillators and, more significantly, to the smallest *non*-oscillators. Seven different five-cell automorphism-defined patterns give rise to the *glider*: the smallest and most common GoL spaceship. This translates itself at one cell diagonally in four steps: the maximum speed for a diagonally moving spaceship. Over time, a glider-producing pattern will cause an unbounded number of cells to enter state 1, although all will shortly revert permanently to state 0. The glider's *cumulative cell count* thus grows without bounds, but its *current cell count* and its *diameter* (the maximum number of cells on the shortest path between any two current state 1 cells) do not: these are always five and three, respectively. However, there are nine other five-cell automorphism-defined patterns for which diameter does increase without bound. The best known of these is the *r-pentomino*, and all the rest turn into this pattern after either one or two steps. After 1103 steps, the r-pentomino completes the "interesting" part of its development, producing a clump of oscillators plus six gliders, receding from the clump in three of the four possible directions. The seven five-cell patterns that are or become a glider, and the nine that are or become an r-pentomino, are shown in figure 3.

The smallest patterns to give rise to spaceships other than the glider have eight cells. One of these is the *lightweight spaceship*, shown at left in figure 4, while others produce either this spaceship (alone or with a clump of oscillators), the *middleweight spaceship* or *heavyweight spaceship* (also shown in fig. 4) plus a clump of oscillators, or in one case two lightweight spaceships traveling in

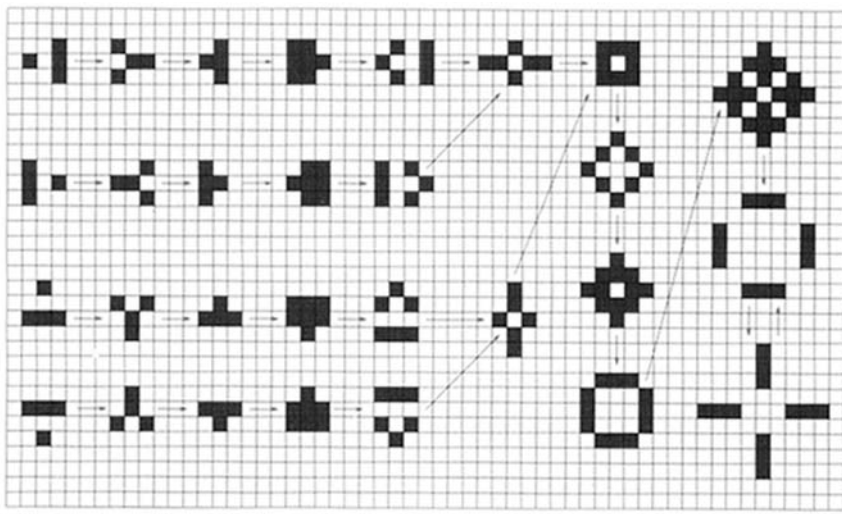

FIGURE 2 "Traffic lights" and their four-cell precursors.

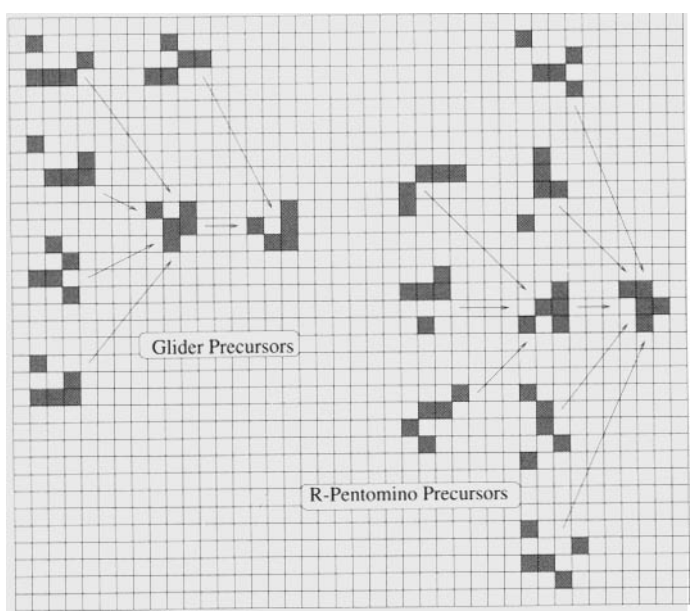

FIGURE 3 The "glider," the "r-pentomino," and their five-cell precursors.

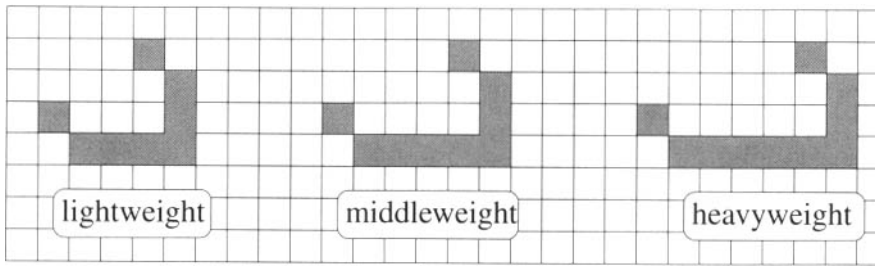

FIGURE 4 The "lightweight," "middleweight," and "heavyweight" spaceships.

opposite directions plus a clump of oscillators. These three spaceships all travel orthogonally at one cell every two steps, the maximum speed for spaceships. The eight-cell pattern sending two lightweight spaceships in opposite directions increases its diameter at one cell per step once the two spaceships have been produced: the maximum sustainable rate for a finite pattern.

Until 1997, the smallest patterns known to show unbounded growth in current cell count contained 11 cells, the first discovered by Charles Corderman [28, pp. 1–2]. Those known all grow into *switch engines*: patterns with a generating "head" that moves diagonally, and an unboundedly growing "tail" of oscillators which develops a spatial periodicity. The head has, in some phases, only eight cells, and by itself will move eight cells diagonally every 96 steps, but produces an additional and unstable clump of state 1 cells, which eventually interacts with and destroys the head. The head can be stabilized by placing a blinker or preblock in various positions, producing the 11-cell patterns mentioned. Two fundamentally different stabilizations are known: the *block-laying switch engine*, the tail of which consists only of blocks once it has attained periodicity, and the *glider-stream switch engine*, which has a more complex tail, and also shoots a stream of gliders ahead of itself. Paul B. Callahan and I showed by exhaustive testing that there are no unbounded growth patterns with fewer than ten cells, and he discovered three 10-cell patterns with this property [14]. Two become block-laying switch engines, and the third, a glider-stream switch engine. All three are included in pattern 1 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩. Figure 5 shows the third of these ten-cell patterns (the two 5-cell clusters enclosed in an irregular curve), and the glider-stream switch engine it produces after 2655 steps (some gliders emitted in various directions early on are not shown).

Exhaustive testing of all possible patterns is straightforward up to five cells; thereafter, it is necessary to consider patterns consisting of two (or, in the case of nine cells, three) separate 1-clusters that may later interact. Once patterns of eight or more cells are considered, one of the clusters may be or become an r-pentomino, and this can send a glider from any distance (the distance between two clusters is the minimum number of empty cells traversed along any neighbor-

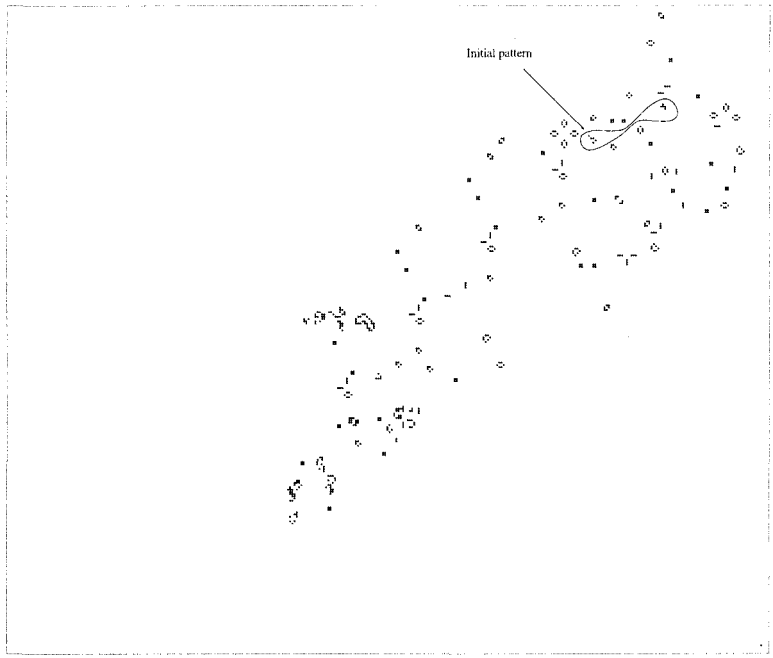

FIGURE 5 Glider-stream switch engine from ten-cell pattern.

hood link path between them) toward the other cluster. If the other cluster is a blinker, it turns out that this reaction can develop in such a way as to send back a glider that interacts with the clump of oscillators produced by the r-pentomino, raising the possibility that simply moving an r-pentomino and blinker further away from each other along a diagonal might generate an infinite set of eight- or nine-cell patterns, all with fundamentally different histories. As it proved, however, both eight- and nine-cell two-cluster patterns involving interaction between an r-pentomino and a distant smaller cluster can be divided into a finite number of classes such that the members of each class develop in the same way, differing only in how long gliders spend on their journeys between the distant clusters.

Just as patterns with bounded current cell count may have a cumulative cell count which grows (linearly) with time, patterns with a linearly growing current cell count can have a quadratically growing cumulative cell count. The smallest known have 16 cells: they consist of two switch engine heads, stabilizing each other in ways that produce "waves" of gliders traveling at right angles to the head. The first known were again discovered by Corderman soon after GoL's discovery [28, p. 3]. Their cumulative cell count grows quadratically as each

glider in a wave follows a different path, and their number grows linearly with time.[2]

The smallest known pattern with quadratic growth in current cell count, which I discovered in 2000, has 52 cells. It is given as pattern 2 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩. It grows in an irregular fashion. Pictures of the pattern and current cell counts after many steps are available from Rokicki [22].

Finally, some patterns "fill space" in the sense that there is an integer n such that, if any cell is selected, there will be a time after which that cell is always within n links of a state 1 cell. The smallest known *spacefillers*, discovered by Tim Coe in 1995, have 187 cells.

3.3 SHORT-TERM, MEDIUM-TERM, AND LONG-TERM EVENTS

The definitions of short-term and long-term events introduced above remain unchanged in the context of sparse random arrays: the "short term" refers to what happens in any specific number of steps, the "long term" to what holds as $t \to \infty$. In sparse arrays, however, the parameter p (or its reciprocal, signified N, which is often more convenient to use) make available an intermediate timescale: we can ask (and in some cases answer) questions about what happens when t is approximately equal to some power of N. For example, consider the smallest (ten cell) clusters with unbounded current cell count growth. When t is approximately equal to N, each such cluster present at $t = 0$ will have grown to a size of around N cells, unless it has interacted with some other cluster. Moreover, it will be seen below that very few of them will have interacted with anything else: only around $\sim p^2$ of them (or 1 in $\sim N^2$ of them[3]) will have done so. In the same way that we can define short-, medium-, and long-term events, we can define local, regional, and global classes of events, and clusters. A class of local clusters or events is one for which there is an integer n such that each instance is contained within a chunk of cells of diameter n: for example, all nine cell 1-clusters, or the process of any such cluster developing into a set of noninteracting repeaters. Regional clusters or events are those with a spatial scale that must be expressed in terms of powers of N. Global clusters or events stretch across the entire array.

[2]There are also ways to construct patterns with bounded current cell count and unbounded but sublinear cumulative cell-count growth, and patterns with linearly growing current cell count and superlinear but subquadratic cumulative cell-count growth.

[3]Powers of p are used in referring to probabilities and densities, powers of N in referring to distances, sizes, and durations. The notation $\sim p$, $\sim N^2$ should be read as "around p," "around N^2." The formula $\mathsf{F}(x) \sim x^a$, for some function F defined over positive values of x, means that $(\mathsf{F}(x)/x^a + x^a/\mathsf{F}(x))/(x^a + x^{-a}) \to 0$ as $x^a + x^{-a} \to \infty$. This covers cases where x is either greater than or less than 1. Related terms used are $\mathsf{F}(x) \ll x^a$, meaning that $\mathsf{F}(x)/x^a \to 0$ as $x^a + x^{-a} \to \infty$, and $\mathsf{F}(x) \gg x^a$, meaning that $x^a/\mathsf{F}(x) \to 0$ as $x^a + x^{-a} \to \infty$.

3.4 THE SHORT TERM

At $t = 0$, the *density* of state 1 cells in the array will of course be p: 1 in N cells will be in state 1. At $t = 1$, the density falls abruptly. It can be calculated, using the transition rule, that for *any* value of p, the exact value of the density in an infinite GoL array, will be $28(3 - p)p^3(1 - p)^5$. As $p \rightarrow 0$, this value approaches $84p^3$. At $t = 2$, when all initial clusters of size 3 have vanished except blinkers and those giving rise to blocks, the density falls to a value approaching $22p^3$ as $p \rightarrow 0$. (For a given cell, there are 16 ways that a cluster of three state 1 cells at $t = 0$ can make that cell part of a block at $t = 2$, and six ways three state 1 cells at $t = 0$ can make it part of a blinker; each of these 22 possibilities has a probability approaching p^3 as $p \rightarrow 0$, and the probabilities of these 22 arrangements are asymptotically independent as $p \rightarrow 0$.) The density will remain arbitrarily close to this value for any specified time if p is sufficiently low. However, we can attempt to calculate the *density contribution* which any class of cluster makes to the total density in the short term (and, as will be seen below, in the medium term).

For example, the traffic lights contain 12 cells (in both phases), and can arise from four cells in 12 ways, so in the short term, from $t = 11$ onward, traffic lights make a density contribution approaching $144p^4$ as $p \rightarrow 0$. The density of traffic lights themselves (the density of cells which occupy, say, the cell in the center of a set of traffic lights) then approaches $12p^4$. The expected distance to the nearest set of traffic lights in any direction appproaches $N^2/\sqrt{12}$, and the expected distance to the cell in the center of a set of traffic lights along an orthogonal or diagonal line, $N^4/12$.

For isolated initial clusters of four or fewer state 1 cells, which constitute the vast majority, all short-term developments are completed by $t = 12$, when the last of those clusters producing traffic lights enter their last novel phase: all those that have not disappeared are still-lifes or period 2 oscillators. Of five-cell clusters, the gliders continue to occupy fresh cells until they meet an obstruction, but the translation-defined clusters produced, repeat in a four-phase cycle. The same does not happen to the r-pentominos and the other five-cell clusters that become r-pentominos at $t = 1$ or $t = 2$: Since these produce gliders going in three directions, the diameter of the cluster produced increases indefinitely, even though the current cell count does not. Nevertheless, by $t = 1105$, the last of them do, in a real sense, complete their development: each thereafter consists of an 18-cluster of oscillators—making up one larger oscillator—and six gliders, three forming a 22-cluster and two a 177-cluster. These parts of the complete cluster will never interact again, so long as no collisions with extraneous clusters occur. This sense of completed development is captured in the following definition of a *quiet cluster*:

A *quiet cluster* (or one that has reached *quiescence*) consists of a set of repeaters, no two of which would ever overlap the same 1-cluster in an otherwise empty array.

The repeaters may move relative to one another, but no two will ever interact. By contrast, an *indefinite growth cluster*, or IGC, would in time exceed any finite bound on its current cell count if it were a pattern; this implies that it would never become quiet. Note that there are clusters which are not IGCs but will never become quiet: it is possible to set up a pattern made of two clusters, one of which uses gliders to push the other further and further away, while the whole pattern remains within a fixed limit on its current cell count.

The smallest IGCs have ten cells. In the short term, their density contribution, although increasing, will therefore remain $\sim p^{10}$, and the expected distance to a state 1 cell belonging to such an IGC, though decreasing, will remain $\sim N^5$. Even if there are IGCs with quadratic growth in current cell count (QGCs) and initial size 10, these statements will also hold for them.

It will be said that two indefinite growth patterns (and by extension clusters) P_1, P_2 are of the same *finite-difference class* if and only if there are integers m, n such that if P_1 is advanced m steps, and then both are advanced synchronously by any number of steps; it will always be possible to change the states of $\leq n$ cells in P_1 so that it is congruent to P_2 (allowing rotations and reflections). Intuitively, the patterns grow indefinitely, but the difference between them does not. All patterns which are not IGCs are in the same finite-difference class.

4 MEDIUM-TERM EVENTS IN SPARSE RANDOM ARRAYS OF CONWAY'S GAME OF LIFE

4.1 ORIGINAL INDEFINITE GROWTH CLUSTERS

An *original cluster* is one derived from a local ancestor of the minimum size for that class (which may be a development class or finite-difference class) at $t = 0$.

After N steps, any original IGCs with linear growth in current cell count will have $\sim N$ cells. The density contribution of switch engines derived from 10-cell initial clusters will be $\sim p^9$, compared to the $\sim p^{10}$ they contributed initially. The expected distance to a state 1 cell in one of these IGCs, however, will still be $\sim N^5$, as their diameter will be $\sim N$, and a distance $\sim N^5$ less a distance $\sim N$ is still $\sim N^5$. The expected distance to such a cell along an orthogonal or diagonal line of cells, on the other hand, will have diminished from $\sim N^{10}$ to $\sim N^9$. This is because if the trail produced by a switch engine (or the glider-stream of the glider-stream switch engine) crosses such a line, there is a probability independent of p that at least one state 1 cell will fall on the line; and if a switch engine begins within N cells of such a line, there is a probability asymptotically independent of p that its trail or glider stream will indeed cross the line.

More generally, so long as nothing interferes with the growth of these switch engines, their density contribution after $\sim N^E$ steps will be $\sim p^{10-E}$, the expected distance to a state 1 cell belonging to one along an orthogonal or diagonal line $\sim N^{10-E}$, but the expected distance to the nearest in any direction would remain $\sim N^5$ until $\sim N^5$ steps had passed, when it would drop to below $\sim N^x$ for any positive x. Before this, however, the switch engines would have interacted with other, initially distant clusters. If there are 10-cell QGCs, their density contribution would reach p^{10-2E} after $\sim N^E$ steps, but the expected distance to the nearest such cluster along a line would be $\sim N^{10-E}$, as for a switch engine: this quantity is affected by the diameter of a cluster rather than its cell count (so long as the cluster forms a d-cluster for some value of d independent of p; i.e., its state 1 cells are not separated from their nearest cluster mates by increasingly large gaps as time goes on).

4.2 INTERACTIONS BETWEEN INITIALLY DISTANT CLUSTERS

However low the value of p, some pairs of initially distant clusters will eventually interact. In this subsection, local clusters of different sizes and dynamic properties, particularly the minimum initial size clusters with various properties, will be considered from this point of view.

First, consider the (non-null) clusters that can develop from those of initial size three: blinkers and blocks (collectively *blonks* from here on). Since these will not interact with each other if initially distant, they simply wait for some moving or growing cluster to encounter them. The same is true of all clusters of initial size four, but at initial size five, the first possibilities for interaction appear.

Consider what will happen to an original glider in an infinite sparse random GoL array. After $\sim N$ steps (this will also be expressed by saying "in era 1," and similarly "after $\sim N^x$ steps," for any positive x, may be expressed as "in era x"), $\sim p^2$ of the gliders in the initial array will have collided with a blonk; $\sim p^3$ of them will have collided with something else instead. To see this, consider what lies in the path of a glider in the initial array. In order to interact with a glider, another cluster must occupy a cell sufficiently close to the glider's path, at an appropriate time. In a sparse random array, the first such object will typically be a blonk at a distance of $\sim N^3$ cells (the blonk is twice as likely to be a block as a blinker). This might suggest that interactions between original gliders (and r-pentominos) and original blonks will become significant in era 3. In fact such interactions will become significant before that era, as argued below, but setting that aside, consider what can happen when a lone glider interacts with a block or blinker.

Specificially consider a glider headed "south-east" ("SE"): toward the lower right of a screen display.[4] If all cells which enter state 1 as the glider moves are marked (the glider's *cumulative image*), the result is a diagonal swathe made up

[4]This will be the default heading for gliders in this chapter. Moreover, it will be assumed that coordinates on the x-axis (east-west) increase to the east, and those on the y-axis (north-

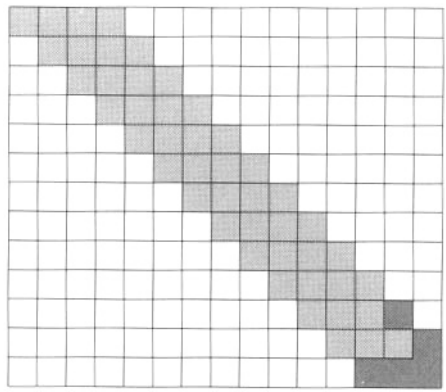

FIGURE 6 The glider and its cumulative image.

of four strips of cells (see fig. 6). Within each strip, or *half-diagonal*, adjacent cells meet at a corner. It can be seen that if the glider were displaced by one cell NE, the path generated would shift by *two* half-diagonals; hence the name. The paths of two *parallel gliders* (gliders moving in the same direction), if not identical, can differ by a minimum of one half-diagonal: consider shifting the glider in figure 6 one cell east.

Now consider an east-west row of cells crossing the glider's path. In order for the glider to interact with a block that has its northern cells falling on that row, the block must occupy at least one cell no more than three half-diagonals from where the edge of the glider's cumulative image will run. This means it must be in one of 12 east-west positions (see fig. 7). The resulting collisions, for a SE-headed glider, will be referred to as *glider/block 1* through *glider/block 12* (higher-numbered collisions having the block further east). In fact, glider/block n and glider/block $13 - n$, for $1 \leq n \leq 6$, are mirror images. Glider/block 1 produces a shifted block, 2 and 3 produce clumps of oscillators, 4, 5, and 6 result in the elimination of both glider and block.

In the blinker's case, matters are complicated by its period two oscillation between horizontal and vertical. Again, there are 12 possible east-west positions for the blinker, once the three adjacent rows in which it occupies cells in its vertical phase are determined. However, since the glider has a four-phase cycle, either the horizontal or the vertical phase of the blinker can coincide with a given phase of the glider. The collisions where the horizontal blinker phase coincides with the glider phase shown in figure 7 will be labeled *glider/blinker 1* (with the blinker at the westernmost location) to *glider/blinker 12* (blinker at the easternmost location), while the others will be *glider/blinker 12b* to *glider/blinker 1b*. The "b"

south) increase to the *south*, contrary to the usual mathematical convention, but in line with those in the program recommended for the demonstration patterns.

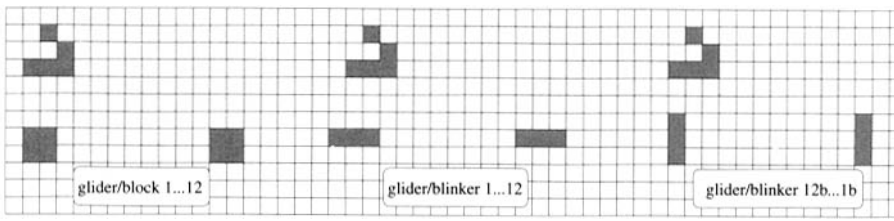

FIGURE 7 Possible glider/block and glider/blinker collisions.

here can be taken to mean "backward": while all of the collisions glider/blinker 1 through glider/blinker 12 are distinct, glider/blinker nb, with $1 \leq n \leq 12$ is a mirror image of glider/blinker n. Glider/blinker 4 and glider/blinker 6 both generate new gliders: one going NW and one SW from glider/blinker 4, one NW, two SE, and two NE from glider/blinker 6.

The two going SE ("onward") from glider/blinker 6 both follow the same path, 52 half-diagonals to the NE of the incoming glider. The first is 1866 steps ahead of the other: that is, the trailing glider will occupy at step $t+1866$ exactly the cells occupied by the leader at step t. Such onward pairs of gliders from glider/blinker 6, *1866-pairs*, will be significant in what follows.

Gliders emitted by a collision between an original glider and blinker (or between a glider or *fleet* of gliders from an original r-pentomino, and an original blonk), may collide with further original blonks, and in some cases emit more gliders. (A *fleet* is a finite set of spaceships with a common velocity far enough apart to travel without mutual interference, that would form a cluster if everything other than spaceships with that velocity were removed from the array. A *subfleet* is any non-null set of spaceships forming a part, not necessarily a proper part, of a fleet.) Gliders emitted from later collisions in a direction opposite to that of the incoming glider(s) may also hit oscillators left over from previous collisions, or from an original r-pentomino. A *standard collision sequence* or *SCS* is a sequence of collisions involving a single original glider or r-pentomino, and one or more original blonks distant from it and from each other. There are constraints on the relationships between the collisions, described below.

Each collision in an SCS involves at least one *SCS fleet*, and may involve an *SCS oscillator*. The original glider, or the three fleets emitted in a different direction by the original r-pentomino, are the first SCS fleet(s); the original blonks, and the compound oscillator produced by the original r-pentomino, if present, are the first SCS oscillators. A collision in an SCS may be *primary*—involving a single SCS fleet and an original blonk, or *secondary*—not involving an original blonk. Either type may produce nothing, or an SCS oscillator (a non-null, possibly compound oscillator produced by the collision), and/or one or

more SCS fleets (each such fleet consisting of all the spaceships that leave the scene of the collision with a given velocity).

An SCS ends if:

1. The collision product is a cluster, such as an IGC, that would never become quiet in an otherwise empty array.
2. The collision product interacts with anything else before becoming quiescent.
3. A secondary collision S occurs which would have happened differently, or not at all, if the original blonk involved in any primary collision P had been shifted, parallel to the path of the SCS fleet that hit that blonk, by an amount which would delay P by the least common multiple of the translation periods of all the SCS oscillators and SCS fleets generated prior to S. This condition serves to rule out collision sequences that depend on special coincidences of timing. For example, two SCS fleets from a collision A might both collide with original blonks and send SCS fleets back toward the collision site. For the two to arrive close together in time would be, in a sparse array, an unlikely coincidence, dependent on the precise spatial relationships between A and the two blonks. Such an event, and many more complicated ones involving similar coincidences, would end the SCS by this condition.

Defining a *standard* collision sequence correctly suggests there are also non-standard ones. To understand the name, and to see why collision sequences become significant before era 3, consider once again the possible fates of an original glider in an infinite sparse random array.

The first obstacle in the path of most original gliders is an original blonk $\sim N^3$ links distant. However, in a proportion $\sim p$ of cases, there is a blonk at a distance N^2 or nearer, and in a proportion $\sim p^2$, one at N or nearer. Furthermore, in $\sim p$ of the total, something other than a blonk (almost always, one of the oscillators derivable from an original cluster of four cells) is the first object in the glider's path, and in $\sim p^3$ of the total cases, such a cluster is at a distance N or less. The same considerations apply to the gliders emitted by an original r-pentomino.

Of the original gliders that hit an original blonk after $\sim N^2$ steps, a proportion asymptotically independent of p and $\gg p$ (specifically, $\rightarrow 1/18$ as $p \rightarrow 0$) will meet a blinker in a collision that emits more gliders, and in $\sim p$ of those cases, at least one of these gliders will hit another blonk in a further $\sim N^2$ steps. Since two lots of $\sim N^2$ steps sum to $\sim N^2$ steps, we can say that in era 2, $\sim p^2$ of all original gliders will have originated an SCS involving two original blonks. Extending the reasoning, $\sim p^n$ of them will have originated an SCS involving n original blonks. The same is true of original r-pentominos: the fact that each emits six gliders in three directions does not affect the exponent of p in these expressions.

Notice that, for three reasons, such an SCS may involve more than n instances of a glider crashing into something in its path. First, two or more gliders may be traveling in a fleet: for example, the fleets of two and three gliders generated by the r-pentomino, or the two 2-glider fleets (the 1866-pair and the two gliders emitted NE) from glider/blinker 6. In such a case, one member of the fleet may hit the cluster produced by another member hitting something—and this may occur while the latter is still developing, or after it has become quiet. In either case, the entire event counts as a single collision. Second, as noted above, gliders emitted backward may hit oscillator clusters already generated in the course of the SCS. Third, gliders from different collisions may encounter each other in ways that do not depend on special coincidences of timing (for example, a glider each from the SE and NE fleets from a glider/blinker 6 can take part in further collisions of the same kind, giving rise to NE-headed and SE-headed gliders which collide).

As long as the probability per step of a glider encountering an original blonk remains at $\sim p^3$, and most original gliders and r-pentominos have encountered no obstacles, the proportion that will have taken part in SCSs involving b original blonks (*order b* SCSs) by era E is $\sim \left(p^{3-E}\right)^b$, or $\sim p^{b(3-E)}$. The density of points in the array where the last collision in an order b SCS has occurred (the *cumulative occurrence density* of such events) will therefore be $\sim p^{5+b(3-E)}$. *Nonstandard collision sequences*, beginning with an original cluster of six or more cells, or involving the $\sim p$ chance of a fleet encountering an original cluster of more than three cells, will have a cumulative occurrence density $\ll p^6$.

Figure 8 illustrates how the cumulative occurrence density of various classes of collision sequences increases over time. Both time and density are logarithmically scaled. The three bold lines, labeled "**5;3**," "**5;3;3**," and "**5;3;3;3**" show the cumulative occurrence densities of SCSs of orders one, two, and three (the labels indicate the sizes of the original clusters taking part in each type of sequence). Similarly, the line parallel to and just below the "**5;3**" line shows the cumulative occurrence density of two classes of collision sequence: those beginning with a six-cell cluster (there are several of these which emit one or more gliders) and involving one original blonk, and those beginning with an original glider or r-pentomino and involving a single cluster which began with four cells. The line labeled "Three: sum 13" shows the cumulative occurrence density of collision sequences involving three original clusters with sizes summing to 13, e.g., "7;3;3" and "5;4;4." The change to broken lines at era 14/5 is explained below.

The switch engines growing from original ten-cell clusters have roughly the same expected distance to an obstacle in their path—$\sim N^3$ links—as a glider does. Unlike glider fleets, however, switch engines leave persistent trails of oscillators, and these may be struck by gliders traveling perpendicular to the trail. The proportion of original gliders involved in such collisions will be insignificant, but once an original switch engine reaches diameter and size $\sim N^{5/2}$, in era 5/2,

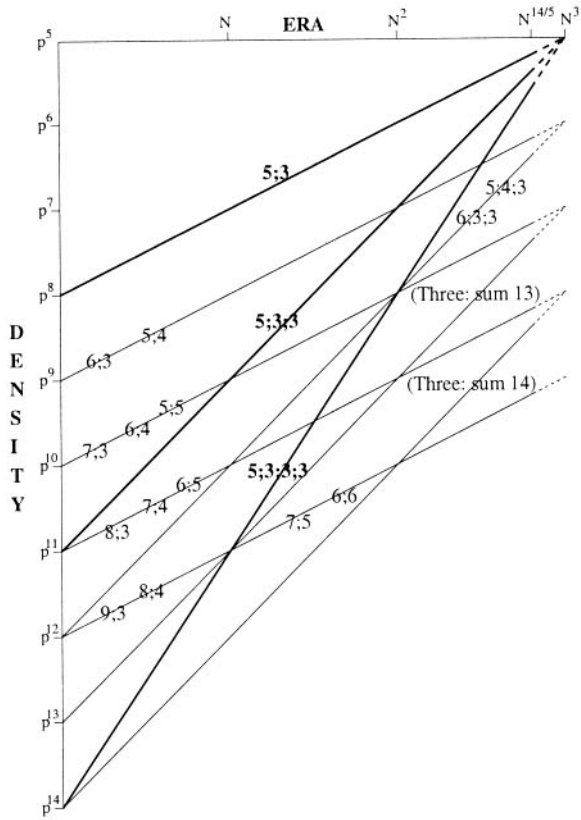

FIGURE 8 Collision sequence frequency to era 3.

its probability per step of being struck by such a glider will reach $\sim p^{5/2}$, and, therefore, one such collision can be expected in $\sim N^{5/2}$ steps. As long as most original gliders have not collided with anything, the nearest original glider moving toward the head of a switch engine as it grows past a particular point will usually be $\sim N^5$ links away, but in $\sim p^{5/2}$ cases it will be close enough to reach the switch engine's trail after no more than $N^{5/2}$ steps. The glider-stream switch engine's growth can be halted by a single glider hitting the forward glider stream and creating an obstruction the head crashes into. A similar collision creates an obstruction to the glider stream, found by Tim Coe, that grows backwards and then causes the switch engine to produce a wave of gliders as the head passes it. The block-laying switch engine cannot be killed by a single glider, or fleet from an r-pentomino, but may be vulnerable to multiple collisions of this kind. If so, both kinds of original switch engines will generally "die" in era 5/2. It is conceivable that some sequence of collisions with the trail could "ignite" it and

cause it to disappear completely. Finally, such bombardment by gliders could cause switch engines to "reproduce" (catalyze the production of more switch engines), or otherwise acquire a superlinear growth rate, at size $\sim N^{5/2}$ cells.

Original local clusters with quadratic growth in their cumulative image (whether or not this also occurs in their current cell count), will typically interact with their environment in era 3/2, when the growing cluster will have neighbored $\sim N^3$ cells. Such a pattern would by then be impinging on $\sim N^{3/2}$ previously unvisited cells per step, and could expect to encounter an original blonk once in $\sim N^{3/2}$ steps. There are some quadratic growth patterns (such as the 187-cell spacefiller mentioned) that would be "killed" by any single encounter of this kind. Others appear less vulnerable, but none have been proved to survive unlimited numbers of encounters with isolated blonks. No such encounter could lead to a faster than quadratic growth rate.

5 STANDARD COLLISION SEQUENCE CONSTRUCTIONS

Figure 8 suggests that in the analysis of medium-term events in sparse random GoL arrays, SCSs are likely to be of central importance, although original IGCs and nonstandard collision sequences must also be considered. In investigating SCSs, and, in particular, the question of what will be constructed by such processes before era 3, three approaches suggest themselves:

1. Exhaustive surveys of some class of SCSs—most obviously, all those of order n or less.
2. Attempts to design SCSs which will produce particular kinds of cluster—most obviously, IGCs.
3. Attempts to construct proofs that any member of some broad class of cluster or event can result from an SCS.

All these approaches have been tried, each with some degree of success. The first two are dealt with in this section; the third, at considerably greater length, in section 6.

5.1 SURVEY OF ALL STANDARD COLLISION SEQUENCE CONSTRUCTIONS OF ORDER 1 AND ORDER 2

The exhaustive survey approach has not been taken very far, but has nevertheless resulted in some progress. In order to carry out a survey of all the SCSs of order 1, it was enough to consider all the ways in which the three fleets emitted by the r-pentomino (fleets of three and two gliders, and the minimal one-glider fleet) could encounter a block or blinker at sufficient distance from the r-pentomino that further increases in distance would make no fundamental difference. (Even if the collisions involving an original lone glider had not already been investigated,

the checks on the single glider fleet from the r-pentomino would cover all the possible cases.)

For each of the six gliders in the r-pentomino fleets, 36 distinct collisions had to be considered: 12 with blocks and 24 with blinkers. (The number would have been less if the members of the 2-glider or 3-glider fleets were sufficiently close for one to preempt some of the collision possibilities involving another.) Although the 36 constitute 18 mirror-image pairs so far as the glider/blonk collisions themselves are concerned, the existence of the other gliders and the oscillator produced by the r-pentomino meant it could not be assumed the members of these pairs are equivalent, and in some cases they are not. In some cases involving the 3-glider fleet, a second glider collides with the result of the first collision. In the majority of the 216 cases, the collision emits no gliders. In only one case does a glider from the collision, in hitting the r-pentomino remnant, generate further gliders, and these do not return toward the collision remnant.

The order 2 SCSs required considerably more effort, although once again those sequences beginning with a lone glider did not require separate consideration. A full description of the structure of the set of possibilities will not be attempted here, but it is worth noting that even if the first collision merely produces a small oscillator as many glider/blonk collisions do, the possibility that gliders emitted from the second collision might interact with this had to be considered. So did the possibility, already noted above, that gliders from the two collisions might collide at a location remote from both the collisions generating them. Also, if the two collisions involve gliders from the r-pentomino headed in different directions, and both send gliders back toward the r-pentomino's oscillators, the result might depend on which arrives first. (It could be assumed that they would not arrive close together in time: this would require an unlikely coincidence in the distances of the two blonks from the r-pentomino, making the collision sequence nonstandard: such a coincidence, in era E, decreases a sequence's probability by a factor of $\sim p^E$.)

The completed survey showed that no order 1 or 2 SCS gives rise to an IGC. This has implications for the density of state 1 cells in the medium term. If a QGC can be produced by an SCS, the SCS must be of at least order 3. As noted, the cumulative occurrence density of events completing SCSs of order b or higher by era E (where $E < 3$) will be $\sim p^{5+b(3-E)}$. Substituting 3 for b gives $\sim p^{14-3E}$. At most (if it produced a QGC that continued to grow quadratically), each of these events could have produced $\sim N^{2E}$ state 1 cells, giving a maximum total density contribution of $\sim p^{14-5E}$. This reaches p^3 in era 11/5, and 1 in era 14/5, so it can be concluded that SCSs cannot possibly produce a density contribution exceeding that of original blonks until the first of these eras, while at any time before the second, there will always be a positive x such that SCSs' density contribution is $\ll p^x$. By a parallel argument, SCSs cannot reduce to $\ll N^3$ the expected distance along an orthogonal or diagonal line to a cluster produced by such processes, until after era 11/4 (the minimum this expected distance could

be in era E is $\sim N^{5+b(3-E)-E}$). SCSs cannot reduce this expected distance to one comparable with the distance an original glider will have traveled before era 14/5, the era in which $N^{5+3(3-E)-E} = N^E$. From this era onward, but not before, it is possible that the rise in cumulative occurrence density of SCSs of each order could be affected by collisions with the products of IGCs already produced by SCSs.

The conclusion concerning the earliest era at which the density of state 1 cells could be $\gg p^3$, but not the others, might be affected by nonstandard collision sequences. The cumulative occurrence density of the completion of any class of nonstandard collision sequence $\ll p^6$ at any time before era 3. Hence, the density contribution due to QGCs produced in such a way would always, before era 3, be $\ll p^x$ for some positive x, and the expected distance to any cluster produced by such a process would, before era 3, always be $\gg N^3$.

The comprehensive survey of order 2 SCSs also had an interesting positive result: an order 2 SCS that produces a lightweight spaceship. Pattern 3 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩ shows this order 2 SCS (the r-pentomino and the two blinkers could be arbitrarily distant from each other along the NE/SW diagonal). The existence of this SCS means that in era 3/2, the number of SCS-produced lightweight spaceships will be comparable to the number of original lightweight spaceships, and in later eras (at least prior to era 14/5) will exceed it by a factor that grows without limit as $p \to 0$.

5.2 AN ORDER 49 STANDARD COLLISION SEQUENCE PRODUCING A SWITCH ENGINE

An order 49 SCS producing a block-laying switch engine was discovered by the author in 2001. One starting point for the discovery was an indefinitely extensible SCS called the *lucky rake*, described in the next section, which allows fleets exceeding any specified number of gliders to be constructed. The second was Callahan's discovery of a considerable number of ways to generate switch engines by colliding small, precisely timed fleets with a single block or blinker. Pattern 4 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩ illustrates the construction. It begins with an r-pentomino and 49 blonks, strung out along a NW-SE diagonal (the distances between clusters along this diagonal could be increased indefinitely). Gliders, sent out NW and SE from the r-pentomino, interact with the blonks in an SCS to produce a block-laying switch engine headed NW. This would, in time, collide with an earlier collision remnant, but switch engines produced in this way could still have an important effect on the medium-term dynamics of sparse GoL arrays, particularly if it turned out there were no shorter SCSs that produce IGCs (although this seems unlikely to be the case).

6 WELL-SPACED GLIDER COLLISIONS AND STANDARD COLLISION SEQUENCES

6.1 WELL-SPACED GLIDER FLEETS

For two gliders that are on the same path, one must be at least 14 steps ahead of the other if the two are not to interfere destructively. Visually, it may be clear that one of a parallel pair of gliders is ahead of the other even if their paths are some way apart. The number of steps one such glider is ahead of or behind another can be determined exactly by considering the half-diagonals perpendicular to their paths. As a glider moves, every alternate step involves the occupation of a cell in a new half-diagonal; the intermediate steps increase the number of occupied cells in the most recently occupied half-diagonal to two. Whether two parallel gliders' paths are the same, or are separated by an even or an odd number of half-diagonals, we can, therefore, compare the perpendicular half-diagonal each has most recently occupied (the most south-easterly half-diagonal reached if the gliders are going SE), check whether each glider occupies one or two cells in that half-diagonal, and, thence, calculate which leads the other and by how many steps.

For some purposes, it is also useful to consider which of two parallel gliders is "leading" the other, and by how many steps in a direction at 45° to that of their path—i.e., which has traveled either further south or further east in the case of SE-moving gliders. The same method can be used as when assessing the gliders' relative position in the direction of movement: As it travels, a glider occupies a cell in a new row (or column) on every second step, and increases the number of cells occupied in the most recently reached row (column) on intervening steps.

For a pair of parallel gliders of which neither leads the other, the minimum difference in paths is seven half-diagonals. As can be seen in figure 9, this separation can be combined with the minimum forward separation of 14 steps for two gliders on the same path to give a closely spaced fleets of parallel gliders of any desired size. Such fleets, of every possible finite number and spacing of gliders, will of course exist in any random GoL array, and will persist indefinitely, as there will be instances of every possible type surrounded by arbitrarily large empty regions in the initial array. All such fleets are repeaters, and, hence, are ω-constructable patterns.

However, it can also be asked how fleets will be produced from predecessors other than translations of themselves at times after $t = 0$. Subfleets which cannot be produced in any other way, if there are any, will decline monotonically in density in any random array as they encounter obstacles (the same cannot be said of fleets, as any fleet can be produced from a larger one by collision with a block which removes a single glider). In the context of sparse random arrays, it is natural to ask in particular what fleets are *SCS-constructable*: i.e., can come into existence as the product of an SCS (not necessarily as the sole product, but forming a local cluster that can take part in continuing the SCS). Taking

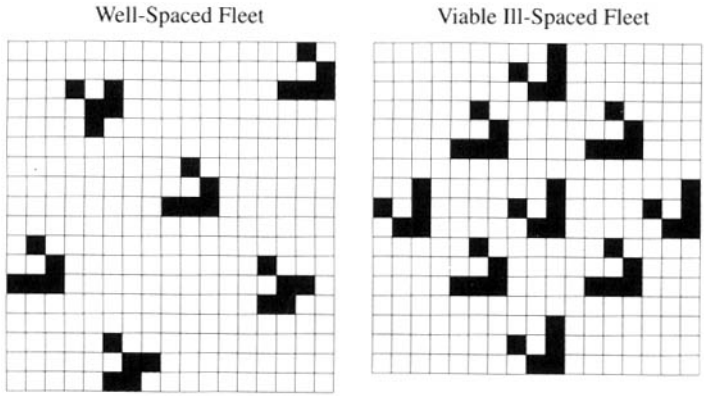

FIGURE 9 Well-spaced and ill-spaced fleets.

this idea a stage further, we can ask what *collisions* between glider fleets are SCS-constructable.

No fleet or fleet collision has been shown to be nonconstructable by a standard collision sequence. A proof is given here—an outline is also published in Gotts [13]—that all member of a large class of *well-spaced fleets*, and all member of a class of collisions between well-spaced fleets, are SCS-constructable.

A well-spaced fleet is defined as one in which any pair of gliders either have paths that differ by at least 12 half-diagonals, or have one member of the pair at least 19 steps ahead of the other. The significance of this class of fleets is simply that it is the largest class for which detailed SCS-construction methods have been worked out. The fleet shown at the left of figure 9 is well spaced, while that shown at right is not, although it can travel without interference between the gliders.

A *well-spaced glider collision* is one which begins with between two and four well-spaced fleets located in separate quadrants of a rectangular part of the GoL array. More precisely, take any rectangular chunk of cells, and mark two adjacent rows and two adjacent columns of cells, forming an orthogonal cross dividing the remainder of the chunk into four quadrants. Each of these quadrants is either empty, or contains a single well-spaced fleet. If there is a fleet in the NW quadrant, its gliders are heading SE and, conversely, any gliders in the SE quadrant are headed NW. The NE quadrant contains nothing or a SW-headed fleet, and the SW quadrant nothing or a NE-headed fleet. Any such arrangement of gliders has an infinite set of predecessors which meet the same conditions, and if any of these predecessors is SCS-constructable, the arrangement of gliders and all its successors are SCS-constructable. Any such arrangement of glider fleets, together with its subsequent history (so long as nothing else interferes) will be

called a well-spaced glider collision. It will be shown that any cluster constituting any stage of a well-spaced glider collision is SCS-constructable.

6.2 CONSTRUCTING AN ARBITRARY WELL-SPACED GLIDER COLLISION

The proof has three main stages:

1. Any member of a subset of the set of well-spaced fleets, *spaced-out fleets*, is SCS-constructable.
2. Any member of the set of well-spaced fleets can be constructed from a finite sequence of collisions between some spaced-out fleet (or a rotation of such a fleet), and some set of arbitrarily widely spaced blonks. Hence, any member of the set of well-spaced fleets is SCS-constructable.
3. Any well-spaced glider collision can be produced by a collision between some well-spaced glider fleet and a single blinker.

The proof inevitably relies on the properties of a number of specific collisions between gliders or glider fleets on the one hand, and blocks, blinkers, other small oscillators and other gliders on the other. Those properties of each such collision that are important in the proof will be identified, and the collisions themselves will be described in sufficient detail for any reader with access to a good GoL simulation program to find or reproduce them. Readers may wish to begin by reading only the initial paragraphs of the main stages and first-level substages, following the proof in full detail later if at all.

1. SCSs can construct any SE-headed fleet in which each pair of gliders are on paths at least 82 half-diagonals apart, and no glider is both on a path to the SW of another, and (even a single step) behind the other in its west-to-east progress. Such a fleet is a spaced-out glider fleet.

   The definition of a spaced-out glider fleet implies a specific kind of (not necessarily strict) monotonicity in the placing of the gliders. Taking the glider on the path furthest to the NE (call it glider 1), call the most easterly column of cells it has reached column C_1. As noted above, the glider may occupy either one or two cells in that column. Moving to the glider on the path nearest to that of glider 1 (glider 2), the most easterly column of cells it has reached (column C_2), will be either the same as column C_1, or to its east. If columns C_1 and C_2 are the same, glider 1 can occupy two cells in that column only if glider 2 also does so. Numbering the gliders in the fleet in order of their paths from NE to SW, an analogous relation will hold between gliders i and $i+1$. Hence, an analogous relation will also hold between gliders i and j whenever $i < j$. Glider $i+1$ will, as a consequence of the definition, also be at least 328 steps (82 rows) south of glider i.

Another kind of monotonicity, equivalent under reflection about a NW-SE axis, could be ensured by specifying that no glider is on a path NE of another, and even a single step behind that other in its north-south progress. The choice between the two is arbitrary.

a. There is a sequence of collisions, known as the *lucky rake*,[5] beginning with a collision between a single glider and a blinker, which can generate a well-spaced fleet exceeding any desired number of gliders. The initial glider is headed NE, and meets a blinker in an inverted version of the collision "glider/blinker 6" (collision 1). The two gliders which emerge perpendicular to the incoming glider's path are headed SE. The paths of this SE-headed pair are 53 half-diagonals apart. The leading member of this pair (the path of which is further SW than its partner) enters collision 2, a glider/blinker 6 collision. The two SE-headed gliders from this collision have a path 52 half-diagonals NE of the incoming glider (hence in this context the glider/blinker 6 and glider/blinker 6b collisions may be called *52-shifts*), and thus one half-diagonal SW of the second glider from collision 1. The delay due to collision 2 puts both of the 1866-pair behind the collision 1 glider.

   Collision 3 initially involves the glider from collision 1 and a block in the glider/block 11 collision, creating a *honey farm* (a highly symmetrical grouping of four beehives). The first of the pair from collision 2 collides with the honey farm in a reaction that creates nine gliders, three of which head SE; the second of the pair from collision 2 hits the debris from this collision, creating no more gliders.

   Of the three SE-headed gliders from collision 3, the leader has a path between the other two: the second glider's path is 30 half-diagonals SW of the leader's, the third's 29 NE of the leader's. The leader enters collision 4, another glider/blinker 6 collision, with the other two gliders both interacting with the collision product before it has reached quiescence. Collision 4 again produces three SE gliders (plus two others). Two of the three have paths differing by a single half-diagonal, with the leader on the path to the NE of the other, while the third is on a path 70 to the NE of the leader. This is the first of the SE-headed gliders produced by the lucky rake that is unaffected by further collisions in the sequence. The other two, which can be regarded as the engine of the lucky rake, resemble the first two onward gliders from collision 2 in the difference between

[5] A *rake* in GoL parlance is a type of IGC, a coherent object moving like a spaceship but producing an evenly spaced sequence of gliders or other spaceships, each following its own path. The coherent object is often called the *engine*. The lucky rake behaves in this way, but only as long as the moving cluster happens to collide with blocks and blinkers in a particular sequence. In a sparse random array there is no limit to how long its run of luck may continue, but no guarantee that it will continue at any point.

their paths, although the leader is further ahead, and the follower is not the first of a 1866-pair.

From this point on, each pair of collisions produces an additional "tooth" of the rake, and shifts the path of the engine by 82 half-diagonals NE.[6] Collision 5 resembles collision 3, except that there is a longer gap between the creation of the honey farm and the arrival of the second glider, and that the third member of the fleet passes by the collision remnant. Collision 6 is exactly like collision 4, creating a second "tooth" for the rake, and collision 7 exactly like collision 5. A pair of collisions such as collisions 5 and 6 will be called an *82-shift*. Each "tooth" glider is 82 half-diagonals NE of its predecessor, and 1279 steps behind it. This places it 1115 steps west and 1443 steps north of its predecessor.

b. The lucky rake collision sequence can be modified by introducing subsequences of eight collisions which shift the engine by 229 half-diagonals NE—hence, these subsequences are *229-shifts*—and delay it by 3728 steps, producing the same lateral gap and delay between adjacent "teeth" of the rake.

The first two collisions of a 229-shift, in fact, constitute an 82-shift, but the third removes the new "tooth" glider. This can be achieved by any of a number of collisions, glider/block 5 being one possibility; if it is not done, the glider interferes with the process of shifting the engine. The fourth collision is just like the first collision in an 82-shift, producing the same engine subfleet of three gliders, but the fifth involves the southwest-most of these three alone, in a northeast-wards 52-shift. This places a 1866-pair on a path between the other two members of the engine subfleet. The sixth collision involves the leading member of the subfleet in a glider/blinker 4 collision. The remaining three members of the engine subfleet crash into the remnant, producing just two onward gliders, with the leader one half-diagonal NE of the other. The seventh and eighth collisions are again effectively the same as those of an 82-shift, the only difference being a shorter delay between the members of the engine pair as they enter the seventh.

The effect of introducing 229-shifts is to produce a *multipart lucky rake*. The first member of the second and each subsequent part is separated from the last member of the preceding part by 229 half-diagonals and 3728 forward steps. Pattern 5 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩ is a demonstration of the first ten collisions in the

[6]The gliders constituting the engine at any point are destroyed in the next collision entered, and the engine itself has different forms after odd-numbered and even-numbered collisions. It is often convenient to refer to gliders and fleets as having an identity that survives such events, when there is continuity in terms of relative position within a larger fleet and/or functional role in a construction.

basic lucky rake sequence, followed by the eight collisions of a 229-shift (the first two collisions in this being the same as those which would occur in a further continuation of the basic lucky rake sequence), four further collisions in the construction of the second part of the rake, and finally, the removal of the engine, which is done using collision glider/block 12.

c. Once the engine is removed from a multipart lucky rake, what remains is a spaced-out fleet. Individual gliders can then be removed (using glider/block 5), leaving at least one glider from every third part of the multipart rake (this is not the only or necessarily the most efficient possible way for SCS-construction of most spaced-out fleets to proceed, but makes for relatively easy exposition).

   An adjacent pair of gliders in the resulting spaced-out fleet will then have paths separated by $l = 82a$ with that on the more southwesterly path $f = 1279a$ steps ahead of the other (where a is a positive integer), or paths separated by $l = 82a+229$, $l = 82a+458$, or $l = 82a+687$ half-diagonals, with that to the SW of the other $f = 1279a+3728$, $f = 1279a+7456$, or $f = 1279a+11184$ steps ahead, respectively (a a nonnegative integer). In all these cases, $a \equiv f$ modulo 2. If $l = 82a$ and $f = 1279a$, then l can have only even residues modulo 52, and f is even if and only if $l \equiv 0$ modulo 4. Having two 229-shifts between a pair of adjacent surviving gliders gives values for f and l such that f is even if and only if $l \equiv 2$ modulo 4, having one or three 229-shifts between a pair of adjacent surviving gliders makes it possible to combine all the *odd* residues modulo 52 for l with either odd or even f. Adding multiples of 52 82-shifts leaves the relevant residues unchanged while allowing f to exceed any desired value. By extension, constructing a multipart lucky rake and then removing gliders from it can produce a spaced-out fleet in which each adjacent pair of gliders has any chosen path-difference modulo 52, and a difference of either an odd or even number of steps in the direction of travel, this difference also being as large as may be desired.

d. The collision glider/blinker 6b, which produces a pair of gliders going in the same direction as the incoming glider, both on the same path, 52 half-diagonals SW from the incoming glider's path, can be used to reduce any of the differences in glider paths by multiples of 52 half-diagonals.

   At the end of this stage (which we will assume to take place, involving a single 52-shift for each glider, even if none of the path differences require reduction—in order to dovetail with stage 1e), the fleet will no longer be a spaced-out fleet of gliders, but can be called a spaced-out fleet of 1866-pairs. In such a fleet, the 1866-pairs are headed SE, the paths of any two 1866-pairs are separated by at least 82 half-diagonals, and if pair x is

on a path SW of pair y, the *front* member of pair y is not even one step further east than the *back* member of pair x.

In addition to the onward 1866-pair, glider/blinker 6 or 6b produces a clump of oscillators, and three additional gliders. One goes NW. The other two go SW if the onward gliders' path is shifted 52 SW relative to the incoming glider (glider/blinker 6b); otherwise, they go NE. This stage of the construction procedure, and the next, require that the additional gliders and the oscillators produced by a 52-shift do not interact with other gliders in the developing fleet.

In this stage, that noninteraction can be ensured for a g-glider fleet by the following procedure:

(1) Put the leading glider (which will be called glider g) through glider-blinker 6b, resulting in a 1866-pair 52 SW of the original (this ensures the fleet consists entirely of 1866-pairs at the end of the stage).

(2) Put each of the remaining gliders through the same process, beginning with the one with the path nearest the leader (glider $g-1$), and working northeastward. (At this point, all gliders in the original fleet have been replaced by 1866-pairs, and if no path-difference reduction was in fact required, the process can halt.)

(3) Apply a SW 52-shift to 1866-pairs $g-1$ through 1 in order, as many times as required to produce the desired difference in paths between 1866-pair g and 1866-pair $g-1$ (any number of half-diagonals ≥ 82). Note that when a 1866-pair is 52-shifted by letting the leading glider collide with a blinker in glider/blinker 6 or 6b, the second member of the pair mutually annihilates with a block left by the initial glider/blinker 6 collision before the result of that collision has reached quiescence, but makes no further difference to the result.

(4) Assuming the original fleet contained at least three gliders, reduce the path difference between 1866-pair $g-1$ and the remaining pairs in the same way (shifting all the 1866-pairs from $g-2$ to 1 in turn), continuing until the gap between 1866-pairs $g-1$ and $g-2$ is as desired. Apply the same procedure to adjust each remaining path-difference, ending with that between 1866-pairs 2 and 1.

If the differences in the forward direction were sufficiently large (which, as stage 1c showed, can be ensured), the result will be a spaced-out 1866-pair fleet.

Pattern 6 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩ shows that if two SE-headed gliders are on paths at least 82 half-diagonals apart, and that on the path to the SW is at least 2030 steps ahead of the other,

both can be 52-shifted SW (the one to the SW being shifted first, the second shift being delayed until the second glider is wholly SE of the most southeasterly of the oscillators produced by the first), the result being two 1866-pairs which obey the east-west constraint.[7] These can then be subjected to another 52-shift SW without interference between the pairs, again shifting the more southwesterly pair first and waiting until clear of the oscillators produced before shifting the other. If the initial path-difference and/or forward gap were greater, interference would simply be avoided by a larger margin.

e. A spaced-out glider fleet can now be reconstituted, while adjusting relationships in the SE-NW direction. If the appropriate choices have been made in the preceding stages, this stage can produce any desired spaced-out glider fleet.

The 52-shift is again the key. This time the 1866-pair on the northeastmost path goes through all its collisions first, then that on the next northeastmost does so, and so forth. Each pair goes through an odd number of collisions, all but the last of which are performed in pairs, shifting the 1866-pair 52 SW, then back again NE (the minimum path separation of 82 half-diagonals may be violated between the two collisions, although it will fall no lower than 30). The fact that the previous stage produced a fleet of 1866-pairs makes it possible to choose either a *short-delay 52-shift* or a *long-delay 52-shift* in each collision. The short-delay 52-shift is the one described in connection with stage 1d: the first of the pair takes part in a glider/blinker 6 (or 6b) collision, and the second merely removes a block from the collision remnant left behind. The long-delay 52-shift uses the second of the 1866-pair to create a glider-blinker 6 (or 6b) collision. The first of the pair can be removed using glider-block 5, or can hit a blinker in collision glider-blinker 3b (or 3), which shifts the blinker so that a second glider on the same path and an even number of steps behind will meet it in a glider-blinker 6 (or 6b) collision. The final collision is a glider-block 5, removing the front member of the pair.

The short-delay 52-shift retards a glider pair by 374 steps, and the long-delay 52-shift by 2240 steps. Since the highest common factor of these numbers is 2, a sequence of shifts exists that will produce any particular multiple of 2 steps alteration in the relative forward position of two 1866-pairs (and hence of the back members of these pairs)—as long as the adjustments do not cause interference between the 1866-pairs. By

[7] If the pattern given by the coordinates is run, it will be noted that the NW-headed glider from the second collision hits the first collision remnant and generates two SE-headed gliders, plus one going NE. In the context of a sparse array, the gap between the two collisions would almost always be, in any given era, $\sim N^E$ steps, and the SE-headed gliders thus produced would not interfere with the collision sequences discussed in this subsection.

extension, there exists a sequence of shifts that can produce any set of relative forward position adjustments by multiples of 2 steps for any fleet produced by stage 1d.

Suppose that the fleet contained only two 1866-pairs. A sequence that would delay 1866-pair 1 (the rear, and more northeasterly pair) by $2x$ steps (where x is a positive or negative integer) relative to 1866-pair 2 can be found by finding integers l, s such that $2240l + 374s = 2x$. The desired sequence is then:

- If l is positive, l long-delay shifts to 1866-pair 1.
- If s is positive, s short-delay shifts to 1866-pair 1.
- If l is negative, l long-delay shifts to 1866-pair 2.
- If s is negative, s short-delay shifts to 1866-pair 2.

Once 1866-pair 1 has been shifted back and forth, its front member is removed using a block. Then 1866-pair 2 goes through its back-and-forth shifts and, so long as the rear member of pair 2 does not end up west of the remaining (rear) member of pair 1 (i.e., so long as the constraints on spaced-out glider fleets are respected), removal of its front member will leave the specified two-glider spaced-out fleet.

In the case of a fleet of g 1866-pairs, the set of shifts that will produce a specific g-glider spaced-out fleet can be calculated as follows.

- Start by calculating for 1866-pairs 1 and 2 as above, but assigning the shifts for 1866-pair 2 to all the remaining pairs as well (applying just these shifts would, in effect, adjust the relationship between two subfleets, one containing pair 1, the other pairs 2 through g).
- Then calculate the shifts that would need to occur to produce the specified relationship between pairs 2 and 3, adding the shifts calculated for pair 2 to the running totals for pairs 1 and 2, the shifts calculated for pair 3 to the totals for each pair 3 through g.
- Continue the calculation through the fleet in this fashion, ending with the shifts needed to adjust the relationship between pair g and the rest of the fleet.

The specified spaced-out fleet will then be produced if pair 1 goes through all its shifts, followed by the removal of its front member, then pair 2 is subjected to the same process, and so on through the fleet to pair g—so long as the rear member of each pair, after its final shift, is not to the west of its immediate predecessor. At any stage in this procedure, the single gliders at the rear will form a spaced-out glider fleet. All the lateral gliders sent SW while a given pair is being shifted cannot affect either the spaced-out glider fleet to its NE, nor the remaining spaced-

out 1866-pair fleet to its SW (which will always be to the south of the shifting pair and, hence, of the lateral gliders). Gliders sent NE during its shifts in that direction will, when generated, be located well to the east of the leading member of the growing spaced-out fleet and, hence, of all its members (and, of course, will remain so, since SE-headed and NE-headed gliders have the same eastward component to their velocity). Pattern 7 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩ shows a 1866-pair being shifted SW and then NE, while sandwiched as closely as possible between a single glider to its NE and a 1866-pair to its SW.

2. Any well-spaced glider fleet heading SE can be produced by colliding a spaced-out glider fleet with a sequence of arbitrarily widely spaced blocks and blinkers.

   a. There is a right-angle collision between two gliders (the *kickback* [3, p. 837]), which produces a single glider traveling in the opposite direction to one of the incomers, on a path shifted one half-diagonal in the direction the other incomer was headed. If a well-spaced fleet is headed SE, two gliders headed NW and NE can be so placed that they will meet in a kickback collision, producing a glider forming part of the well-spaced fleet. The glider produced can occupy any position that is no further back (i.e., NW) than any glider in the existing fleet. Hence any well-spaced fleet can be built up from the rear, one glider at a time, using such kickback collisions.

      Consider a well-spaced fleet moving SE, and one NE-headed and one NW-headed glider on course to collide in the kickback reaction in such a way as to add another SE-headed glider to the fleet in a position that will maintain the fleet's well-spaced property. Assume that any other state 1 cells in the array are sufficiently distant to be ignored. Call the state 1 cells contained in the combination of the NW-headed glider and the NE-headed glider, so long as they are separate, and in the product of the collision between the two once they have met, the *kickback cells*. If it can be shown that, during the process of adding a glider in such a position, no state 0 cell neighboring a glider in the original well-spaced fleet can ever come to have three or more state 1 cell neighbors, with at least one being a kickback cell, it will follow that the new glider can indeed always be added without otherwise altering the well-spaced fleet, since no state 1 cells will at any point be added to or subtracted from the sum of those in the existing fleet and the kickback cells, and the latter will in time produce a single SE-glider and nothing else.

      (1) No two gliders in a well-spaced fleet can neighbor the same cell: figure 10 shows the closest cells to a glider in such a fleet that may be occupied by another glider, for two of the glider's phases (the other two

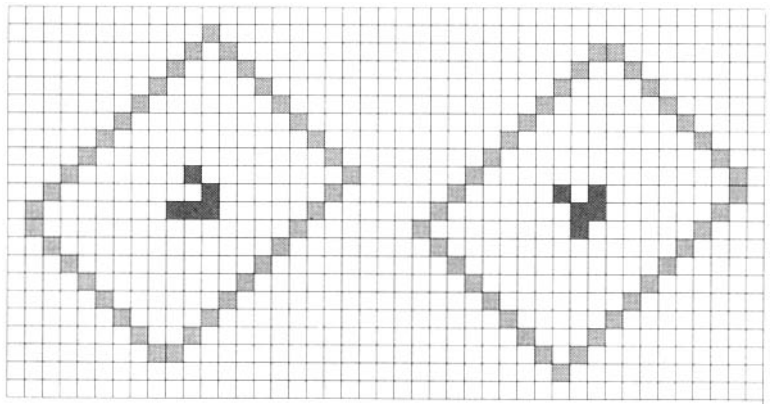

FIGURE 10 Empty space around a glider in a well-spaced fleet.

phases are mirror images of these). Therefore, it need only be shown that no *single* glider in the existing fleet can come to share a state 0 neighbor with the kickback cells in such a way that that glider and the kickback cells together give it three or more state 1 neighbors.

(2) As long as there are three empty half-diagonals between two parts of a cluster, no empty cell can be next to more than one state 1 cell in each part. Suppose the half-diagonals run NW-SE, as in figure 11: the cells in the southwestern half-diagonal cannot have neighbors in the northeastern part of the cluster, those in the northeastern half-diagonal cannot have neighbors in the southwestern part of the cluster, and those in the central half-diagonal can have no more than one neighbor in each part of the cluster, so no cell in this half-diagonal can be switched into state 1. Thus the kickback cells and a glider in the well-spaced fleet cannot interact as long as there are three empty half-diagonals between them.

(3) There are then three cases to consider (the gliders labeled X, Y, and Z in each of the three parts of figure 12 are representative of these three cases); it can be confirmed by inspection that if the claims made apply to these representative gliders, they would apply to any others meeting the given conditions, as none could occupy cells separated from the kickback cells by fewer half-diagonals at any point.

   (a) For any glider, such as X, that is at least 19 steps behind (i.e., NW) of the expected position of the glider to be produced by the kickback cells, there will always be at least three NW-SE half-diagonals between that glider and the kickback cells. As the NE-headed and NW-headed gliders approach each other, the number

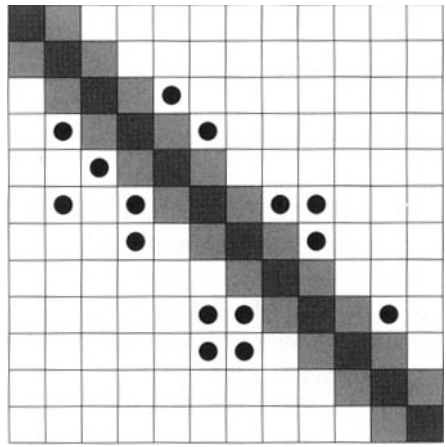

FIGURE 11 Three empty half-diagonals divide a cluster.

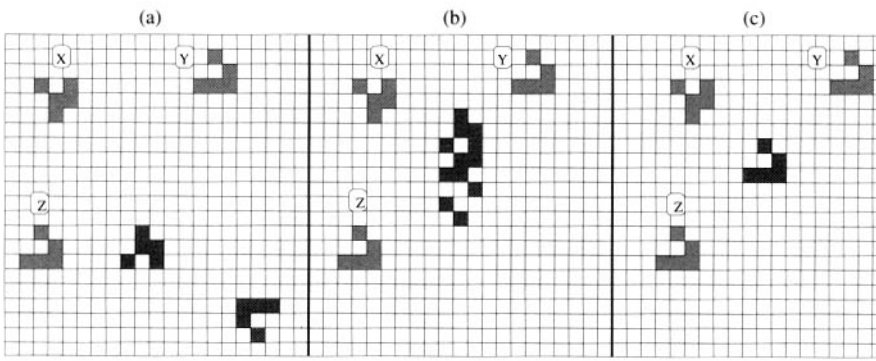

FIGURE 12 Adding a glider to a well-spaced fleet.

of these half-diagonals falls, and reaches a minimum of three for the three steps immediately before the kickback cells are reduced to the new glider and nothing else. At that point, it increases to six, and, henceforth, alternates between five and six.

(b) For any glider on a path at least 12 half-diagonals NE of the expected path of the kickback glider, like glider *Y*, there will always be at least three NW-SE half-diagonals between that glider and the kickback cells. Again, the number of these half-diagonals reaches a minimum of three for the three steps immediately before the kickback cells are reduced to the new glider and nothing else. The number then rises to eight and remains there.

(c) For any glider on a path like that of Z, at least 12 half-diagonals SW of the expected path of the kickback glider and no further forward than the kickback glider's expected position (this condition also applies in case (b) above, but is not needed there), there will always be *either* at least three SW-NE half-diagonals *or* three SW-NE half-diagonals between the glider and the kickback cells. Until 22 steps before the kickback cells are reduced to the new glider and nothing else, there are at least three intervening SW-NE half-diagonals and from that point (part (a) of the figure), there will be at least three intervening NW-SE half-diagonals, as the NE-headed glider moves further NE. Part (b) shows a stage in the actual collision, part (c) the well-spaced fleet with the new glider in place.

b. An *adjustment pair* of NE-headed gliders, so placed that their paths are $2h$ half-diagonals apart (where $h \geq 6$), the two gliders are in the same phase, and the glider on the path further to the SE is 3 cells (12 steps) south of the other, can take part in two successive kickback collisions with a glider headed SE, the first of which results in a glider headed NW, while the second is a collision such as that described in 2a. The net result is that the SE-headed glider's path is shifted two half-diagonals NE, and it is shifted $8h - 20$ steps backward relative to an untouched glider.

c. Such a pair of NE-headed gliders, with $h = 351$, can itself be produced from a fleet of six SE-headed gliders (this will be called an *adjustment fleet*). Such an adjustment fleet can, by colliding with a blinker, produce a pair of NE-headed gliders that will shift the target glider two half-diagonals NE, and 2788 steps backward.

d. The adjustment fleet is not a spaced-out glider fleet, but can be produced, using only 52-shifts, from a six-glider predecessor so placed that it and the target together form a spaced-out fleet. No premature interference with the target will occur in the process. Furthermore, whatever well-spaced fleet is following the target, the adjustment fleet's predecessor can be located so the process will not interfere with that well-spaced fleet.

Pattern 8 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩ consists of 10 gliders and 39 blinkers. The fleet of gliders consists of a close three-glider group at the rear, the target glider whose position relative to this group is to be adjusted, and the 6-glider predecessor of the adjustment fleet. The target plus the predecessor fleet constitute a spaced-out fleet. The target will end up 19 steps directly ahead of one of the three-glider group, and with the other two 12 half-diagonals to either side. The process of producing, moving, and using the adjustment fleet will be described, numbering the gliders from (1)—most SW path—to (6)—most NE path.

All the shifts involved are long delay 52-shifts once the initial single gliders are transformed into pairs. There are four stages: initial rearrangement, SW-move, final rearrangement, and the adjustment itself.

(1) *Initial rearrangement.* Glider (5) is 52-shifted NE once, then each of (4), (3), (2), and (1) is shifted NE three times. Then each of (1), (2), (3), (4), and (5) is shifted SW once. At this point, all the gliders except (6) have been turned into 1866-pairs. (We could now remove the front members of pairs, rather than using them to place a blinker in the right place for the second member of the pair, at the cost of increasing the number of collisions.)

(2) *SW move.* 1866-pairs (1), (2), (3), (4), (5), and (6) are each shifted SW once. This sequence of six collisions could be repeated as many times as desired, without affecting any gliders on paths sufficiently far NE of the predecessor fleet. Hence there may be many positions of the predecessor fleet, differing from each other by multiples of 52 half-diagonals in the NE/SW direction, and of 2240 steps in the direction of movement (SE), which produce an adjustment fleet in the same relation to the target. For any such final position of the adjustment fleet, however, there are only finitely many such initial predecessor fleet positions for which target and predecessor fleet constitute a spaced-out fleet.

(3) *Final rearrangement.* Gliders (2) through (6) are 52-shifted SW once (the adjustment subfleet cannot travel SW in its final form—(1) has to be one shift further SW relative to (2) during the travel process than it will be just before adjustment). Then the front member of each of the 1866-pairs (1) to (5) is removed, (6) is 52-shifted once more (it has to lag behind during the SW-move), and the front member of 1866-pair (6) is removed.

(4) *The adjustment itself.* Glider (1) hits a blinker in a glider/blinker 4 collision, which produces two gliders, headed NE and NW. Other members of the adjustment fleet interact with these to form the adjustment pair. The north-south distance between adjustment fleet and target can be increased (or, up to a point, decreased) by multiples of two cells without affecting the final position of the target. Hence there are *infinitely* many initial positions for the predecessor fleet that make it plus the target glider a spaced-out fleet, and lead to the same adjustment of the target glider. The predecessor fleet can begin arbitrarily far east of the target, and can also begin on paths arbitrarily far SW of the target's path, so long as it produces an adjustment fleet far enough south of the target.

Hence, however far east and/or how far SW the well-spaced fleet to which the target is to be added extends, stage (1) could take place arbitrarily far to *its* east and SW, ensuring that oscillators and additional gliders from the process do not interfere with it: it can be ensured that oscillators and gliders sent NW and SW are well SW of the following fleet, while gliders sent NE in stage (1) are well east of it. Furthermore, a target glider and *multiple* copies of the predecessor fleet could be so placed, in a spaced-out fleet, that each predecessor fleet in turn (beginning with that closest to the target) could be transformed into an adjustment fleet, shifted SW, and used to move the target—all the while remaining on paths as far as desired NE of those being used by the remaining predecessor fleets. It is thus possible to begin with a spaced-out fleet consisting of a target glider, and n predecessor fleets placed on paths arbitrarily far SW of the target, and end with a single glider traveling in the same direction, shifted $2n$ half-diagonals NE and $2788n$ steps backwards relative to the position the target would have occupied if no collisions had occurred (plus debris left behind and gliders traveling in other directions).

(e) Given any desired well-spaced SE-headed fleet, it is, therefore, possible to specify a spaced-out fleet that can produce it through a sequence of collisions with arbitrarily widely seperated blonks.

(1) Number the gliders in the required fleet, beginning at the rear (NW) and if two or more are equally far forward (SE), proceeding from SW to NE along the diagonal (the opposite choice could equally well be made here). Each of the desired gliders will have a precursor in the spaced-out fleet (glider 1 being its own precursor).

(2) Glider 2's precursor will be the second rearmost member of the spaced-out fleet, placed so that it is on a path $2n$ half-diagonals SW of glider 2's desired position relative to glider 1, and $2788n$ steps forward (SE) from that position, for n sufficient to make its relationship to glider 1 compatible with being part of the same spaced-out fleet. How large n will be depends on the required final relationship between the two. It could be zero if the desired relationship between gliders 1 and 2 itself allows them to be part of a spaced-out fleet.

(3) The next members of the spaced-out fleet, moving to more south-westerly paths and at the same time further SE along those paths, will be $6n$ gliders, making up n copies of the predecessor fleet.

(4) Following these will be the precursor of glider 3, placed on a path sufficiently far SW that the adjustment fleets for glider 2 will always be on paths well to the NE of its path. The predecessor fleets necessary to move glider 3's precursor to its final position, once glider 2 has been

placed, will be next. Continuing in this way, a spaced-out predecessor can be specified for any well-spaced SE-headed fleet.

(f) By symmetry, any well-spaced fleet is SCS-constructable.

3. Any well-spaced glider collision can be produced by a collision between some well-spaced glider fleet and a single blinker.

   a. Any well-spaced fleet headed in a specified direction can be the sole product of a collision between some well-spaced fleet headed at right angles to the specified direction, and a single blinker. Assume the original fleet is headed SE. It will be shown that an arbitrary well-spaced fleet headed NE (or, by symmetry, SW) can be created.

      (1) Define a *nicely ordered fleet* as one in which, of any two gliders, one is at least 19 steps ahead of the other. The g gliders of a nicely ordered fleet x can be numbered from back (x_1) to front (x_n). Divide the possible NE-headed nicely ordered fleets into *stretch-resistant fleet classes* as follows: two such fleets x, y are in the same stretch-resistant fleet class if and only if:

         (a) They contain the same number g of gliders.

         (b) For each pair of pairs of correspondingly numbered gliders $(x_i, x_{i+1}), (y_i, y_{i+1})$, the difference in the paths of x_i and x_{i+1} is the same as that between the paths of y_i and y_{i+1}.

         (c) For each pair of pairs of correspondingly numbered gliders $(x_i, x_{i+1}), (y_i, y_{i+1})$ the number of steps separating x_i and x_{i+1} minus the number of steps separating y_i and y_{i+1} gives the same result, which is a multiple of eight.

      Each stretch-resistant fleet class thus has a minimally stretched member, in which the number of steps separating at least one pair of adjacent gliders is between 19 and 26, and the members of the class can be indexed by the minimum number of steps separating any pair of adjacent gliders. It will be shown that all members of each NE-headed stretch-resistant fleet class with a sufficiently large index (how large this is varies from class to class) can be the sole product of a collision between some well-spaced fleet headed SE, and a single blinker.

      (2) There exists a set of related collisions between well-spaced fleets of nine gliders, and a single blinker, which produce a blinker shifted to a new position, and a glider traveling at right angles to the incoming gliders. Assume the fleet to be headed SE.

         (a) The first member of the fleet transforms the blinker into a small still-life, the *ship*.

(b) The second turns the ship into a block, and a glider traveling NW. The number of steps between the first and second can be any number > 18 (in fact, it could be between 14 and 18, but then the fleet would not be well spaced as the second glider's path is just one half-diagonal SW of the first's path).

(c) The third glider collides with the NW glider, creating a pond (fig. 1). The distance between the second and third gliders determines how far NW of the block this pond is. Increasing the distance by eight steps shifts the pond one cell NW (any smaller increase produces the wrong glider collision). The SE-headed glider may be on a path five half-diagonals NE of the NW-headed glider, or five diagonals SW of it. If the possible NE-headed paths are numbered consecutively, one of these alternatives permits the NE-headed glider eventually produced to be on even-numbered paths only, while the other permits only odd-numbered paths.

(d) The fourth glider turns the pond into a *tub*.

(e) The fifth glider turns the tub into yet another small still-life, the *boat*.

(f) The sixth glider collides with the boat, resulting in a new glider going NE (its path determined by how far NW the boat was), and nothing else.

(g) The last three gliders turn the block back into a blinker.

The result is a blinker shifted somewhat (two cells east, 16 cells south) from the original position, and a NE-headed glider.

Two examples are given as Patterns 9 and 10 on ⟨http://psoup.math.wisc.edu/NewConstruction⟩, one using each of the mirror-image possibilities for the collision between NW-headed and SE-headed gliders. The gliders in these examples are not bunched as closely as possible, but the gap after the second glider could be increased by any multiple of eight steps to shift the NE-headed glider two half-diagonals NW (and four steps backwards), while the gap after the sixth could be increased by any number of steps to delay recreating the blinker by the same amount. Other gaps could also be increased, but it is most convenient here to hold them fixed at the values shown in the examples. The two examples, along with the possibility of increasing the gap after the second glider, permit the outgoing glider to be on any SW/NE half-diagonal sufficiently far NW of the blinker's central cell.

(3) Now consider how this approach could be used to create a two-glider fleet going NE. This involves 16 SE-headed gliders. The NE glider created by the first six SE-headed gliders can travel arbitrarily far before any further gliders arrive if the gap after the sixth glider is wide enough, allowing any possible interference to be avoided. The gap after the ninth glider could also be varied, but will be fixed arbitrarily at 100 steps to aid exposition. Specifically, we can ensure that the new glider's rearmost cell is at least four half-diagonals NE of the NE edge of the track of any of the remaining gliders, before the leader among them advances to within less than four half-diagonals of the new glider's track (thus ensuring at least three empty half-diagonals between them at all times). Even if the second NE-headed glider is to follow a track to the SE of the first, this will ensure noninterference. Since both the first and second NE gliders can, independently, be produced arbitrarily far NW of the restored blinker, and since the two ways in which the third glider can meet the NW glider allow either an odd or an even path-difference between them, all possible path-differences can result. The ability to increase the gap between the sixth and seventh gliders by an arbitrary amount allows the first to lead the second by any amount, above a minimum which depends on the relationship between their paths, and therefore by any number of steps modulo 8.

This ensures that, for any two-glider stretch-resistant fleet class, an index exists such that any member with an index at least that large can be produced as the sole result of a collision between a blinker and a sixteen-glider fleet: one nine-glider fleet, and a second modified to contain a single glider to remove the block (and itself) using glider/block 5, rather than three gliders to restore the blinker. The gap before the final glider can be extended to ensure non-interference with the NE-headed fleet just like that between the first six gliders and the rest.

(4) Furthermore, the result generalizes to fleets of any number of gliders. For suppose that for any g-glider stretch-resistant fleet class, there is an index such that any member with an index at least that large can be produced as the sole result of a collision between a blinker and a $(9g-2)$-glider fleet. Given a $(g+1)$-glider stretch-resistant fleet class, consider the fleets produced by removing the leading glider. These will constitute a (not necessarily proper) subset of a g-glider stretch-resistant fleet class. Beginning with the member of that subset with the minimum index allowing it to be constructed using the method described, consider whether a member of either of the sets of two nine-glider fleets of which examples were given above, can be added to the front of the fleet used to construct it (with a 100-step gap between

the existing fleet and the rearmost of the added gliders), to produce a fleet that can construct the corresponding member of the $(g+1)$-glider class (this will have the same or a lower index—the latter if, for members of that class, the gap between the leading and second gliders is the smallest). Which of the two sets should be considered depends on the value modulo two of the path-difference required between gliders $g+1$ and g of the new fleet. The gap required between the second and third of the nine added SE-headed gliders will be determined by the exact value of this path-difference. The value of the forward gap desired between the first and second gliders of the new fleet will determine the number of steps needed between the sixth and seventh of the added gliders. If this is not sufficient to ensure that the first glider travels NE beyond the path of any of the remaining SE-headed gliders, as described in the two-glider case above, the index can be increased, together with *all* gaps between sixth and seventh gliders in the nine-glider construction subfleets, until this criterion is met.

(5) Any well-spaced NE-headed fleet containing g gliders can be created by colliding any member of a NE-headed stretch-resistant fleet class with a sufficiently large index (how large this is depends on the well-spaced fleet to be created), and a well-spaced SE-headed fleet containing $2(g-1)$ gliders.

Adjustment pairs of SE-headed gliders, similar to the NE-headed pairs described in stage 2, can be used to change the relationships between the gliders of the NE-headed fleet. As shown in stage 2, such a pair can add a glider to the front of a well-spaced fleet in any position that leaves it still well spaced. Starting with the second glider from the back and moving forward through the fleet, each can be shifted two half-diagonals to the SE, and $8n-20$ steps backward, where $n \geq 6$. For any specified well-spaced NE-headed fleet, it is possible to identify a stretch-resistant fleet class such that any member of that class with a sufficiently large index can be used to produce it through collision with $g-1$ adjustment pairs.

(a) Number the gliders in the required fleet 1 to g, beginning at the rear (SW) and if two or more are equally far forward (NE), proceeding from NW to SE along the diagonal. Members of the stretch-resistant class of nicely ordered NE-headed fleets to be identified will also have g gliders, and the relationships between the paths they follow will be the same as for the specified well-spaced fleet, except that glider 1 will be on a path two half-diagonals further SE relative to the other gliders than in the well-spaced fleet. The number of steps between any two gliders in the SW-NE direc-

tion will be the same modulo eight as in the specified well-spaced fleet, except for gliders 1 and 2, where it should be four different modulo 8 (see (b) below).

(b) Take the specified well-spaced fleet and move glider 2 forward (NE), until it is 28 steps further forward, relative to glider 1, than it needs to be in the specified well-spaced fleet.

(c) Then, if $g > 2$, work through gliders 3 through g, placing glider i so it is at least 36 steps east of glider $i-1$ (36 is the minimum necessary to be certain of avoiding interference with the collision between glider $i-1$ and the first member of its adjustment pair: that (SE-headed) glider will be 17 steps east of glider $i-1$, and cannot collide with any glider going NE and at least 19 steps to *its* east). Each of gliders 3 through g should also be no nearer its predecessor in the SW-NE direction than the required distance between the corresponding gliders in the specified well-spaced fleet, and this number of steps must be correct modulo 8. Subject to these three conditions, place each glider as close as possible to its predecessor.

The result is the member of a suitable stretch-resistant class (not the *only* such class), with the lowest index possible. Each glider from 2 through g can in turn collide successively with the two members of a SE-heading adjustment pair, the first turning it to the SW, and the second back to the NE. Twenty-eight steps backward is the least adjustment that can be made: the adjustment pair are then on SE-headed paths differing by 12 half-diagonals. Each successive adjustment will be at least as large as its predecessor. The successive adjustment pairs can be arbitrarily far apart, so a well-spaced SE-headed fleet of $2(g-1)$ gliders can be used to turn this fleet into the specified well-spaced NE-headed fleet. Any member of the stretch-resistant class with a larger index could also be used, by increasing the path-differences within each adjustment pair: if the index increases by eight (the smallest possible jump), the paths of the two gliders in the ith adjustment pair must be set $2i$ half-diagonals further apart, so that glider $i+1$ is shifted backwards by $8i$ steps more.

(6) Since 3a(4) shows that any member of any stretch-resistant class of nicely ordered NE-headed fleets with a sufficiently large index can be the sole product of colliding a well-spaced SE-headed fleet with a single blinker, 3a(5) establishes that any well-spaced NE-headed fleet can be the sole product of colliding some well-spaced SE-headed fleet with a single blinker, because the adjustment pairs can form part of the same well-spaced fleet. By symmetry, any well-spaced fleet can be the

sole product of a collision between a well-spaced fleet headed at right angles to the fleet produced, and a single blinker.

It is worth noting that any SCS-constructable fleet can therefore be produced in such a way that no other products of the SCS are located directly behind it (see section 5.2 and footnote 7).

b. Part of a well-spaced SE-headed fleet can produce a well-spaced NE-headed fleet by colliding with a blinker, and subsequently (and after an arbitrarily long time) the fleet can collide with a second well-spaced SE-headed fleet. Since the two SE-headed fleets can be arbitrarily distant, they can form part of a single well-spaced fleet. Therefore, any well-spaced glider collision involving SE-headed and NE-headed fleets, and by symmetry any collision between two fleets traveling at right angles can be created from a collision between some well-spaced glider fleet and a single blinker.

c. As the first acts in a well-spaced collision between two fleets traveling SE and NE, two pairs of gliders can create two blinkers [3, p. 831]. The northeastern part of the SE-headed fleet can then collide with one of these two blinkers to create a well-spaced SW-traveling fleet, the southeastern part of the NE-traveling fleet with the other blinker to create a well-spaced NW-traveling fleet. One of the blinker-creating collisions can occur an arbitrarily long time before the other, and the second an arbitrarily long time before the collisions creating the SW- and NW-headed fleets begin. The four fleets can be arbitrarily far apart when their construction is completed, avoiding any possible interference.

Therefore, from stages 1, 2, and 3, any well-spaced glider collision can be created from a well-spaced glider collision involving two fleets at right angles.

4. Therefore, any well-spaced glider collision is SCS-constructable.

6.3 SCARCITY-LEVEL-FIVE AND SCARCITY-LEVEL-SIX COLLISION SEQUENCES

An alternative name for SCSs, suggested by figure 8, would be sequences of scarcity level five: the cumulative occurrence density of different orders of SCSs cannot exceed $\sim p^5$. Collision sequences which begin with a six-cell cluster and otherwise involve only original three-cell clusters, and those that begin with a five-cell cluster but involve one original four-cell cluster in addition to zero or more original three-cell clusters, are, analogously, sequences of scarcity level six: their cumulative occurrence density cannot exceed $\sim p^6$. Similarly, we can define sequences of scarcity levels seven, eight, and so forth.

As figure 8 indicates, sequences of any order belonging to scarcity class n will become more common than those of any order of scarcity class $n + 1$ at some point before era 3, if collision processes such as those described continue effectively unchecked to that point. However, there might be clusters that can be produced *only* by sequences of scarcity level greater than five. In general this possibility remains open, but it can be shown that anything constructable by sequences of scarcity level six is also constructable by sequences of scarcity level five—i.e., is SCS-constructable. The most succinct way of showing this also involves showing that anything SCS-constructable is constructable in a well-spaced glider-collision—showing that SCS-constructability and well-spaced glider-collision constructability are equivalent, except that the latter, has been taken to imply that the object of construction is a *pattern*—i.e., it is the sole product of the glider collision—whereas SCS-construction leaves a litter of debris in its wake.

1. For those sequences of scarcity level six that begin with a glider or r-pentomino, it needs to be shown that any collision between an SCS-constructable fleet and a cluster derivable from a four-cell original is itself SCS-constructable.

    a. Any SCS-constructable fleet is constructable from a well-spaced glider collision.

    Any SCS-constructable fleet is derived from a finite sequence of collisions beginning with a glider or r-pentomino, and involving a single block or blinker at a time. While there are constraints on the positions of these blonks relative to each other, the minimum distance between any two can be made arbitrarily large in any given SCS.

    Given an arrangement of oscillators with minimum distance d_1 between any pair of oscillators and a minimum "safe distance" d_2, it will always be possible, if d_1 is sufficiently greater than d_2, to find a path consisting of half-diagonal stretches, from outside the arrangement to any point within it at least d_2 cells clear of any oscillator, while avoiding all the oscillators by that amount. From a point directly NW of the desired point, one such path goes SE until the minimum safe distance is about to be breached. It then goes SW until it can go SE again, goes SE until clear of the oscillator in its way, then NE as soon as possible to get back to the original track, turning SE once more when directly NW of the goal. If the difference between d_1 and d_2 is sufficiently great, it will always be possible to go around each obstacle in this way and return to the original NW-SE track. For a path wider than a single half-diagonal, it is only necessary to increase d_1 correspondingly.

A block, a blinker, and a beehive can each be the sole product of a right-angled collision between a pair of gliders [18]. An r-pentomino can be created as the sole product of a collision between a beehive and a glider. A boat can be constructed by a collision between a fleet of four arbitrarily widely spaced gliders and a blinker (glider/blinker 1 creates a pond, and three more gliders turn this into a boat—see section 6.2). Both beehive and boat can take any of their possible orientations (the beehive has two, the boat four), whichever two perpendicular directions the gliders producing it come from.

A boat can turn a glider through a right angle (to left or right, depending on the boat's orientation), vanishing as it does so. Since all the collisions referred to in the previous paragraph affect an area of finite diameter, any arrangement of blonks, beehives, and boats with a sufficient minimum distance between any two can be built up one oscillator at a time in a well-spaced collision between glider fleets headed SE and NW: the oscillators are constructed, working from south to north and, where two are equally far north, from west to east. The boats (if any are needed) are placed so that they will guide a final glider to the required location within the arrangement, along a path like that described above. Provided the minimum spacing between any two oscillators is sufficient, this will always be possible without interference between glider/boat collisions and other oscillators. Once guided, the final glider will collide with a beehive, producing an r-pentomino, or with the first blonk in the desired collision sequence. (The symmetries of GoL make it sufficient to be able to produce a glider going in one direction (SE) at the desired location, since the oscillator arrangement can be built in any orientation.)

b. The traffic lights, pond, and tub (the only non-null oscillators derivable from four cells other than the block and blinker—see figure 1) can all be created from blinkers by collisions with either a single glider, or two gliders with an arbitrarily long gap between their arrival. Hence, any collision sequence beginning with a five-cell cluster and involving one four-cell cluster plus any number of three-cell clusters is SCS-constructable using a simple modification of the above technique in which a set of traffic lights, a pond, or a tub is substituted for one of the blonks.

2. The other kind of scarcity-level-6 collision sequence begins with a six-cell cluster and, thereafter, involves only three-cell clusters. It has been shown by exhaustive survey that there are only five development classes of initial six-cell patterns that produce gliders, and that all of these are constructable from a well-spaced glider collision (details available from the author). Hence, anything constructable by a collision sequence of scarcity level six is also SCS-constructable.

7 SCS-CONSTRUCTIONS AND GLOBAL MEDIUM-TERM EVENTS

The relationship between SCS-constructability and well-spaced glider fleet constructability link this work with work described elsewhere in this volume [19], on constructing GoL patterns by colliding glider fleets. The fleets used in such constructions are not necessarily well-spaced, but if not can generally be modified to be so fairly easily. It can be shown that, among others, and in addition to the lightweight spaceship and block-laying switch engine for which constructions have been described, the following clusters are SCS-constructable:

- The two next smallest orthogonal spaceships (middleweight and heavyweight).
- The glider-stream switch engine.
- At least one of the minimal known patterns with quadratic growth of its cumulative image, mentioned in subsection 3.2.
- The minimal known pattern with quadratic growth in its current cell count, mentioned in subsection 3.2 (more strictly, patterns in the same development class).
- Bill Gosper's *space rake* [27, p. 7], the first to be discovered of a class of IGCs with a moving head that emits spaceships (in this case, gliders) traveling at an angle to its own path, without leaving a permanent trail of oscillators.
- The *glider gun*, an IGC consisting of a stationary part that emits a stream of gliders (one every 30 steps), plus the growing stream of gliders.
- The *eater*, a small pattern that can be created in a two-glider collision, and can absorb the glider gun's stream of gliders.

The last three items can be used to construct patterns including infinite families of more complex guns that fire either gliders, orthogonal spaceships, or structures such as the space rake itself (giving the pattern quadratic growth in cell count); and of *puffer trains*, with moving heads and various products including oscillators, spaceships, guns, and other puffer trains. Finally, it is likely that self-reproducing universal computers of the kinds reportedly shown to exist in early and unpublished work on GoL [6] would be SCS-constructable, as their reproductive capability apparently depended on the construction of blocks, eaters, glider guns and other components in collisions between glider fleets with long delays between gliders.

Anything which can be constructed by an SCS will be in an infinite random GoL array of any density. In a sparse random array, any specific structure will be constructed in this way at a rate determined by the shortest SCS that produces it (as in the example of the lightweight spaceship given above). The process outlined in section 6.2 would probably never be anything like the shortest sequence.

As has been argued above, the final collisions in SCSs of order b will attain cumulative occurrence densities of $\sim p^{5+b(3-E)}$ until the era in which typical original gliders collide with something in their way. Equivalently, the rate per

step at which bth collisions in SCSs will occur will be $\sim p^{8+(b-1)(3-E)}$ until this era (fleets of gliders—or other spaceships—resulting from order $b-1$ SCSs will have a density of $\sim p^{5+(b-1)(3-E)}$, and there is a $\sim p^3$ probability per step of such a fleet encountering an original blonk in the bth collision of the sequence). The critical era will not be later than era 3—when, if nothing else interferes, effectively all original gliders will hit original blonks—nor earlier than era 14/5, as argued in section 5.1. If the shortest SCSs leading to construction of IGCs are of order b (and assuming, as seems likely, that the IGCs concerned show linear growth in diameter and current cell count, like switch engines, glider guns, and all others known with fewer than 52 cells), their density contribution would reach $\sim p^3$ in era $(3b+2)/(b+1)$, and they would reach a diameter at which effectively all of them would collide with each other (if they were block-laying switch engines, by running into each others' trails) in era $(3b+5)/(b+2)$.

However, this overlooks the fact, explained earlier, that IGCs such as switch engines eventually reach a size ($\sim N^{5/2}$ cells, $\sim N^{5/2}$ steps after their construction) which makes them liable to bombardment by original gliders and the fleets from original r-pentominos. In the case of QGCs, after $\sim N^{3/2}$ steps, at cell count $\sim N^3$ but diameter $\sim N^{3/2}$, reactions with original blonks will become significant. Interactions of these kinds have so far proved analytically intractable—chiefly because of the unlimited number of (apparently) fundamentally different cases that need to be considered, and they limit what can be said about the medium-term history of sparse GoL fields. For example, it has not yet been possible to determine whether the global density of state 1 cells ever (after $t=0$) $\gg p^3$. The most that can be said is that, if this *is* the case sometime before era 3, IGCs produced by collision sequences will have been causally concerned in producing almost all state 1 cells in the array and, if it is not, SCSs will have produced, at some time before era 3, the nearest example to almost every cell of every finite-difference class of IGC which is SCS-constructable (see Gotts [13] for further description of the current state of knowledge in this regard, and supporting arguments).

Discovery of an order 49 SCS producing a block-laying switch engine, ensures that in any era later than era 281/96 but sufficiently close to that era, at least one of the following must be true:

1. "Live" (still growing) IGCs produced by collision sequences outnumber original IGCs.
2. State 1 cells belonging to IGCs produced by collision sequences form the vast majority of those in the array.

The rate of production of block-laying switch engines by the order 49 SCS will be $\sim p^8 \times p^{48(3-E)}$, unless the rate of SCS completion slows down significantly (for which it is a precondition that the global density of state 1 cells becomes $\gg p^3$). Again assuming that the global density of state 1 cells does not become $\gg p^3$, each will exist (and grow) for at least $\sim N^{3/2}$ steps, so the number in existence at any era will be at least $\sim p^{13/2} \times p^{48(3-E)}$. This will reach $\sim p^{10}$,

the density of original indefinite growth clusters, in era 281/96, and exceed it in subsequent eras (although the number of IGCs may subsequently fall as they collide with each other).

8 OPEN QUESTIONS AND GENERALIZATIONS

The open questions concerning GoL on infinite arrays raised by this chapter fall into two main groups, concerning the relations between various types of constructability, and the medium-term history of sparse random GoL arrays.

It has been shown here that SCS-constructability coincides, in a sense, with well-spaced glider-fleet constructability. Obvious targets of future work are to show that well-spaced glider-fleet constructability is (or is not) equivalent to glider-constructability (constructability from any set of converging glider fleets, well spaced or not), that the latter is (or is not) equivalent to ω-constructability, and that this is (or is not) equivalent to n-constructability for some finite n; and to discover the relations between n-constructability and $(n-1)$-constructability for $n \geq 1$. Another obvious aim, more closely related to the work reported here and linking to questions about the medium-term history of sparse random GoL arrays, is to show whether SCS-constructability encompasses constructability by collision sequences of scarcity levels greater than six.

The long-term fate of sparse random GoL arrays remains undetermined, and the same is true of their medium-term history after era 14/5. This limit could be pushed back by extending the exhaustive survey of SCSs: showing there are no order 3 SCSs that generate IGCs, for example, would push the limit of certain knowledge about the rate at which SCSs of different orders are completed back to era 17/6, as it would be known that most original gliders, and most gliders from original r-pentominos, would meet no obstruction before then. The minimum value of x for which it is not known that the density of the array after N^x steps $\rightarrow 0$ as $N \rightarrow \infty$ (i.e., as $p \rightarrow 0$) would also become 17/6: for any lower value of x, the density would be known not to exceed p^y for some positive y.

Depending on the rate at which the number of distinct SCSs grows, the lowest order of SCS which produces an IGC, and the discovery of efficient ways of searching the space of SCSs, it may be possible to determine what that lowest order is, and what class of IGC is produced. Similarly, it is feasible to discover whether the two finite-difference classes of switch engine are the only such classes of IGC with 10-cell precursors, and so determine tighter limits on the medium-term influence of original IGCs. Finally and more speculatively, it may be possible to prove that there are at least some linear IGCs that will go on growing past $\sim N^{5/2}$ cells despite glider bombardment, or some QGCs that go on growing past $\sim N^3$ cells despite interaction with original blonks.

Looking beyond GoL on infinite arrays, attempts can be made to apply the concepts and methods described here to other CA. The most obvious targets are finite (toroidal) arrays of GoL itself. In toroidal sparse random GoL arrays which are very large in both "north-south" and "east-west" directions we can select,

for any sufficiently low value of p, dimensions for the toroidal array which will, for example, make it likely that the original configuration contains many gliders and r-pentominos, but has no sets of six or more state 1 cells close together. The probability that this will be so can be pushed arbitrarily close to certainty by decreasing p and increasing the size of the torus at the same time. IGCs may then be produced by SCSs in arrays that originally had no IGCs at all, and will dominate the medium-term dynamics of the array. In the long term, any finite array will become periodic, but there may be interesting questions about the kinds of periodicity that result and how long they take to emerge: for example, what happens to the mean or median period of the array's final state as p and the dimensions of the array are varied?

Looking beyond GoL, some exploratory attempts have been made to apply aspects of the approach developed here to the *elementary* one-dimensional CA studied by, for example, Wolfram [29], Braga et al. [4], and Dhar et al. [8]—and to the CA *HighLife*, which is closely related to GoL [2]. HighLife's transition rule is identical to that of GoL, except that *six* state 1 neighbors, as well as three, turn a state 0 cell into a state 1. The glider (since it always has five state 1 cells) operates in HighLife as in GoL, but none of the glider/blonk collisions generates further gliders. There is a six-cell IGC in HighLife, of a kind quite different to the smallest IGCs in GoL: its current cell count, although unbounded, also returns an unbounded number of times to a minimum of 22. The initial six-cell pattern, in fact, becomes an 11-cell self-replicating pattern at $t = 2$, and the subsequent 22-cell minima consist of two copies of this, further apart each time. Between these minima, there are steps at which ever larger numbers of copies of this subpattern appear in a diagonal row (the pattern does not increase in width). The mean number of cells increases at a rate $\propto t^{\log_3 / \log_2 - 1}$. This is significantly slower than a linear increase, and most examples of this IGC in a sparse random field will run into an original blonk at one end before a glider runs into them—which would not be the case for an initial six-cell cluster with a linearly increasing current cell count. Collision with a blonk causes the row of replicators to be eaten away.

More widely, the concepts developed in order to study sparse random GoL fields have promise as tools for CA classification. In studying GoL, it has been important to determine, so far as possible, what are the minimum size patterns, or *interruptions* in a uniform state 0 configuration, which have dynamic properties such as persistence, affecting an unbounded number of cells over time, unbounded increase in the current cell count, and so forth. Such a *minimal size cluster profile* might be used in classifying CA; it seems reasonable to conjecture that similarities in profile indicate wider similarities in the dynamics of different CA. (Classifications in terms of whether there are *any* finite patterns with particular dynamic properties are suggested by, for example, Culik et al. [7] and Braga et al. [4].) The approach is not limited to considering interruptions in completely uniform configurations. Many CA, such as Wolfram's rule 110 [29, pp. 547–549] and rule 54 of Hanson and Crutchfield [15], have a strong tendency to produce

configurations consisting of moving and interacting interruptions or *particles* on a nonuniform but strongly patterned background, each stretch between two interruptions having the same kind of spatial and temporal periodicity. It makes sense to ask of such a background whether, for example, it is sufficiently stable that any finite interruption in it will remain of bounded size, and if not, what is the smallest interruption (in terms of cell size or diameter) that does not do so. The same idea can be applied to classes of configuration which are not periodic but forbid the appearance of certain sequences (in the one-dimensional case) or chunks (in the two-dimensional).

A criterion for the boundedness of an interruption that will apply in both one-dimensional and two-dimensional cases is not obvious. Hamming distance, for example, would not work well in the rule 110 case mentioned above, as a small interruption can give rise to two interruptions of bounded size that move away from each other, with an ever-growing out-of-phase patch of background between them. This is intuitively an interruption that remains finite, but the Hamming distance between this configuration and an uninterrupted periodic sequence grows without bound. The following is one possibility. Define a *window* as any finite configuration of cells (e.g., in the one-dimensional case it could be a single cell, n adjacent cells for any n, six cells in one row of four and one row of two with m empty cells between them for any m, etc.). Such a window can clearly be placed on an infinite array of cells in an infinite number of ways. What can be seen through the window in a given placement may or may not be one of the arrangements of cell-states that could appear if the window were placed over the uninterrupted background. If, for every window, there is a number w such that the number of placements showing a pattern that could not belong to the background never exceeds w, the interruption is regarded as bounded.

The work reported casts light on the emergence of complex structures and behaviors from simple local interactions, and the relationships between events on different spatial and temporal scales, in a particular CA. However, these are phenomena with much wider relevance within CA, as the question at the end of the introduction suggests, and also in the study of real-world examples of complex, spatially extended systems with discrete aspects (physical, biological, and social), which CA are well suited to model [9, 12].

ACKNOWLEDGMENTS

I wish to thank the Scottish Executive Rural Affairs Department for funding support for the time required to attend the CA98 workshop and write this chapter, the Santa Fe Institute and National Science Foundation for funding my travel and accommodation costs, and the workshop organizers, editors of this volume, and SFI staff for their assistance.

The contributions of Paul B. Callahan and the late Robert Norman are gratefully acknowledged. Many people contributed to discussions of sparse arrays, to

software I have used, or both. These individuals include David Bell, Jon Bennett, Johan Bontes, Dave Buckingham, Tim Coe, John Conway, Noam Elkies, Achim Flammenkamp, Bill Gosper, Janko Gravner, Alan Hensel, Dean Hickerson, Dan Hoey, Heinrich Koenig, the late Dietrich Leithner, Mark Niemiec, Gabriel Nivasch, Tomas Rokicki, Rich Schroeppel, Stephen Silver, Andrew Trevorrow, Bob Wainwright, and Allan Wechsler.

REFERENCES

[1] Bagnoli, F., R. Rechtman, and S. Ruffo. "Some Facts of Life." *Physica A* **171(2)** (1991): 249–264.

[2] Bell, D. I. "Highlife: An Interesting Variant of Life." Available from David Bell's web page: `http://www.tip.net.au/~dbell/`, May 1994.

[3] Berlekamp, E., J. H. Conway, and R. Guy. *Winning Ways*, vol. 2. New York: Academic Press, 1982.

[4] Braga, G., G. Cattaneo, P. Flocchini, and C. Quaranta Vogliotti. "Pattern Growth in Elementary Cellular Automata." *Theor. Comp. Sci.* **145** (1995): 1–26.

[5] Chou, H.-H., and J. A. Reggia. "Emergence of Self-Replicating Structures in a Cellular Automata Space." *Physica D* **110(3-4)** (1997): 252–276.

[6] Conway, J. H. Personal communication, 1999.

[7] Culik, K. L., P. Hurd, and S. Yu. "Computation Theoretic Aspects of Cellular Automata." *Physica D* **45** (1990): 357–378.

[8] Dhar, A., P. Lakdawala, and G. Mandal. "Role of Initial Conditions in the Classification of the Rule-Space of Cellular-Automata Dynamics." *Phys. Rev. E* **51(4A)** (1995): 3032–3037.

[9] Epstein, J. M. *Growing Artificial Societies: Social Science from the Bottom Up.* Washington, DC: Brookings Institution Press, 1996.

[10] Garcia, J. B. C., M. A. F. Gomes, T. I. Jyh, T. I. Ren, and T. R. M. Sales. "Nonlinear Dynamics of the Cellular-Automaton 'Game of Life.'" *Phys. Rev. E* **48(5)** (1993): 3345–3351.

[11] Gibbs, P., and D. Stauffer. "Search for Asymptotic Death in Game of Life." *Intl. J. Mod. Phys. C* **8(3)** (1997): 601–604.

[12] Gilbert, N., and K. G. Troitzsch. *Simulation for the Social Scientist.* United Kingdom: Open University Press, 1999.

[13] Gotts, N. M. "Emergent Phenomena in Large Sparse Random Arrays of Conway's 'Game of Life.'" *Intl. J. Sys. Sci.* **31(7)** (2000): 873–894.

[14] Gotts, N. M., and P. B. Callahan. "Emergent Structures in Sparse Fields of Conway's 'Game of Life.'" In *Artificial Life VI: Proceedings of the Sixth International Conference on Artificial Life*, edited by C. Adami, R. K. Belew, H. Kitano, and C. Taylor. Cambridge, MA: MIT Press, 1998.

[15] Hanson, J. E., and J. P. Crutchfield. "Computational Mechanics of Cellular Automata: An Example." *Physica D* **103(1-4)** (1997): 169–189.

[16] Langton, C. G. "Self-Reproduction in Cellular Automata." *Physica D* **10** (1984): 134–144.

[17] Malarz, K., K. Kulakowski, M. Antoniuk, M. Grodecki, and D. Stauffer. "Some New Facts of Life." *Intl. J. Mod. Phys. C* **9(3)** (1998): 449–458.

[18] Niemiec, M. D. "Mark D. Niemiec's Life Page." ⟨`http://home.interserv.com/~mniemiec/lifepage.htm`⟩.

[19] Niemiec, M. D. "Synthesis of Complex Life Objects from Gliders." This volume.

[20] Poundstone, W. *The Recursive Universe.* New York: William Morrow, 1985.

[21] Reggia, J. A., J. D. Lohn, and H.-H. Chou. "Self-Replicating Structures: Evolution, Emergence, and Computation." *Artificial Life* **4(3)** (1998): 283–302.

[22] Rokicki, T. "An Exploration of Metacatacryst." ⟨`http://tomas.rokicki.com/mcc/`⟩.

[23] Sales, T. R. M., J. B. C. Garcia, T. I. Jyh, T. I. Ren, and M. A. F. Gomes. "On the Game of Life: Population and Its Diversity." *Physica A* **197** (1993): 604–612.

[24] Shante, V. K. S., and S. Kirkpatrick. "An Introduction to Percolation Theory." *Adv. Phys.* **20** (1971): 325–357.

[25] Silver, S. "Life Lexicon," release 18, March 26, 2001. Available from ⟨`http://www.argentum.freeserve.co.uk/lex_home.htm`⟩.

[26] Silver, S. Personal communication, 1998.

[27] Wainwright, R. T., ed. *Lifeline: A Quarterly Newsletter for Enthusiasts of John Conway's Game of Life.* **3** (September 1971).

[28] Wainwright, R. T., ed. *Lifeline: A Quarterly Newsletter for Enthusiasts of John Conway's Game of Life.* **4** (December 1971).

[29] Wolfram, S. *Theory and Applications of Cellular Automata.* Singapore: World Scientific, 1986.

This page intentionally left blank

Synthesis of Complex Life Objects from Gliders

Mark D. Niemiec

1 INTRODUCTION

Life, like many other cellular automata, contains many interesting objects, such as still lifes, oscillators, spaceships, spaceship guns, puffer trains, breeders, and the like. While many of these, like blocks, blinkers, and gliders, occur naturally with great frequency, there are many others that occur infrequently, and countless others that have never yet been observed in any natural context.

This chapter deals with methods for synthesizing such complex objects from simple building blocks, such as gliders or other easy-to-synthesize objects. Once an object can be shown to be built in this manner, the object may be used as a building block in larger relocatable structures, such as Turing machines or universal constructors. In addition, the existence of a natural synthesis of an object from a bounded number of gliders implies that the object will form naturally in a sufficiently large, sufficiently sparse field [2].

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

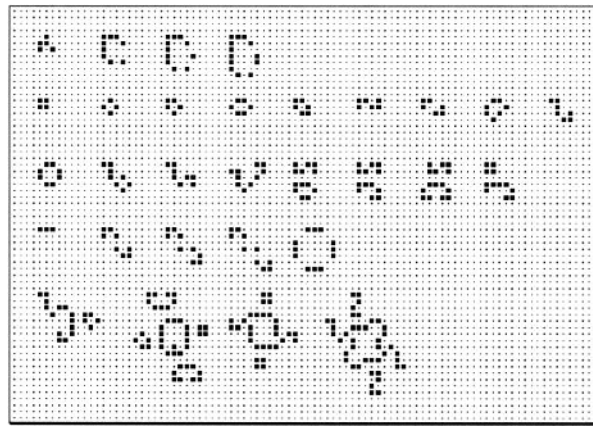

FIGURE 1 Some simple objects mentioned in this chapter. Row 1: Spaceships: glider, lightweight spaceship (LWSS), middleweight spaceship (MWSS), heavyweight spaceship (HWSS). Row 2: Still lifes: block, tub, boat, beehive, ship, snake, carrier, loaf, eater. Row 3: Still lifes: pond, tub with tail, hook with tail, 10.16, snakes on bun, bookend, and house; block on long bookend. Row 4: Flip-flops: blinker, bipole, tripole, quadpole, traffic-lights. Row 5: Oscillators: cuphook, Hertz oscillator, Hustler, Hustler II.

Inasmuch as this chapter deals mainly with practical aspects of object synthesis, rather than theoretical ones, it may resemble a talk on chemical engineering, rather than abstract mathematics.

1.1 FIGURES

All figures shown here, unless otherwise specified, show "before" and "after" images of collision sequences; the "before" sequences are shown on the left with dark cells, and the "after" sequences to the right of them in lighter cells. In some cases, unwanted debris is also generated and must be removed later; this debris is shown in the lightest color.

2 BASIC SYNTHESIS METHODOLOGIES

There are several basic ways in which objects can be synthesized.

2.1 NATURAL OBJECTS

The most common objects occur in great abundance in nature, so there are many random collisions of a small number of gliders that will produce them.

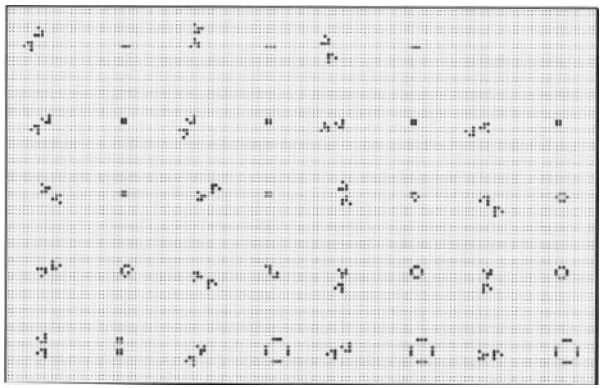

FIGURE 2 Some objects that can be synthesized from two gliders. Row 1: Blinkers. Row 2: Blocks. Row 3: Blocks, boat, beehive. Row 4: Loaf, eater, ponds. Row 5: Block-on-block, traffic-lights.

There have been many random broth experiments conducted in Life, in which fields initialized to random initial configurations have been run until they became periodic, and then the resulting ash analyzed. The results of two such series of experiments, performed by Achim Flammenkamp [1] and Heinrich Koenig [3], are available on the Web.

If the objects are sorted in order of decreasing frequency of natural occurrence, the list is also in order of increasing synthesis cost in gliders (with a few rare objects out of place). Of course, these observations are only approximate, since they rely on randomly generated fields, and the cost in gliders depends on known technology that sometimes improves with time. Nevertheless, these experiments confirm the intuitive notion that rarer objects need more gliders to synthesize.

Some examples of naturally occurring objects appear in figures 2–4.

2.2 SYMMETRICAL OBJECTS

There are many objects that are normally extremely rare, but that occur with much greater frequency in highly symmetrical fields, due to the unusual distribution of neighborhoods along lines and points of symmetry.

In figures 5–7 are some more colorful examples, most found by David Buckingham, Dean Hickerson, and the author.

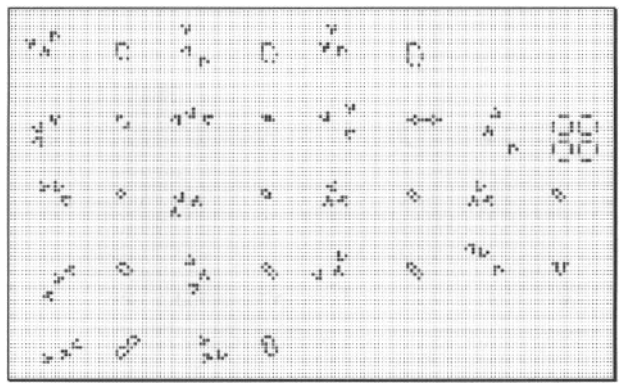

FIGURE 3 Some objects that can be synthesized from three gliders. Row 1: Spaceships: LWSS, MWSS, HWSS. Row 2: Oscillators: Beacon, toad, pentadecathlon, pulsar. Row 3: Still lifes: Tub, ship, barge, long boat. Row 4: Still lifes: Cigar (mango), long barge, very long boat, hat. Row 5: Still lifes: 14.30 (half-bakery), 14.79 (paperclip).

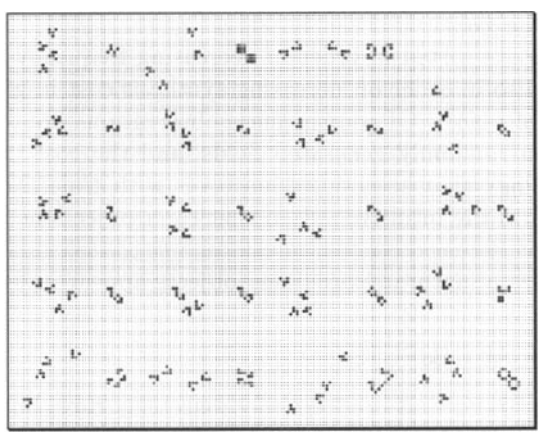

FIGURE 4 Some objects that can be synthesized from four gliders. Row 1: Oscillators: Clock, figure-8, Spark-coil. Row 2: Snake, carrier, long snake, long ship. Row 3: Shillelagh, tub with tail, canoe, 9.4 (integral). Row 4: 9.5 (up-boat with tail), 9.6 (down-boat with tail), 10.24 (bowtie), 10.25 (block on table). Row 5: 12.107, 14.507, 14.526, 16.1749 (bi-pond).

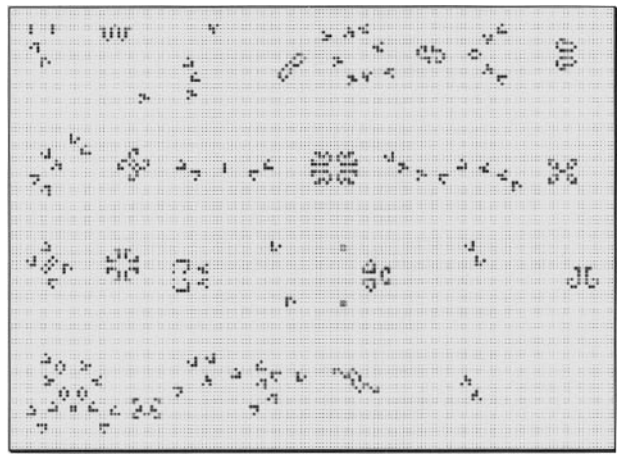

FIGURE 5 Several symmetrical objects, part 1.

2.3 RESCUABLE OBJECTS

Many frequently occurring unstable patterns bear a very close resemblance to rare objects. By adding small sparks or other alterations, cheap collisions can be modified to produce more sophisticated objects.

In figure 8 are some examples, most by David Buckingham.

The Hertz oscillator is a prime example of this method. Before Buckingham generated this synthesis, it was conjectured that billiard-table oscillators (those with stable outer shells, and variable cells only in the interior) would be difficult, if not impossible to synthesize. Remarkably, there is a very common object, that can be made from as few as two gliders, that strongly resembles the interior of a Hertz oscillator. By adding induction coils on all sides at appropriate times, this mechanism can be tamed, yielding this extremely unnatural object from a mere eleven gliders.

The cuphook is another, simpler example, in which a still life is grown into the oscillator core, while a suitable induction coil is added at the same time (see fig. 9).

2.4 SIMULTANEOUS SYNTHESES

For many complicated objects, the synthesis task can be made a little easier by performing the operation in several smaller steps. Unfortunately, for many oscillators, this is not possible, and the entire object must be constructed in such a way that all active components are activated simultaneously. This often poses great logistical problems, since it is often difficult to bring all the necessary components to bear within a small area (see fig. 10).

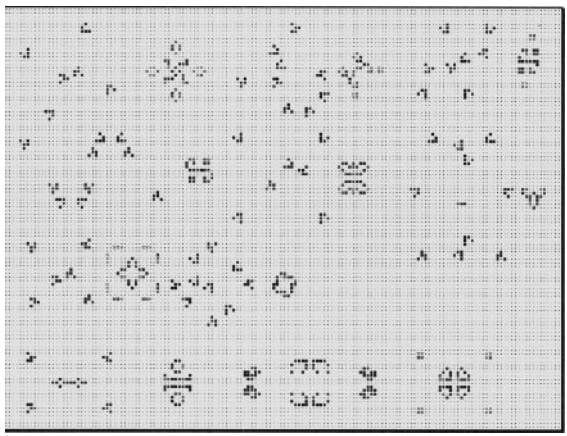

FIGURE 6 Several symmetrical objects, part 2.

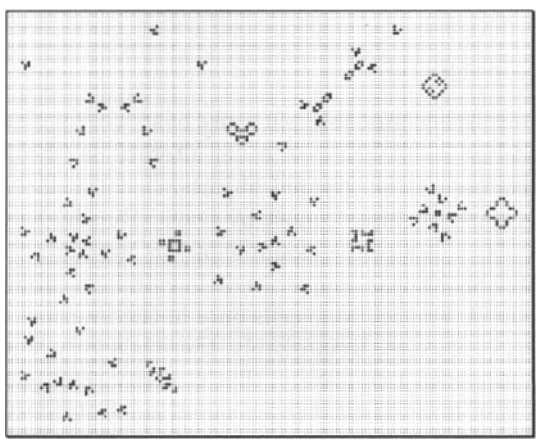

FIGURE 7 Several symmetrical objects, part 3.

2.5 INCREMENTAL SYNTHESES

Fortunately, most objects do not have to be constructed all at once. They can be built incrementally, starting with simpler objects, and making minor alterations. For example, of the 1353 fifteen-cell still lifes, around 97% can be built in this way.

Buckingham developed hundreds of such tools while attempting to synthesize all objects up to 14 cells, and many additional tools have been developed subsequently (see figs. 11–12).

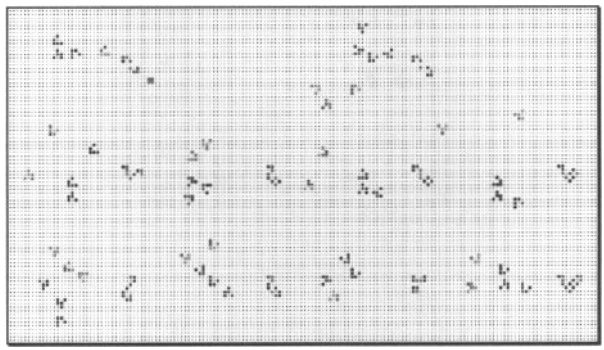

FIGURE 8 Several rescued objects (gliders performing rescue are shown in lighter color). Row 1: Bipole, tripole. Row 2: 9.8, 9.9, 10.17, 10.6. Row 3: 10.21, 10.23, 10.25, 13.96.

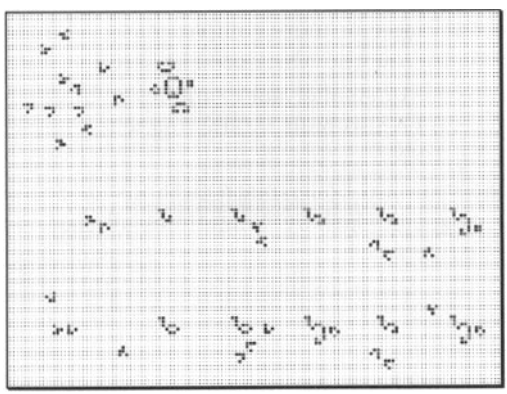

FIGURE 9 Natural Billiard-Table Oscillators. Row 1: Hertz oscillator: 3 gliders make core, 2 gliders make inducting house on top, 1 glider makes inducting boat on left, 2 gliders make inducting block on right, 2 gliders make inducting house on bottom. Row 2: Cuphook with tail: 2 gliders make eater, which 2 more gliders change into claw with tail; 2 gliders change claw to cuphook and 1 glider adds low inducting block. Row 3: Cuphook with tail: 4 gliders make beehive with tail; 2 gliders change beehive to cuphook and 1 glider adds high inducting boat. Also, variant of row 2 method also with high inducting boat.

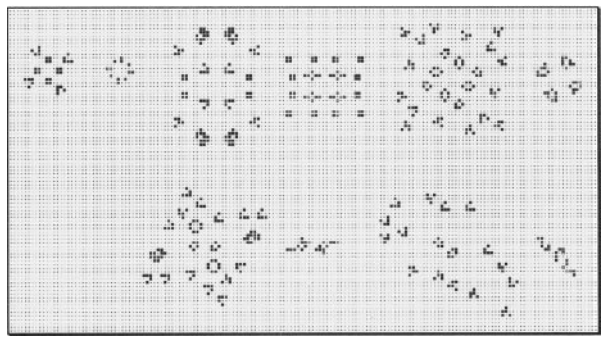

FIGURE 10 Simultaneously constructed oscillators. Row 1: Phoenix (period 2), Two pulsars hassled by 16 blocks (period 10), Achim Flammenkamp's 20-bit period 8. Row 2: A 20-bit period 8, "Biting off more than they can chew" (period 4).

Particularly easy to construct are pseudo-objects, that are aggregates of two or more objects immediately adjacent to one another that do not touch or otherwise interact. Normally, constructing two objects in such close quarters would require special techniques; however, in most cases, it is easy to start with one object (usually the larger or more complex one), add a small wart such as a block or boat, and then grow the wart into the second object. Of the over 2734 pseudo-still lifes of 16 cells or smaller, for example, all but one can be synthesized, and all up to 15 cells can be also built by starting with smaller object and adding the larger one (see fig. 13).

Occasionally, the standard boiler-plate methods need to be modified because unwanted protrusions get in the way. Sometimes all that is needed is to adjust the position of a glider or spark. At other times, a completely different method must be developed. The following example shows two similar pseudo-objects, one of which uses a standard method, and the other that uses a heavily modified version of the same method.

The first pseudo-objects, an up-tripole above 10.16, requires 31 gliders. It uses the standard method for adding a 10.16 to an existing object (see fig. 14).

The second pseudo-object, a down-tripole above 10.16, requires 49 gliders. It uses the same method, but it cannot be done in the same order because the snake cannot be added once the carrier is in place. Instead, the snake must be added first, and even then an indirect method must be used, due to the snake's proximity to the tripole. Furthermore, once the snake is in place, the carrier cannot be added the usual way. The final synthesis uses the method in figure 15.

Perhaps the most complex syntheses ever attempted are of complex billiard-table oscillators. Unlike the Hertz oscillator mentioned earlier, most billiard tables contain structures that are extremely unnatural, and difficult to

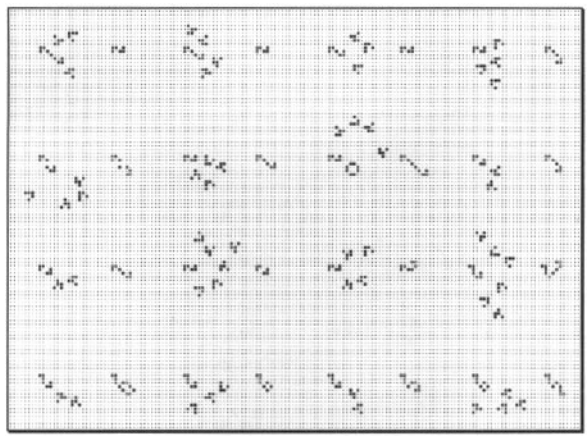

FIGURE 11 Various tools for expansion or modification of still lifes. Row 1: Shrink snake by 3 or more, 2, and 1; flip snake and grow by 2. Row 2: Grow barber-pole or snake by 1; grow snake by 2, and 4; change carrier to canoe. Row 3: Change carrier to very long snake; change snake to carrier and shillelagh; change eater head to cis-hook. Row 4: Change eater head to cigar, tub, and claw; shorten tub and add tail.

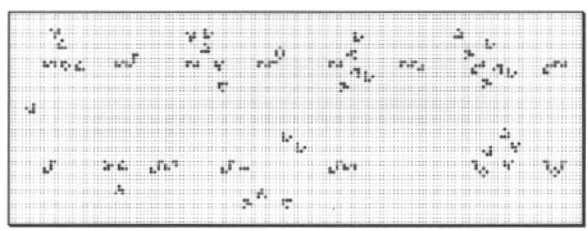

FIGURE 12 Various tools for adding pieces to still lifes. Row 1: Add siamese eater, beehive with tail, carrier, and snake. Row 2: Add corner-connected carrier and snake; and tail.

synthesize. The following is Buckingham's synthesis of the period 3 Hustler from 89 gliders, and the derivative, period 4 Hustler II from 159 gliders (Hustler plus 70 additional gliders). These syntheses are as complex as they are because they all operate on an oscillator core that is actually running; this is analogous to performing open heart surgery to replace a heart valve while the heart is still beating (see figs. 16 and 17).

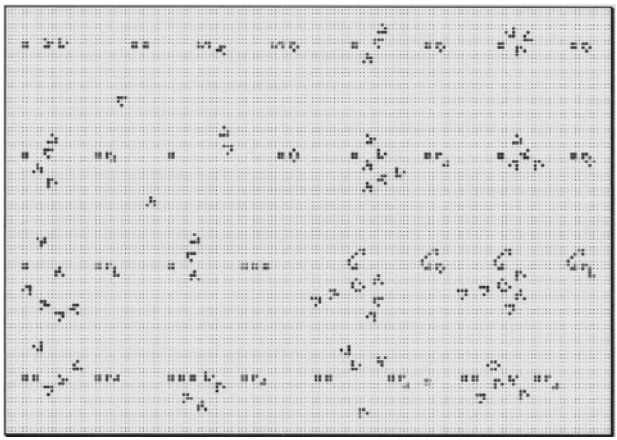

FIGURE 13 Various tools for construction of pseudo-objects. Row 1: Add block, and add boat in three different ways (to snake, from below, and from above). Row 2: Add ship, beehive, beacon, and long-boat. Row 3: Add eater, block-on-block; also, add boat and eater in a difficult geometry (on head of 10.20). Row 4: Convert block(s) to snake, trans-carrier, cis-carrier, and beacon.

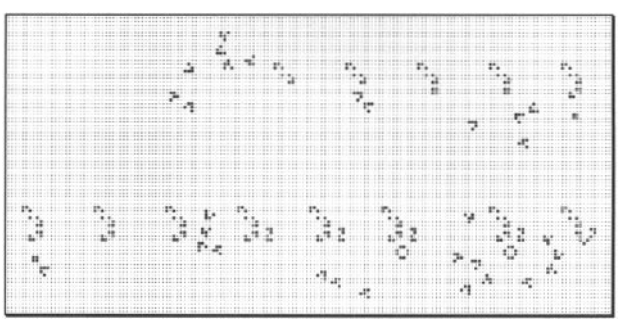

FIGURE 14 Up tripole above 10.16. Step 1. 7 gliders make a tripole. Step 2. 2 gliders add block to tripole. Step 3. 4 gliders turn the block to a cis-carrier (leaving a spurious block). Step 4. 1 glider removes spurious block. Step 5. 4 gliders place a nearby snake. Step 6. 3 gliders place a nearby pond. Step 7. 9 gliders and the pond weld the end of the snake to the side of the carrier.

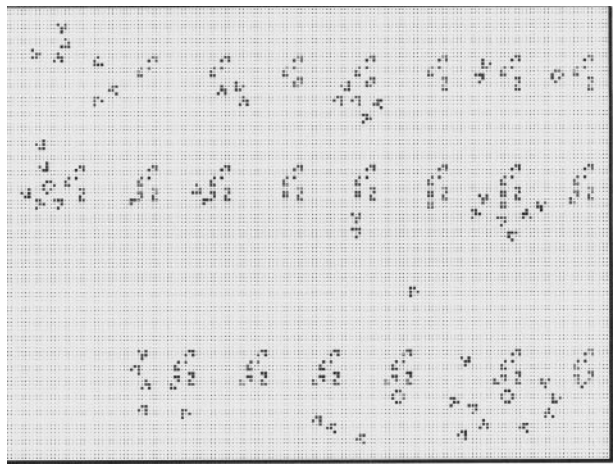

FIGURE 15 Down tripole above 10.16. Step 1. 7 gliders make a tripole. Step 2. 3 gliders add a nearby ship. Step 3. 5 gliders turn the ship into a snake. Step 4. 2 gliders add a nearby loaf. Step 5. 5 gliders and the loaf attach an eater to the tripole. Step 6. 1 glider turns the eater into block. Step 7. 3 gliders add another block. Step 8. 6 gliders turn the two blocks into a trans-carrier (this normally takes 4 gliders; the extra ones are needed to ensure that the resulting explosion bounces harmlessly off the snake). Step 9. 5 gliders flip the trans-carrier into a cis-carrier (the remainder of the synthesis is normal). Step 10. 2 gliders place a nearby pond. Step 11. 9 gliders and the pond weld the end of the snake to the side of the carrier. (As this goes to press, an improved method has been found that combines steps 2–6; unfortunately, it is too late to include it here.)

2.6 MYSTERIOUS SYNTHESES

Last, but not least, are those syntheses that defy analysis. These appear to use principles of black magic, rather than chemistry; various components, none of which is remotely connected to the final object, all come together at once, and suddenly, with no warning, cause the object to erupt at once.

Someone like Buckingham who has spent 20 years doing this could probably explain these; they are still a complete mystery to the author, who has only been actively working with this technology for around 3 years (see fig. 18).

2.7 COMPUTATIONAL MACHINERY

Life permits computational machinery to be built, using gliders and other spaceships as signals, and using other active components such as still lifes, oscillators, glider guns, and puffer trains to create, manipulate, and destroy such signals. In order to build complex machinery, it is necessary for all such components to be

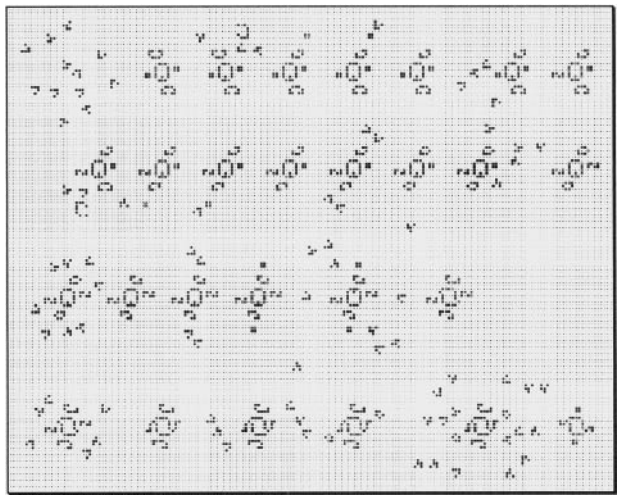

FIGURE 16 Hustler from 89 gliders. Row 1: Step 1. 12 gliders make Hertz Oscillator bounded by two houses and two blocks. Step 2. 3 gliders and one LWSS (that costs 3 more) change top house on one side into a bookend on the other, plus spurious block. Step 3. 1 glider removes the block. Step 4. 4 gliders turn left block into a snake. Row 2: Steps 5 and 6. Like steps 2 and 3 but on bottom. Step 7. 2 gliders turn each of the bookends into buns. Step 8. 5 gliders turn right block into a snake. Row 3: Step 9. 4 gliders on each side turn bun into bookend facing in opposite direction. Step 10. 2 gliders on each side add a nearby block. Step 11. 5 gliders plus the block on each side puff out the side of the interior and simultaneously lengthen the supporting bookend, yielding a Hustler supported by two snakes and two long bookends (this is the key step). Row 4: Step 12. 3 gliders on each side turn the inducting snake into an attached tail. Step 13. 3 gliders on each side add a nearby boat. Step 14. 8 gliders plus the boat on each side flip the attached tail to an attached hook, and simultaneously change the inducting long bookend into a block. (This yields a Hustler with the minimum population.)

constructable from gliders. Fortunately, the most commonly used components are extremely easy to construct from gliders (see figs. 19 and 20).

3 DEVELOPMENT OF SYNTHESIS TECHNOLOGIES

As useful as all of the above tools are, they merely provide methods for building objects from a small number of predefined building blocks. In order to achieve ever-increasing orders of complexity, it is necessary to have techniques for the development of new building blocks.

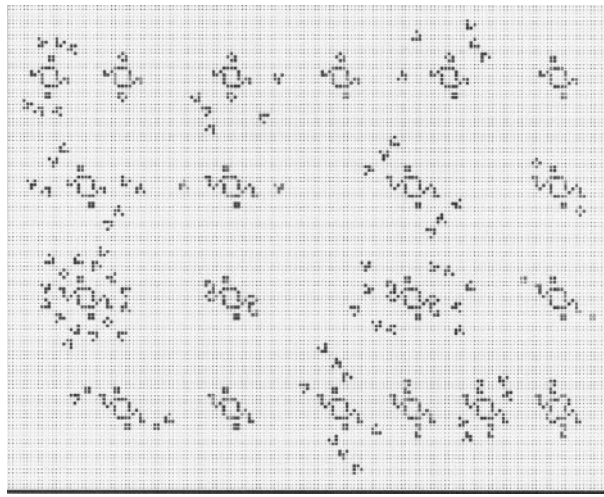

FIGURE 17 Hustler II from 159 gliders (Hustler plus 70 gliders). Step 1. 89 gliders make Hustler (fig. 16). Row 1: Step 2. 3 gliders on each side change inducting block into a boat. Step 3. 5 gliders on bottom change boat into block on opposite side. Step 4. 5 gliders on top change boat into block on opposite side. Row 2: Step 5. 4 gliders on each side change hook into bit with tail, plus an escaping glider. Step 6. 1 glider on each side (not shown) removes the escaping glider. Step 7. 3 gliders on each side add a nearby tub. Row 3: Step 8. 7 gliders and tub on each side change bit with tail into bookend with siamese beehive. Step 9. 5 gliders on each side bookend with siamese beehive into tail (without the bit) and a spurious block. Row 4: Step 10. 1 glider on each side removes the spurious block. Step 11. 4 gliders on each side change inducting block into a snake. Step 12. 2 gliders on each side puff out the corner, yielding a Hustler II (this is the key step).

3.1 SEARCH FOR PREDECESSORS

The most important step in synthesizing any object is the identification of a likely predecessor that will evolve into the object, but is also itself possible to synthesize.

The first example below shows the conversion of a block into a boat, something that is often done when building induction coils, like those found in pseudo-objects or billiard-table oscillators. Since all cells in a block are saturated (having the maximum three living neighbors), and allow no additional neighbors, anything that is attached to the block causes the adjacent block cells to immediately die. The only way to preserve the inducting front of the block, and to simultaneously turn the back into a boat, is to attach a single cell diagonally to a back corner of the block. Unfortunately, any new cell that provides the necessary additional neighbors for the birth will also cause additional births that will kill

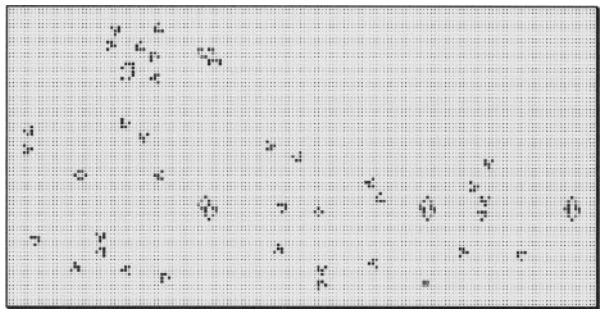

FIGURE 18 Some of Buckingham's more mysterious syntheses. Row 1: 14.95 from 9 gliders (6 gliders plus LWSS). Row 2: 14.35 from 13 gliders, 14.78 from 13 gliders, 14.77 from 7 gliders. (Buckingham referred to 14.35 as "the still life from Hell," since the previous synthesis required over a hundred gliders, and after optimization was still a massive 33 gliders before he found the above synthesis.)

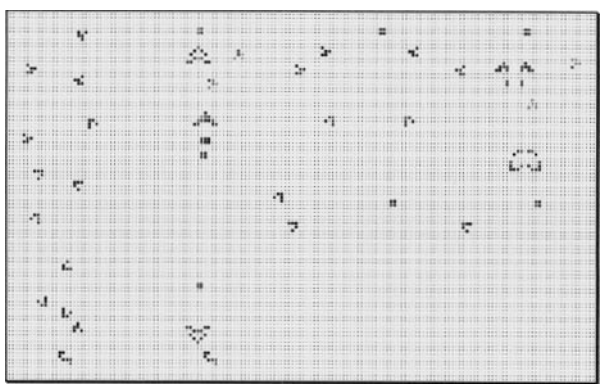

FIGURE 19 Some glider guns. Row 1: Period 30 glider gun, period 46 glider gun. Row 2: Buckaroo (period 30 glider-reflector).

the block. The solution to this is to cause a wavefront that will totally induct the back of the block for the one generation while the "new" cell does its job to cause the one desired birth. Once this fact is realized, any number of different implementations can be generated, all of which work in essentially the same way (see fig. 21).

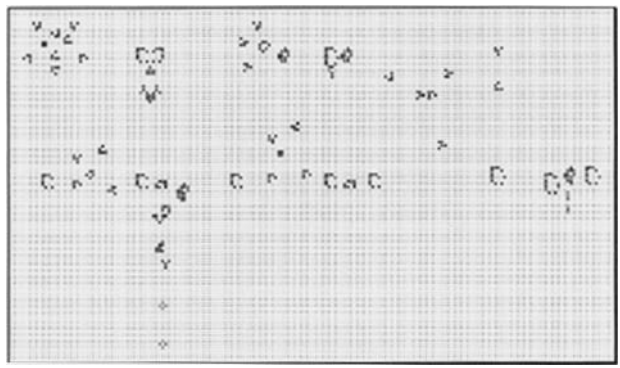

FIGURE 20 Some puffer trains. Row 1: Schick Ship (period 12), Tim Coe's puffer (period 16). Row 2: Buckingham's B-heptomino puffer (period 24), Gosper's B-heptomino puffer (period 20).

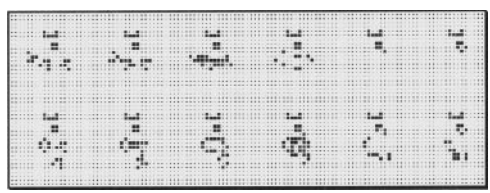

FIGURE 21 Two ways of converting a block to a boat (showing each generation).

3.2 USE OF SPARKS

Many of the simpler methods involve applying one or more simple "sparks." One of the nice features of Life is that the majority of random patterns, and indeed of collisions of gliders and small objects, results in small numbers of disconnected cells that quickly die. This allows for very carefully controlled modifications, like "increase the neighborhood of this cell by one for just one step" by having a nearby collision supply an isolated dying cell to appear at the appropriate time. Due to the large abundance of dying interactions, most of the common sparks can be made from as few as three gliders in hundreds of different ways, allowing for great flexibility in mixing of sparks together. In many cases, if a standard tool fails to work in a particular situation because of an awkward geometry, the same tool can be made to work by simply generating a vital spark using a different collision.

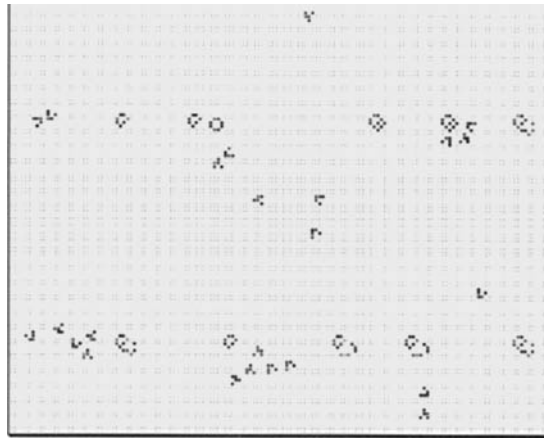

FIGURE 22 Three ways of making Mold. Row 1: Incremental Mold synthesis: Step 1. Create loaf (or any object with a loaf-like bonding site). Step 2. Two groups of three gliders each from two sparks that attach a Siamese barge to the loaf. Step 3. Three gliders change the barge to Mold. Row 2: Atomic synthesis of Mold from 5 gliders, plus improved incremental Mold synthesis: Step 1: Same as step 1 above (not shown). Step 2: Groups of two and three gliders add two sparks that attach Jam to the loaf. Step 3: Three gliders change the Jam into Mold.

3.3 CONTROLLED EXPLOSIONS

One technique that is frequently used is the creation of explosions, or "shaped charges," that cause a large number of short-lived cells to appear simultaneously. These are often used to momentarily prevent births (as in the previous block-to-boat example), or to attach something in an unique way.

For example, Mold is a simple, period 4 oscillator, that grows on a loaf. The basic oscillator can be built very simply. However, Mold may also grow on many objects that have loaf-shaped bonding sites. To build such oscillators, a method is needed to add Mold to an existing loaf. This problem stumped the author for over a year, until it was noticed that it is not necessary to attach the Mold directly; and the problem can be broken into two steps: attaching a Siamese barge to the loaf, and then changing the barge into Mold. The second step is extremely simple. The first requires adding hinges to the loaf that grab hold of each other. To this end, two explosions that are one step out of phase do the trick.

Since this chapter was first written, the author has found an improved Mold synthesis, based on Buckingham's synthesis of the similar oscillator Jam (another oscillator that forms on a loaf; see fig. 22).

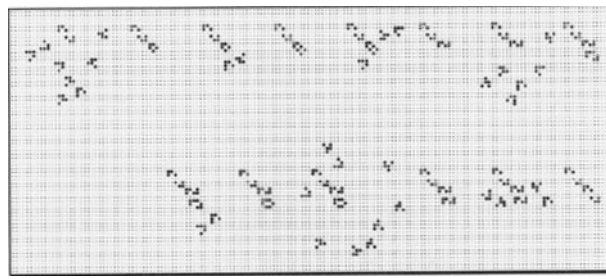

FIGURE 23 Tying a bipole. Row 1: Step 1. Create the base object (bipole, not shown). Step 2. 8 gliders tie a boat (there are many ways to do this). Step 3. 2 gliders turn boat into long boat. Step 4. 3 gliders turn long boat into snake. Step 5. 6 gliders add a bookend. Row 2: Step 6. 2 gliders turn bookend into bun. Step 7. 9 gliders turn the bun into a snake. (This is the most fragile step.) Step 8. 4 gliders remove the "amber," yielding a bipole.

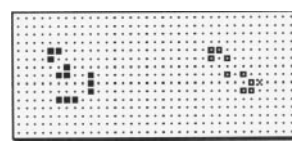

FIGURE 24 Two blinkers turn a carrier into a tripole plus a toxic cell.

Another long-unsolved problem was that of tying a bipole to the corner of an existing object. There was no known method to grow a barber pole from an existing object. Several years ago, Heinrich Koenig discovered a reaction in a random broth experiment that yielded a mechanism to turn a ship into a quadpole. This also yields all larger barber pole oscillators, using Buckingham's mechanism to grow a barber pole. Unfortunately, the two smallest barber poles (bipole and tripole) remained elusive.

If one looks at two side-by-side snakes, these resemble a bipole frozen in amber, so to speak. This suggests a synthesis in which two snakes are added first, and then the "amber" is removed. It turns out that this can be made to work; unfortunately, it is not suitable for growing pseudo-objects containing bipoles (see fig. 23).

Since this chapter was first written, I have also found an extremely simple way of adding a tripole; this is based on the following observation: two blinkers can convert a carrier into a tripole; unfortunately there is a single toxic cell attached (see fig. 24).

By adding a sufficiently convoluted explosion, it is possible to suppress the formation of this toxic cell, yielding a viable tripole from a carrier. Since a carrier

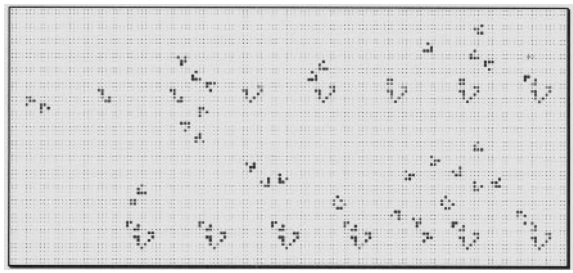

FIGURE 25 Up-tripole above 10.16 from 27 gliders. Row 1: Step 1. 2 gliders make an eater. Step 2. 6 gliders turn the eater into a 10.16. Step 3. 2 gliders add an inducting block. Step 4. 4 gliders turn the block to a cis-carrier (leaving a spurious block). Row 2: Step 5. 1 glider removes the spurious block. Step 6. 3 glider add a nearby loaf. Step 7. 9 gliders and loaf turn carrier into tripole.

can easily be tied to another object, or added as an induction coil to another object (as shown in figs. 14 and 15), it is now possible to likewise add tied and inducted tripoles. (As this goes to press, a similar method has been found to change a carrier to a bipole. Unfortunately, it is too late to include it here.)

This required explosion was found by brute-force trial and error. Often, this is a method of last resort in finding suitable syntheses, and is somewhat analogous to solving the following problem: *There is a broken vase on the floor that must be reassembled. Find a placement of various sticks of dynamite that will blow all the pieces together in exactly the right time and position so that the vase will re-form.* Fortunately, in Life one can repeat such experiments multiple times (see fig. 25).

4 SYNTHESES IN OTHER CELLULAR AUTOMATA

Most of the techniques discussed previously apply to many other cellular automata as well. The individual details will, of course, vary, but the philosophies remain similar.

4.1 3–4 LIFE

The 3–4 Life rule is an interesting variant first studied by the MIT AI group in the 1970s, and described in *Lifeline* (see issues 4–9 and 11 [5]). It is defined in a Moore neighborhood, with both birth and survival on 3 or 4. Its most notable features are the almost total absence of small still lifes, and the fact that random patterns do not fragment like in Life; instead, many expand randomly, and tend to grow chaotically without limit.

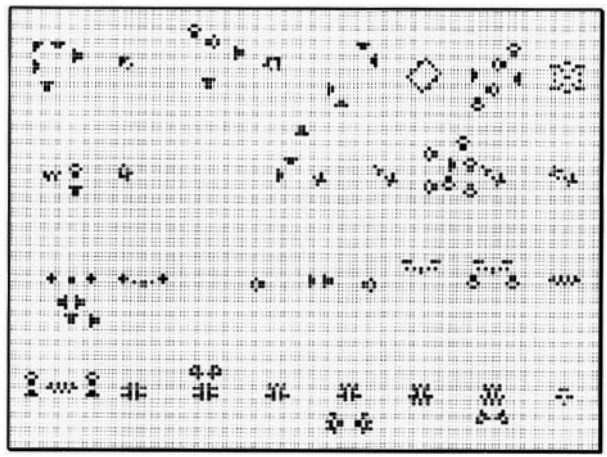

FIGURE 26 Various 3–4 Life syntheses. Row 1: The first two are syntheses of two rare naturally occurring spaceships, in both cases by adding a one-cell spark to a common object; the next two are rather remarkable natural symmetrical objects. Row 2: A rare naturally occurring oscillator formed from a common one, and two tools used to attach a clock to an existing oscillator. Rows 3 + 4: The final step in building Wainwright's period 3 block-hassler, plus a 6-step synthesis of an extremely rare 8-cell flip-flop.

The lack of survival on 2 makes still lifes other than the block impossible, except for huge fortress-like objects 36 cells and larger. Instead, the vast majority of repeating patterns are period 2 oscillators. The inclusion of birth on 4 makes pseudo-objects impossible.

The major problem with syntheses in 3–4 Life (see fig. 26) is the tendency for unstable patterns to fulminate rather than fragment as they do in Life. This often causes unbounded growth, and makes accessible sparks much harder to come by. Nevertheless, many complex syntheses are still possible.

5 CONCLUSIONS

This technology is still in relative infancy, but it demonstrates that objects of high complexity could be constructed from machinery, such as puffer trains or universal constructors. Furthermore, such objects will naturally evolve in a sufficiently large, sufficiently sparse universe [2].

6 SUGGESTIONS FOR FUTURE WORK

6.1 AUTOMATED SYNTHESIS OF SIMPLE OBJECTS

It should be possible, given some fairly simple pattern matching, to construct an expert system that would automatically suggest syntheses based on pre-defined tool templates. This would help pinpoint the deficiencies in the tool set, which could be supplemented by hand as required. This would quickly yield lists of objects whose syntheses are yet unknown, allowing energy to be channeled in the creative areas, instead of being wasted on a lot of redundant bookkeeping, as is now unfortunately the case.

6.2 AUTOMATED SYNTHESIS OF ALL STILL LIFES

One ultimate goal would be to prove that all still lifes can be synthesized, and to provide an algorithm for specifying a concrete synthesis recipe for each object (for example, a set of coordinates for a series of required gliders).

One step in this direction would be to show that all still lifes could be built by induction.

Starting with a desired still life, draw a horizontal line between two rows of the still life, so that the cells above the line must exactly match the desired still life, while the cells below the line may be anything we choose, sufficient to render the entire object stable. The goal is to weave the still life, so the area above the line is completed, while the area below the line is under construction and subject to change.

Typically, the cells below the line will consist of 3–4 rows of stabilizing cells. The row immediately above the line is in its final form, but is affected by cells below the line. The row immediately below the line is severely constrained by the row immediately above the line (see fig. 27).

The first requirement is that this model be proved to be complete; that is, for every combination of two rows (B+C) above the line, a valid combination of finite rows of cells below the line exists. Secondly, the model must be modified slightly, so that the line has a kink in it; given a corner cell (x,y), the line extends above the cell to the right, and below the cell to the right. This corner cell is itself considered below the line. This model must also be proven complete (see fig. 28).

Third, a construction sequence must be found for every possible combination of cells near the corner cell (*), that inverts the state of the corner cell, but does not affect any cells above the line.

Once this is done, it will be fairly simple to prove by induction that all finite still lifes could be synthesized. Starting with the corner square above the top left edges of the object, the corner advances through the object top-to-bottom, left-to-right; if the cell at the corner is in the correct state, nothing happens; otherwise, the appropriate construction is used to invert it. In either case, the corner square advances until the corner is several rows below the bottom right

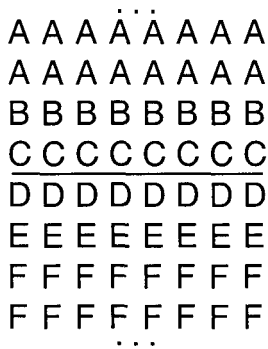

FIGURE 27 Simple "scaffolding" model. Rows A–C are part of the still life to be constructed; rows D–F are supporting structure. Row A is a completed part of still that life no longer needs to be considered. Row B is a completed part of the still life that still contributes to the neighborhood of row C. Row C is a completed part of the still life that requires external stabilization. Row D is the part of the external scaffolding which directly supports the outermost layer of the still life, row C. Rows E + F are whatever it takes to stabilize row D.

corner of the object, and the entire object is within the "completed" area; any remaining scaffolding could be removed by conventional techniques.

Unfortunately, there appear to be a very large number of possible cell combinations, so a formidable number of constructions will be needed. For the time being, the required technology is beyond the state of the art, but one can hope that this will not always be the case.

ACKNOWLEDGMENTS

The syntheses and methods described herein have been developed by many people, over the span of the last three decades.

Some of the first efforts were first published in the early 1970s in Robert T. Wainwright's Life newsletter *Lifeline* [5], which was one of the first media by which Life-enthusiasts could communicate their discoveries to one another.

The systematic application of these techniques to the synthesis of large numbers of objects, and the development of large numbers of new techniques, was first done by David Buckingham, who over the last twenty years has developed syntheses for all of the over 1000 stable, oscillating, and moving Life objects with populations up to 14 cells. In recent years, he has optimized many of these so that all of these stable objects can be constructed at a cost of no more than 1 glider per living cell, and all but about a dozen for strictly less than 1 glider per

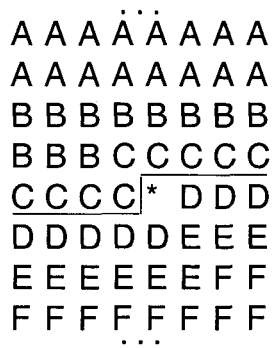

FIGURE 28 Modified "scaffolding" model.

cell. (Recently discovered bookkeeping errors indicate that one 14-cell still life costs 15 gliders, and the synthesis of one 14-cell flip-flop has been lost.)

GLOSSARY

The following terms are used in this chapter:

Breeder. A spaceship gun that emits puffers, or a puffer train that emits puffer trains or spaceship guns.

Constellation. A collection of two or more disconnected objects that all evolve from a single predecessor.

Flip-flop. An oscillator that alternates between two states.

Glider. Technically, this term refers in general to spaceships that possess glide-symmetry, but in Life it usually refers to the simplest spaceship. The term sometimes loosely includes larger spaceships.

Induction coils. Components that are adjacent to one another in such a way that they suppress all births that would occur between them.

Object. A collection of living cells that are connected, or that affect each other directly or indirectly. (When referring to syntheses of objects, this term is sometimes used in a more general sense to also include pseudo-objects or constellations.)

Oscillator. An object that reforms itself after several generations.

Pseudo-object. Two or more objects that are adjacent, but that are not connected and do not otherwise affect each other.

Pseudo-still life. A pseudo-object that remains stable.

Puffer train. A spaceship that emits still lifes, oscillators, and/or spaceships.

Spaceship. An object that reforms itself translated in space after several generations.
Spaceship gun. An oscillator that emits spaceships.
Still life. An object that has neither births nor deaths, hence remains stable.

REFERENCES

[1] Flammenkamp, Achim. Web pages. Game of Life page:
`http://www.uni-bielefeld.de/~achim/gol.html`.
Most Seen Objects page:
`http://www.uni-bielefeld.de/~achim/freq_top_life.html`.
Natural Grown Oscillators page:
`http://www.uni-bielefeld.de/~achim/oscill.html`.
Spontaneous Appeared Spaceships page:
`http://www.uni-bielefeld.de/~achim/moving.html`.
[2] Gotts, Nick. "Self-Organized Construction in Sparse Random Arrays of Conway's Game of Life." This volume.
[3] Koenig, Heinrich. Web pages. Gate of Life page:
`http://www.pentadecathlon.com/LifeInfo/LifeInfo.html`.
[4] Niemiec, Mark. Web page. Life page:
`http://home.interserv.com/~mniemiec/lifepage.htm`.
[5] Wainwright, Robert T. *Lifeline*. Self-published. Eleven issues from March 1971 to September 1973. See Robert T. Wainwright's Life page:
`http://home.earthlink.net/~hilery/life/`.

Even though much about Life has been published, most of the information about glider synthesis exists only in unpublished correspondence. The author also maintains a Web page continaing many object lists and syntheses of many objects from gliders; see the author's home page [4].

This page intentionally left blank

A Two-Dimensional Cellular Automaton Crystal with Irrational Density

David Griffeath
Dean Hickerson

We solve a problem posed recently by Gravner and Griffeath [4]: to find a finite seed A_0 of 1s for a simple $\{0,1\}$-valued cellular automaton growth model on $\mathbb{Z}^2$ such that the occupied crystal A_n after n updates spreads with a two-dimensional asymptotic shape and a provably irrational density. Our solution exhibits an initial A_0 of 2,392 cells for Conway's Game of Life from which A_n covers nT with asymptotic density $(3-\sqrt{5})/90$, where T is the triangle with vertices $(0,0)$, $(-1/4,-1/4)$, and $(1/6,0)$.

1 INTRODUCTION

In "Cellular Automaton Growth on $\mathbb{Z}^2$: Theorems, Examples, and Problems" [4], Gravner and Griffeath recently presented a mathematical framework for the study of Cellular Automata (CA) crystal growth and asymptotic shape, focusing on two-dimensional dynamics. For simplicity, at any discrete time n, each lattice site is assumed to be either *empty* (0) or *occupied* (1). Under a wide variety of simple CA rules, starting from most finite seeds A_0 of initial 1s, the set A_n of

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

occupied sites after n updates grows linearly in each dimension, attaining an asymptotic density ρ within a limit shape L:

$$n^{-1}A_n \to \rho \cdot 1_L \,. \tag{1}$$

This phenomenology is developed rigorously in Gravner and Griffeath [4] for *Threshold Growth*, a class of monotone solidification automata (in which case $\rho = 1$), and for various nonmonotone CA which evolve recursively.

The coarse-grain crystal geometry of models which do not fill the lattice completely is captured by their *asymptotic density*, as precisely formulated in Gravner and Griffeath [4]. It may happen that a varying "hydrodynamic" profile $\rho(x)$ emerges over the normalized support L of the crystal. The most common scenario, however, would appear to be eq. (1), with some constant density ρ throughout L.

All the asymptotic densities identified by Gravner and Griffeath are rational, corresponding to growth which is either exactly periodic in space and time, or nearly so. For instance, it is shown that *Exactly 1 Solidification*, in which an empty cell permanently joins the crystal if exactly one of its eight nearest (Moore) neighbors is occupied, fills the plane with density 4/9 starting from a singleton. By contrast, the analogous *Exactly 2* solidification rule spreads chaotically from a dyad (cf. fig. 9 of Gravner and Griffeath [4]). The asymptotic shape is presumably $\mathcal{D} = \{x \in \mathbb{R}^2 : ||x||_1 \leq 1\}$, and there appears to be a characteristic uniform density ρ throughout, but rigorous analysis of this model seems well beyond the reach of current mathematical techniques. Indeed, for the vast majority of CA rules satisfying eq. (1), growth seems intractably aperiodic, in which case its density ρ is presumably irrational. In this context, Gravner and Griffeath posed the following

Problem 8: *Find an elementary CA on $\mathbb{Z}^2$ with a computable asymptotic density which is irrational.*

Shortly thereafter, Dean Hickerson suggested that such an example could be built in Conway's Game of Life (GoL) by combining two of his creations from the early 1990s: the *Irrat5* seed [6], which grows linearly with an irrational multiplier and the *Stifled Breeder* [7], which produces sporadic two-dimensional growth controlled by intermittent input signals. Our object here is to present the detailed solution to Problem 8 in a manner accessible to CA enthusiasts outside the core of Life experts, thereby illustrating some state-of-the-art constructive methods for the most thoroughly studied of all cellular automaton rules.

We will assume familiarity with the Game of Life's basic phenomenology and early history, as beautifully described in *Winning Ways for Your Mathematical Plays* [1, vol. 2, ch. 25]. An excellent World Wide Web resource for background information is Paul Callahan's site, *Patterns, Programs, and Links for Conway's Game of Life* [2]. Recall some of the ubiquitous ingredients in Life constructions (shown in fig. 1), the *still lifes* (invariant configurations) block and boat, and the

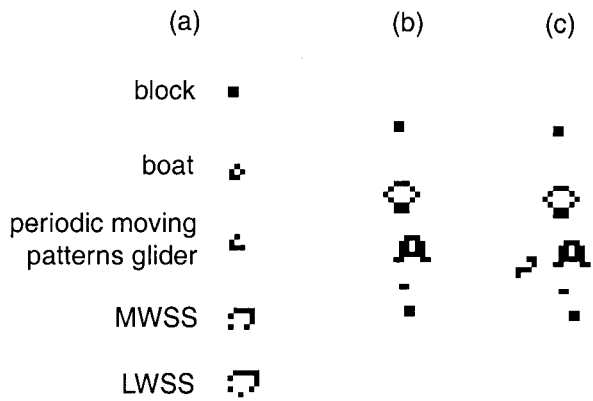

FIGURE 1 (a) Still lifes, (b) glider gun, and (c) stifled glider gun.

periodic moving patterns glider, LWSS and MWSS. (*LWSS* and *MWSS* stand for lightweight and middleweight spaceships, respectively.) Our construction employs a fairly elaborate orchestration of collision rules between these stationary and ballistic structures, along with other more complicated ingredients. But the overriding strategy is a variant of the earliest approach to quadratic population growth, based on Gosper's 1970 discovery of the celebrated glider gun (fig. 1(b)), which emits a steady stream of gliders, one every 30 generations (traveling SW in the case shown). A *breeder* is any pattern which grows quadratically by creating a steady stream of copies of a second object, each of which creates a stream of a third. In the early 1970s Gosper also exhibited the first breeder, a 4060-cell traveling "puffer" which deposits a trail of glider guns, each of which then emits gliders. Several smaller breeder seeds discovered more recently are cataloged in Callahan [2].

In order to control the density of our two-dimensional crystal A_n, the idea is to produce a mix of the glider gun and stifled glider gun (fig. 1(c)), this latter structure differing only by the presence of an *eater* along the lower left side which destroys each glider and then restores itself before the next arrives. Hickerson's original *Stifled Breeder* contains a puffer which produces a stream of stifled glider guns, each of which converts to a true glider gun once its eater is deleted by collision with an appropriate external glider. For our purposes, then, the strategy is to subject the puffer's stifled breeders to an external source of gliders with irrational density, thereby producing steady streams of gliders at an irrational rate. The design of the desired glider source is based on Hickerson's *Irrat5* pattern, and emulates a sequence of 0s and 1s known as the *golden string*.

The organization of the remainder of the chapter follows. In section 2 we offer a self-contained derivation of known recursive and limiting properties of

the golden string, used to relate the asymptotic density of our Life crystal to the golden mean $\phi = (1+\sqrt{5})/2$. Then in section 3 we construct *Irrat5'*, a pattern which emits a stream of gliders at rate $(3-\sqrt{5})/240$. Section 4 completes the solution to Problem 8 of Gravner and Griffeath [4] by showing how interaction between the above glider stream and a stifled breeder gives rise to limit shape T, the triangle with vertices $(0,0)$, $(-1/4,-1/4)$, and $(1/6,0)$, and an asymptotic population density $(3-\sqrt{5})/90$ on T. Finally, section 5 describes essential online resources for interactive visualization of our construction. A series of 18 experiments may be downloaded, and executed using one of several public domain GoL engines for various computer platforms. The first nine demos illustrate key collisions, subject to suitable positioning in space and time:

$$\begin{array}{lcl}
(C1) & glider + block & = \emptyset \\
(C2) & glider + glider & = block \\
(C3) & glider + glider + glider & = MWSS \\
(C4) & MWSS + glider & = block \\
(C5) & MWSS + glider & = \emptyset \\
(C6) & MWSS + boat & = glider \\
(C7) & glider + glider + glider & = LWSS \\
(C8) & LWSS + glider & = \emptyset \\
(C9) & glider + stifled\ glider\ gun & = glider\ gun
\end{array} \tag{2}$$

The remaining demos synthesize the initial seed A_0 for the solution of Problem 8, beginning with the puffers and guns which combine to form the *Irrat5'* glider generator, then adding a stifled breeder and additional guns, and culminating in *Seed*, the pattern of size 2,392 which grows our desired crystal.

2 THE GOLDEN STRING

Recall that the Fibonacci numbers F_n are generated recursively by

$$F_0 = 0, \quad F_1 = 1, \text{ and } F_{n+1} = F_n + F_{n-1} \text{ for } n \geq 1\,.$$

Produce a sequence of strings by starting with 1 and then repeatedly applying the mapping T which replaces each 0 by 1, and each 1 by 10:

$$1 \xrightarrow{T} 10 \xrightarrow{T} 101 \xrightarrow{T} 10110 \xrightarrow{T} 10110101 \xrightarrow{T} \ldots\,.$$

For $n \geq 1$, let $f_n = (f_n(i))$ be the nth string in this sequence. If $\oplus$ symbolizes concatenation, evidently $f_3 = f_2 \oplus f_1$, and then for $n \geq 3$,

$$f_{n+1} = Tf = T(f_{n-1} \oplus f_{n-2}) = f_n \oplus f_{n-1}\,.$$

Hence, the recursion gives consistent initial segments of an infinite *golden string* (cf. Schroeder [10], or Exercise 36 on p.86 of Knuth [9], or Knott [8])

$$f = 10110101101101011010 1\ldots ,$$

with $Tf = f$. Denote by N_n, N_0^1, and N_n^0 the length of f_n, its number of 1s, and its number of 0s, respectively. One easily checks by induction that

$$N_n = F_{n+1}, \quad N_n^1 = F_n, \text{ and } N_n^0 = F_{n-1}\,.$$

The Fibonacci numbers satisfy the well-known formula $F_n = (\phi^n - (-\phi^{-1})^n)/\sqrt{5}$, where $\phi = (\sqrt{5}+1)/2$ is the Golden Ratio. Since f is "fractal," it splits into finite blocks having proportions of 1s of the form F_m/F_{m+1}, with $m \geq m_0$ large. Hence, the asymptotic densities of 1s and 0s equal $\phi^{-1} = (\sqrt{5}-1)/2 \approx 0.618$, and $\phi^{-2} = (3-\sqrt{5})/2 \approx 0.382$, respectively.

For our purposes, another equivalent recursive development of f is needed. Write

$$\iota_j^1 = \text{ index of the } j\text{th 1 in } f, \quad \iota_j^0 = \text{ index of the } j\text{th 0 in } f\,. \tag{3}$$

Thus, $\iota^1 = (1, 3, 4, \ldots)$ and $\iota^0 = (2, 5, 7, \ldots)$. If $\iota_j^1 = \ell$, then the 1 in the ℓth place of f is preceded by $j-1$ 1s and $\ell - j$ 0s. Thus, Tf has a 0 in the $\ell + j$th place preceded immediately by a 1, and with $\ell - 1$ 1s and $j-1$ 0s before that. Since $Tf = f$, we conclude that

$$\iota_j^0 = \iota_j^1 + j\,. \tag{4}$$

Moreover, the terms of f are uniquely determined by $\iota_1^1 = 1$, eq. (4), and the proviso that ι_j^1 is always the smallest integer not previously used in the development ι^1 or ι^0.

In fact, the terms of ι^1 or ι^0 are given by the beautiful formulae:

$$\iota_j^1 = \lfloor j\phi \rfloor\,, \qquad \iota_j^0 = \lfloor j\phi^2 \rfloor\,. \tag{5}$$

Coxeter [3] relates a nice proof of this fact during his discussion of a two-player game related to Nim. One first observes that since ϕ is irrational, with $\phi^{-1} + \phi^{-2} = 1$, the number of multiples of either ϕ or ϕ^2 less than $n+1$ always equals n. Thus, there is exactly one j so that $j\phi$ or $j\phi^2$ (but not both!) lies in $(n, n+1)$; i.e., the sequences on the right side of eq. (5) partition the positive integers. Since $\phi^2 = \phi + 1$, these sequences satisfy the same recursion as eq. (4) with the same initial conditions, so the formulae hold. They provide another proof that the densities of 1s and 0s in f equal ϕ^{-1} and ϕ^{-2}, in the strongest possible sense. In the next section we will use characterization eq. (4) to produce a stream of gliders corresponding in a precise way to the 0s of the golden string.

FIGURE 2 Irrat5′.

FIGURE 3 The MWSS puffer.

3 A STREAM OF GLIDERS WITH RATE $(3 - \sqrt{5})/240$

The Game of Life pattern *Irrat5*, constructed by Hickerson in 1991 and described in Callahan's web site [2, p. 6], produces linear population growth with an irrational multiplier. A modification of *Irrat5* better suited to the present context produces the glider stream with irrational frequency which will ultimately control the growth rate of our crystal A_n. This section details the mechanics of the output produced by the 693 cells in figure 2, which we call *Irrat5′*.

A *puffer* is any moving pattern which leaves a trail. Two components on the right side of *Irrat5′* are puffers which head east at speed 1/2, producing a westward MWSS, and a (static) boat, respectively, every 60 generations. These puffers are shown in figures 3 and 4; they are arranged so that the trail of boats is deposited a suitable distance below the MWSS stream.

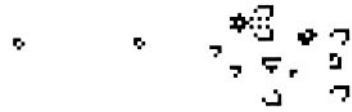

FIGURE 4 The boat puffer.

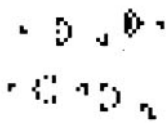

FIGURE 5 The glider gun (period 60).

At the top left of *Irrat5'*, a period 60 gun (fig. 5) produces gliders heading SE. (This object is obtained from two period 30 glider guns using reactions (C1) and (C2) of eq. (2).) At the bottom left of figure 2 is a period 120 gun (fig. 6) which emits a MWSS stream moving east. (Again, this object is a hybrid of several smaller guns, using (C3).) If the puffers are removed from *Irrat5'*, then the guns of figures 5 and 6 interact so that half of the gliders and the MWSSs annihilate, while the remaining gliders escape to the southeast, one every 120 updates.

However, each westward MWSS from the puffer of figure 3 which arrives within close proximity of the glider gun of figure 5 deletes (or *zaps*) two consecutive gliders, the first by reaction (C4), the second by (C1). In this manner, one output of the SE glider stream is suppressed, and then one eastward MWSS is allowed to escape. Each such spaceship later hits a boat left behind by the puffer of figure 4, reaction (C6) then creating a NE glider which later deletes a westward spaceship from the MWSS puffer by (C5). Thus, each westward MWSS

FIGURE 6 MWSS gun 1 (period 120).

that reaches the SE glider stream causes the deletion of a later westward MWSS. To quantify this feedback effect, for $k \geq 1$ let w_k, g_k, e_k, b_k denote the kth westward MWSS, glider, eastward MWSS, and boat produced in these respective streams. Then the initial reactions may be described thus:

w_1 stops g_1, which zaps g_2	e_1 hits b_1, and then w_3 dies
w_2 stops g_3, which zaps g_4	e_2 hits b_2, and then w_5 dies
w_3 gone, so g_5 escapes, g_6 zaps e_3	e_3 gone

Moreover, the recursive development of this sequence is dictated by the following conditions:

$$\begin{gathered} \text{Unless already gone, } w_k \text{ stops } g_{2k-1}, \text{ which zaps } g_{2k}\,. \\ \text{If } w_k \text{ is gone, then } g_{2k-1} \text{ escapes, and } g_{2k} \text{ zaps } e_k\,. \\ \text{If } e_\ell \text{ is the } k\text{th eastward MWSS not zapped by a glider} \\ \text{then } e_\ell \text{ hits } b_k \text{ after which } w_{k+\ell+1} \text{ dies.} \end{gathered} \tag{6}$$

The first two assertions of eq. (6) are clear from the preceding description, but the third requires a little calculation. With respect to a suitable Origin and with appropriate conventions for the central locations of our various static and moving GoL objects, since the MWSS puffer moves eastward at speed 1/2 and produces a new westward spaceship with speed 1/4 about every 60 updates, ship w_i is at horizontal position $30(i-j)$ at time $60(i+j)$ $(0 \leq i \leq j)$, until it experiences a collision. Also, since (suitably positioned) MWSS gun 1 produces a speed 1/2 eastward spaceship every 120 generations, and the boat stream has spatial period 30, e_ℓ hits b_k at horizontal position $30k$ at time $120\ell + 60k$. Finally, the vertical distance between the two puffers is arranged so that a (horizontal speed 1/2) NE glider takes 60 generations to travel up from a boat to the westward stream of spaceships. Choose $i = \ell + k + 1$, $j = \ell$, to see that $w_{\ell+k+1}$ dies via (C5).

The interactions in eq. (6) make it clear that all gliders g_{2k} with even index die. Calling the remaining gliders *odd*, let ν_1^0 and ν_k^1 denote the index of the kth odd glider which escapes, and which collides with a westward spaceship, respectively. (For instance, $\nu_2^1 = 2$ because g_3, the second odd glider, collides with w_2.) If $\nu_k^1 = \ell$, then $k-1$ odd gliders die and $\ell - k$ odd gliders escape before the collision of $g_{2\ell-1}$ with w_ℓ. Each of the $k-1$ previously zapped odd gliders released an eastward MWSS which led to the deletion of a westward MWSS. Then e_ℓ hits b_k, after which $w_{\ell+k+1}$ dies, so the $\ell + k + 1$st odd glider is the kth to escape. That is, $\nu_1^1 = 1$, and for $k \geq 1$,

$$\nu_k^0 = \nu_k^1 + k + 1\,. \tag{7}$$

Let $\gamma = (\gamma_k)$ be the string of 0s and 1s determined by eq. (7). Thus, $\gamma_k = 0$ if g_{2k-1} escapes. Form string f by prepending 10 to γ : $f = (1, 0, \gamma_1, \gamma_2, \ldots)$,

FIGURE 7 The stifled breeder.

make the change of variables $j = k + 1$, and define ι^0 and ι^1 as in eq. (3). Then $\iota^0_j = 2 + \nu^0_{j-1}$, and $\iota^1_{j-1} = 2 + \nu^1_{j-1}$ for $j \geq 2$, so eq. (7) implies that f satisfies recursion eq. (4) with $\iota^1_1 = 1$. Hence, f is the golden string, and for $k \geq 1$,

$$\text{glider } g_{2k-1} \text{ survives forever if and only if } f_{k+2} = 0\,. \tag{8}$$

In particular, since odd gliders arise every 120 generations, the density of times at which an SE glider escapes from *Irrat5′* is equal to $1/120\varphi^2 = (3 - \sqrt{5})/240$, as claimed.

4 CONSTRUCTION OF THE SEED

A *stifled breeder* is a flotilla of puffers which produces, every 60 generations, a stifled glider gun (of the kind shown in the Introduction) and a boat. Such a structure was devised in 1992 [7] to obtain an example of GoL population growth at rate $cn \log n$. Here, we use the 1,318-cell stifled breeder of figure 7.

If a MWSS is sent along the stifled breeder's trail of boats with the proper path and timing, then it will hit the first remaining boat, becoming a glider (by (C6)) that deletes the stifled gun's eater, thereby turning on that gun. In Hickerson [7], every gun was turned on eventually. As explained earlier, our strategy here is to turn on only a subcollection of glider guns corresponding to the irrational density of surviving gliders in the stream of the previous section. To this end, we place the stifled breeder suitably to the south and east of *Irrat5′*, and introduce two additional stationary guns, which produce a LWSS (via (C7)) and a MWSS (via (C3)), respectively, every 120 generations. (A LWSS is produced

FIGURE 8 LWSS gun (period 120).

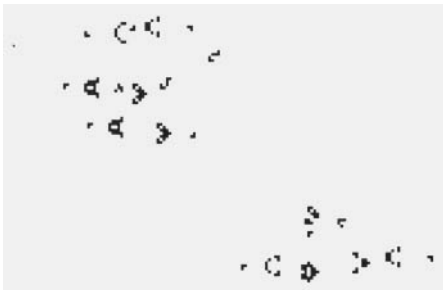

FIGURE 9 MWSS gun 2 (period 120).

every 30 generations, but 3/4 of them are subsequently deleted by interaction with two oscillators found by Robert T. Wainwright: the period 4 "MW emulator" and the period 8 "blocker.") The two guns, shown in figures 8 and 9, occupy the SW corner of our complete *Seed*, with the LWSS gun strategically placed so that its output can travel east between the halves of MWSS gun 2. Figure 10 gives the complete 2,392-cell solution to Problem 8 from Gravner and Griffeath [4].

Each MWSS heading east from the top half of the gun of figure 9 eventually hits a boat, becoming a glider and heading for a stifled glider gun deposited by the stifled breeder. Usually a corresponding LWSS from the gun of figure 8 subsequently zaps the glider, leaving that stifled gun inactive. But when a SE glider from *Irrat5′* survives as output, it zaps a spaceship from the LWSS gun near its source (by (C8)), and so the SE glider emanating from a boat reaches its eater. After reaction (C9) that stifled glider gun becomes active.

Thus, there is a one-to-one correspondence between surviving gliders from *Irrat5′* and activated glider guns in the stifled breeder's trail after the time when g_1 might have reached its target LWSS. In fact, our construction achieves an exact correspondence with the golden string f by producing first a stifled glider gun, then an active glider gun before the interaction with *Irrat5′* begins. In this way, by eq. (5), the kth gun deposited by the stifled breeder becomes active if k

FIGURE 10 The entire Seed.

has the form $\lfloor n\phi^2 \rfloor$ for some integer $n \geq 1$. In particular, the limiting density of activated guns is $\phi^{-2} = (3 - \sqrt{5})/2$.

The limit shape for A_n is now easy to derive. The eastmost point at which guns are activated advances at the speed with which outputs from MWSS gun 2 hit boats: 1/6 since the boats are 30 cells apart, and the speed 1/2 spaceships are 60 cells apart. Once a gun is activated, its glider stream proceeds SW at speed $\sqrt{2}/4$. Hence, after n generations, the outputs of the guns form a triangular region T_n with vertices at about (0,0), $(-n/4, -n/4)$, and $(n/6, 0)$. Figure 11 shows the growth after 10,000 updates. Hence, eq. (1) holds, in the sense of weak convergence of measures (cf. Gravner and Griffeath [4]), where $L = T$: the triangle with vertices (0,0), $(-1/4, -1/4)$, and $(1/6, 0)$.

To compute the asymptotic density ρ, we note that T_n can be partitioned naturally into 30×30 blocks such that each block covering an active SW stream contains four gliders (five cells each), while the remaining blocks corresponding to stifled streams of course contain none. Hence, the density is given by

$$\rho = \frac{4 \times 5}{30 \times 30} \times \text{ proportion of activated guns } = \frac{2}{90} \times \frac{3 - \sqrt{5}}{2} = \frac{3 - \sqrt{5}}{90}.$$

Note that since the puffers of our construction all move east at speed 1/2, the asymptotic shape L_H in the sense of Hausdorff convergence (again, see, Gravner and Griffeath [4]) consists of the triangle T together with a line segment from $(1/6, 0)$ to $(1/2, 0)$. Although an example with $L = L_H$ would be more aesthetic,

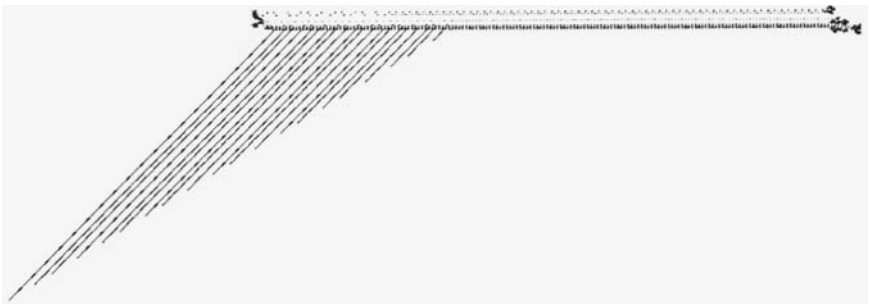

FIGURE 11 The growth of Seed from afar, after 10,000 updates.

such a construction would require even larger puffers, so the extra effort hardly seems worthwhile.

Seed grows to occupy 3/8 of the plane: namely, the fourth quadrant and the lower half of the third. Of course a reflection of *Seed* about $y = x$ fills the second quadrant and the upper half of the third. Finally, exactly the first quadrant is filled by a horizontal reflection of *Seed* together with a suitable northward eater puffer to stop gliders which reach its trail. Combining appropriate translations of these three configurations, we obtain an initial pattern for Life which fills the whole plane with density $(3 - \sqrt{5})/90$.

In conclusion, a few caveats about our solution to Problem 8 are in order. As noted in the Introduction, *Seed* is primarily intended to illustrate various, rather sophisticated, synthetic techniques whereby Life manifests some of its renowned complexity. Our crystal's structure is artificial, dimension-dependent, and (as with almost all GoL structures) *extremely* sensitive to the smallest perturbations in its initial condition. A more satisfactory answer to the original question of Gravner and Griffeath [4] would identify some simple CA rule from which most initial seeds grow with the same lattice symmetric asymptotic shape and characteristic "background ether," where that ether is fundamentally aperiodic in space and time, and has an irrational density which is nevertheless computable.

5 WEB RESOURCES

In order to thoroughly understand this chapter's construction, a computer simulator capable of interactive visualization of Life dynamics is indispensable. At ⟨`http://psoup.math.wisc.edu/Irrational/`⟩ we have posted a page with links to several programs capable of evolving Life from patterns in the standard *.lif* file format. From that same page the reader can also download configurations

illustrating each of the basic reactions (C1)–(C9) of eq. (1), as well as all the building blocks of figures 2–10 which contribute to the construction of Seed. Needless to say, the verbal descriptions of sections 1–5 are much more accessible if one simultaneously observes the dynamics of those patterns running under one of the available Life engines.

ACKNOWLEDGMENTS

We are grateful to Janko Gravner for helpful discussions, and to Johan Bontes, Paul Callahan, Al Hensel, and Andrew Trevorrow for their superb electronic GoL resources, which were indispensable in the preparation of this chapter. Thanks as well to all participants in the LifeList for their remarkable collective development of constructive methods over the years.

REFERENCES

[1] Berlekamp, E., J. Conway, and R. Guy. In *Winning Ways for Your Mathematical Plays*, vol. 2, ch. 25. New York: Academic Press, 1982

[2] Callahan, P. "Patterns, Programs, and Links for Conway's Game of Life." ⟨`http://www.cs.jhu.edu/~callahan/lifepage.html`.⟩

[3] Coxeter, H. S. M. "The Golden Section, Phyllotaxis, and Wythoff's Game." *Scripta Math.* **19** (1953): 135–143.

[4] Gravner, J., and D. Griffeath. "Cellular Automaton Growth on $\mathbf{Z}^2$: Theorems, Examples, and Problems." *Adv. Appl. Math.* **21** (1998): 241–304.

[5] Griffeath, D. "Primordial Soup Kitchen." ⟨`http://psoup.math.wisc.edu/kitchen.html`.⟩

[6] Hickerson, D. ⟨`http://www.cs.jhu.edu/~callahan/patterns/irrat5.html`⟩

[7] Hickerson, D. ⟨`http://www.cs.jhu.edu/~callahan/patterns/breedst.html`⟩

[8] Knott, R. "Fibonacci Numbers and the Golden Section," ⟨`http://www.mcs.surrey.ac.uk/Personal/R.Knott/Fibonacci/fib.html`⟩

[9] Knuth, D. *The Art of Computer Programming: Fundamental Algorithms*, vol. 1, 3rd ed. New York: Addison-Wesley, 1997.

[10] Schroeder, M. R. *Number Theory in Science and Communication, with Applications in Cryptography.* New York: Springer-Verlag, 1990.

This page intentionally left blank

Still Life Theory

Matthew Cook

1 INTRODUCTION

In Conway's *Game of Life* [2], if one starts with a large[1] array of randomly[2] set cells, then after around twenty thousand generations one will see that all motion has died down, and only stationary objects of low period remain, providing a final density of about .0287. No methods are known for proving rigorously that this behavior should occur, but it is reliably observed in simulations (see figure 1).

This brings up several interesting related questions. Why does this "freezing" occur? After everything has frozen, what is the remaining debris composed of? Is there some construction that can "eat through" the debris? If we start with an infinitely large random grid, so that all constructions appear somewhere, what

[1]For example, a torus large enough that even speed-of-light signals are unable to wrap around the torus during the run.

[2]The numbers mentioned above are for initial densities around one half. Much higher or lower densities will converge more quickly to a sparser result [3].

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

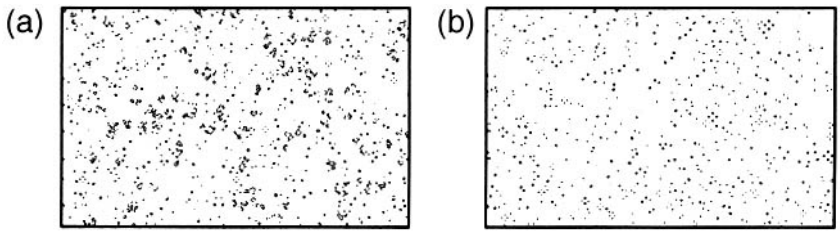

FIGURE 1 (a) After 500 steps, active areas are still common. (b) Eventually all activity subsides, leaving only debris.

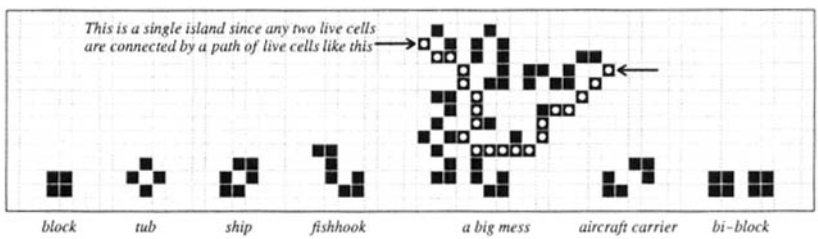

FIGURE 2 Some stable objects.

will the long term behavior be? It seems clear that knowing the composition of typical debris is central to many such questions.

Much effort has gone into analyzing the objects that occur in such stationary debris, as well as into determining what stationary objects can exist at all in Life [4, 8]. Both of these endeavors depend on having some notion of what an "object" is in the first place. One simple notion is that of an *island*, a maximal set of live cells connected to each other by paths of purely live cells. But many common objects, such as the "aircraft carrier," are not connected so strongly. They are composed of more than one island, but we think of them as a single object anyway, since their constituent islands are not separately stable.

Any pattern that is *stable* (has period one, i.e., does not change over time) is called a *still life*. Since a collection of stable objects can satisfy this definition, the term *strict still life* is used to refer to a single indivisible stable object, and *pseudo still life* is used to refer to a stable pattern that is composed of distinct strict still lifes. For example, the bi-block is a pseudo still life, since it is composed of two blocks, but the aircraft carrier is a strict still life, since its islands are not stable on their own. The entirety of figure 2 is stable, but it is clearly a pseudo still life, since it can be separated into a block, tub, ship, and so on. The distinction

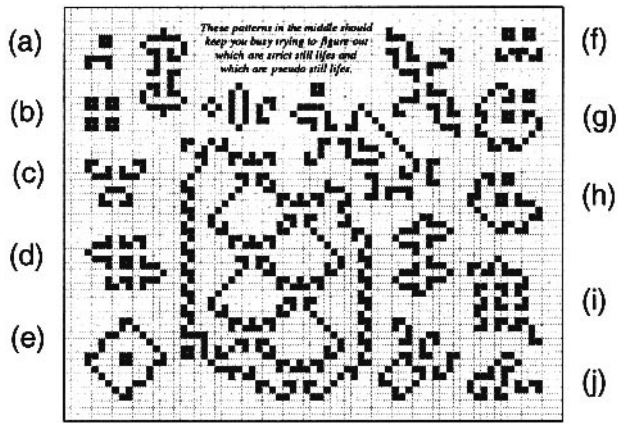

FIGURE 3 (a) Block on table: The block is stable, but the table alone is not; it is stabilized by the block. (b) Quad block: If you remove one block, the other three become unstable. But removing any two preserves stability. (c) Two bookends on coil: The coil only needs one bookend to be stable, but each bookend needs the coil. (d) Two snakes, two hats, and a block: Each item is stable alone but, if two object types are present, the rest must be there too. (e) Enclosed block: One island can be completely surrounded by another. (f) Switch: The lower island is stabilized if at least one block is present. (g) Gossamer: If one of the blocks is removed, the other must be removed as well. Compare to the quad block. (h) Sleigh: Each island alone is stable, but any two require the third. The middle three columns may be repeated to load more blocks. (i) Fragile four: Each island alone is stable, but any two require all four. (j) Dog with snake: An overcrowded cell touching two stable islands might be overcrowded by one island alone, as under the dog's chin.

between strict and pseudo still lifes is intended to capture the separability of objects that is usually easy to detect by eye.

After this distinction started being used, in the early 1970s, it was discovered that things were not as simple as they had appeared. One might have a pattern such as the "block on table" which consists of two islands, only one of which is stable alone. Is that one object or two? Or one might have a pattern such as the "switch" which consists of three islands, of which two are stable and one is stable only if at least one of the other two is present. How many objects is this? What are the objects?

As one looks at more and more patterns (see figure 3[3]), it becomes clear that a very precise definition is necessary.

Eventually, in the late 1980s, the following became accepted as the standard definition:

Definition 1 (Pseudo Still Life). *A pseudo still life is any stable pattern whose islands can be partitioned into exactly two nonempty sets, each of which is stable on its own.*[4]

This definition is mathematically unambiguous, which was a significant improvement over previous definitions. But from a computational point of view, it might be of some concern that the natural algorithm for deciding whether a pattern is a pseudo still life is exponential in the number of islands (which in turn can be linear in the amount of input, i.e., the area of the pattern), since it appears that all possible partitions into two sets must be tried. In practice so far the definition has not been used in situations with large numbers of islands, and people have accepted the exponential complexity.

In section 2 we will show that the problem in fact does not have exponential complexity as previously thought. We give an algorithm that solves it in $O(n^2)$ time.

In section 3 we will propose a natural modification of the definition to allow partitioning patterns into any number of sets of islands, rather than just two. We will show that this is equivalent to the definition allowing at most four sets of islands, due to an application of the four-color theorem.

In section 4 we will show that an intermediate definition, allowing at most three sets of islands, is NP-complete. This shows that subtle differences in the problem definition can have drastic consequences on the complexity.

In section 5 we will show that the problem of testing patterns according to the modified definition proposed in section 3 is, in fact, NP-complete as well. This is the most complicated proof, but it is greatly simplified by using ideas from the earlier proofs.

2 FAST STRICTNESS TESTING FOR STILL LIFES

Here we will show that the problem of determining whether a pattern is a strict or pseudo still life takes only $O(n^2)$ time, by giving a decision algorithm which for the most part takes only linear time, but can take quadratic time in the number of "switch"-like instances (see figure 4).

[3]The "enclosed block" was designed (9/13/98) by Noam Elkies as the smallest example of one island enclosing another, after H. König. The "fragile four" is based on a pattern designed (9/27/98) by Gabriel Nivasch as one of the smallest known of this kind, after the author. The "switch" was originally found in the early 1970s, either by Peter Raynham or by David Buckingham.

[4]This definition was first proposed by Mark D. Niemiec.

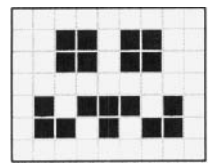

FIGURE 4 Switch.

Note that if we like, we may verify in $O(n)$ time that a pattern is stable by checking that each live cell has two or three live neighbors, and no empty cell has three live neighbors.

The algorithm has two main parts: First, we convert the pattern into a connectivity graph of switches, thereby converting the problem into one we will call *Switch Cycle*. Then, we give a simple algorithm for solving the *Switch Cycle* problem in quadratic time.

2.1 INFRASTRUCTURE TRANSFORMS THE PROBLEM

Given a pattern, we want to find whether we can decompose its islands into two stable sets. This is equivalent to finding a boundary that separates the two sets from each other. If we think of the islands as consisting of "land" cells, and we think of empty cells as "water" cells, then this boundary must go through the water, separating the islands in a stable way.

Since stability is a local property, possibilities for the boundary are determined locally throughout the pattern. All we have to do is figure out whether there exists a global boundary that divides the islands and is everywhere consistent with the local possibilities.

There are essentially three different kinds of restrictions that can occur locally. The first is that it may simply not be possible to have a boundary go through a certain water cell—any such boundary would lead to instability on one side or the other. On such cells, we will build *dams* (see figures 5 & 6) so, if we think of the boundary path as a water route, then the boundary may not cross a dammed cell any more than water traffic can cross a dam.[5]

The next kind of infrastructure we will need is *aquaducts* (see figure 7). Crossing aquaducts are placed on a cell to prevent the boundary from being able to make a turn there. The boundary must either go straight east-west, using one of the aquaducts, or go straight north-south, using the other one, but turning is not allowed. A cell containing crossing aquaducts is like two different cells: One that can be used by a north-south boundary, and one that can be used by an

[5]Of course, in real life some kinds of water traffic can cross dams. For an example of water traffic crossing a dam, see: `http://www.ils.nwu.edu/~eric/matt.html` so the analogy must not be taken too literally.

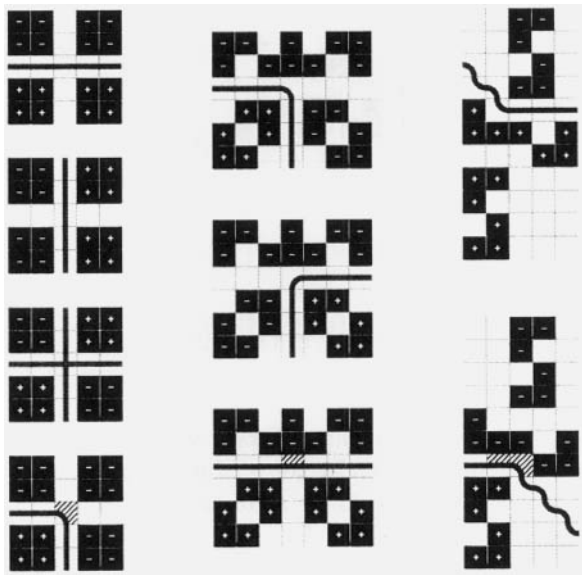

FIGURE 5 The boundaries in the upper pictures divide the patterns into two parts, shown with − and +, in a locally stable way. The boundaries in the three bottom pictures, however, would result in instabilities as shown, so they must be prohibited. Note how the two crossing boundaries in the third picture on the left result in diagonally opposite blocks belonging to the same group, since each crossing of the boundary represents switching from one group to the other.

If both aquaducts of an aquaduct crossing are used by the boundary, as might happen in the first arrangement shown in figure 8, then opposite corners wind up being in the same set, since crossing the boundary corresponds to switching from one set to the other, so crossing the boundary twice results in the original set once again. This corresponds to an "even-odd fill" (or "winding number parity") of the boundary being used to determine to which set an island should belong.

The last kind of infrastucture needed is *locks*. A lock consists of a pair of gates, each of which may be either open or closed, except that they may not both be open. If one gate of a lock is open, then the other must be closed. The boundary path may only pass through an open gate of a lock. It is easy to see that a lock is exactly what is needed to make sure that any path through the following arrangement will result in a stable boundary. This is the only arrangement requiring a lock (see figure 9).

A lock is in some sense the opposite of crossing aquaducts, since locks only allow boundaries that turn, while aquaducts only allow boundaries that go straight.

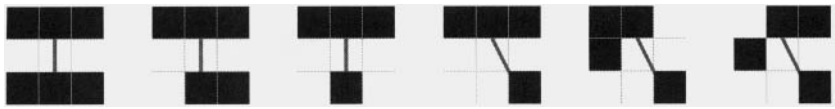

FIGURE 6 The six local arrangements where a dam is required to prevent the boundary from passing through the center cell, since any such boundary would result in instability. These are the configurations where a cell touches two stretches of land, one at three points.

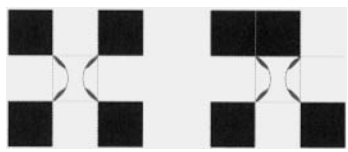

FIGURE 7 The two local arrangements requiring aquaduct crossings, shown as an east-west aquaduct passing under a north-south aquaduct. In the second arrangement, the north-south route ends immediately and so cannot be used by a boundary, and the net effect is the same as if a dam were to be placed between the two southern land cells.

see that a lock is exactly what is needed to make sure that any path through the following arrangement will result in a stable boundary. This is the only arrangement requiring a lock (see figure 9).

A lock is in some sense the opposite of crossing aquaducts, since locks only allow boundaries that turn, while aquaducts only allow boundaries that go straight.

These three types of infrastructure correspond to the three types of local restrictions that can exist for cells on the route of the boundary path. Except where infrastructure is required, as explained above, there is no restriction on the water path of a boundary. Any boundary loop that respects the infrastructure and divides some islands from others will divide the islands into two stable sets, and if there is a division of islands into two stable sets, then we can draw a boundary loop around one of them (or around an isolated part of one of them).

Once we have installed all the infrastructure, the problem takes on a much simpler form if we consider the *seas* that result as shown in figure 10. A sea is a bunch of water cells that are connected to each other after the infrastructure is installed. For the purpose of determining the extents of the seas, all the lock gates must be closed.

If any sea is not simply connected (that is, there is land both inside and outside it), then we can draw a boundary loop dividing land through that sea alone, and so the pattern is a pseudo still life. As we determine the extent of a sea, we can simultaneously check whether it is simply connected, all in $O(s)$

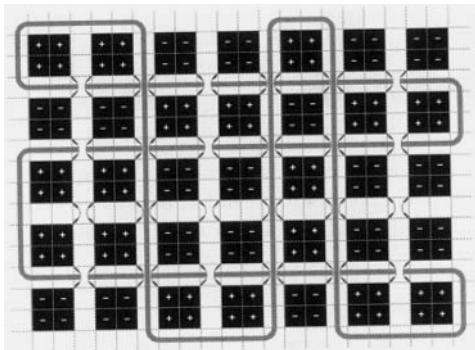

FIGURE 8 A self-crossing boundary path can be used to divide a pattern into two stable sets, if an "even-odd fill" is used to determine set membership.

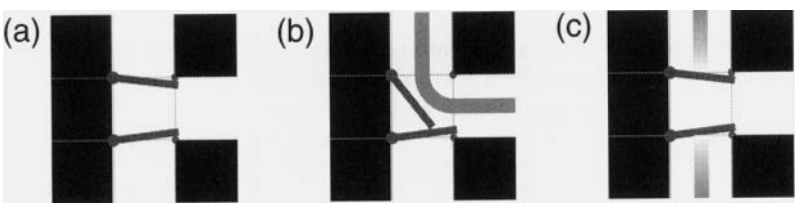

FIGURE 9 (a) A lock with both gates closed. (b) A lock with one gate open, allowing the boundary to pass through. (c) The boundary cannot cross a lock like this since the gates cannot both be open.

time, where s is the size of the sea. We merely need to "grow" the sea one cell at a time, keeping a list of neighboring sea cells that need to be added. (Cells with crossing aquaducts should be treated as two separate cells, one for each crossing direction.)

To check for simply connectedness, when adding a cell, we first check whether more than one of its four neighbors has already been added to the sea, and if so, we check whether they are connected by cells already added to the sea among just the eight neighbors of the cell being added. If not, then adding this cell could, for the moment, result in not being simply connected, so instead of adding the cell to the sea now, we move it to a list of "postponed" cells. Any time a cell is added to the sea, if any of its eight neighbors are marked as "postponed," they are moved back to the regular list of cells waiting to be added, so they can be tried again. Once we have finished processing the regular list, then if any cells remain in the "postponed" list, the sea is not simply connected.

The "outside sea" requires special treatment during the check for simply connectedness: Any water cell at the edge of the pattern should be considered

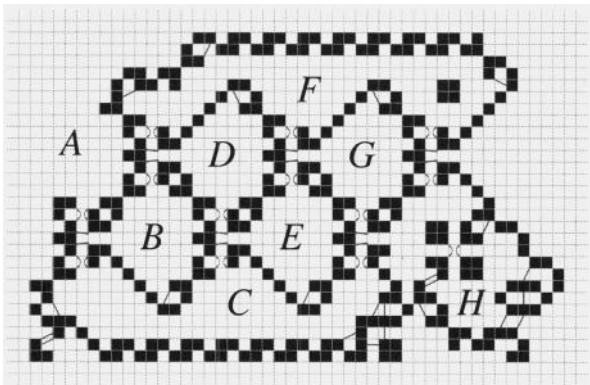

FIGURE 10 With all lock gates closed, each connected body of water is a sea. If any sea is not simply connected, then a boundary can be drawn, within that sea alone, as in sea F, which encloses a block, or H, which has a "figure 8" loop using the aquaduct. Otherwise, lock gates need to be judiciously opened to allow a boundary if possible. Note that there are many very small seas of just one or two cells which are not marked with a letter in this diagram. These seas cannot possibly be used by a boundary, since they are simply connected and do not touch any locks.

as connected to all other water cells at the edge of the pattern. This "outside sea" can be thought of as wrapping all the way around the planet. When treated this way, the above algorithm will work correctly for it, too.

When growing the seas, we must keep track for each sea of what locks it touches, and whether each sea touches them in the middle or at a side gate. Then, assuming all of the seas are simply connected, we need to determine whether it is possible, by opening lock gates, to allow a boundary loop to be drawn.

We will think of the seas and locks as a combinatorial graph [6], with special vertices for the locks (we will call them "switch" vertices), and edges connecting the locks through the seas, representing possible routes for a boundary. If a sea touches more than two locks, we may use any tree of edges and vertices to represent the sea.

Figure 11 shows how such a graph can easily be turned into one in which *all* vertices are switch vertices, in $O(n)$ time, without affecting the question of whether there is a cycle or not. The question of whether such a graph contains a cycle is a problem that we will call *Switch Cycle*.

Everything up to now has only taken $O(n)$ time, but *Switch Cycle* will take a little longer. Fortunately, at this stage of processing, almost all small to medium-size patterns have already been classified, and only very large patterns full of switches will present nontrivial questions for *Switch Cycle*.

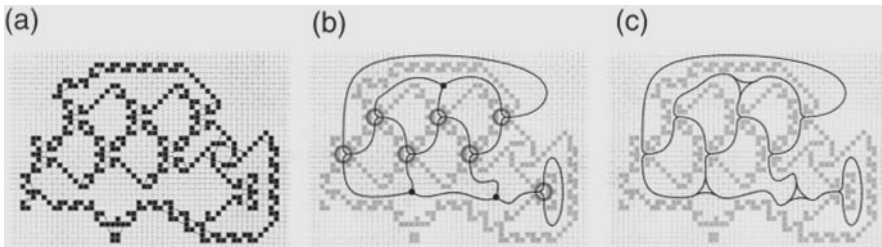

FIGURE 11 (a) Here is a pattern in which all the seas are simply connected. We want to know if lock gates can be opened so as to permit a boundary that will divide the pattern into two stable sets of islands. (b) In each sea, we draw lines connecting up the lock parts it touches. When going through a lock, the boundary must follow one of the smooth curves rather than passing straight across. Like a train on a track, it cannot have a cusp in its route. (c) We can convert the graph to a form which only uses "train track branch" vertices, by first drawing each sea as a trivalent tree connecting its locks as in the previous diagram, and then replacing each tree vertex with a traiangular branching arrangement as shown here.

2.2 AN $O(N^2)$ ALGORITHM FOR SWITCH CYCLE

A *switch graph* is like an ordinary combinatorial graph, but some of its vertices may be "switches," which are vertices of degree three (touching three edges) which "prefer" one of their edges. A switch vertex acts like a switch, in that it can connect its preferred edge to exactly one of the other two edges, leaving the third edge disconnected. A switch graph specifies the preferred edge for each switch vertex, but does not specify the settings of the switches, i.e., it does not specify to which other edge each preferred edge should be connected (see figure 12).

Given a switch graph, we are interested in the following question: Can the switches be set so that the resulting connections contain a cycle? In other words, is there a cycle such that for each switch vertex in the cycle, the preferred edge of that vertex is also in the cycle? We will call such a cycle a "switch cycle," and we will call the problem of determining whether such a cycle exists the *Switch-Cycle* problem.

We will prove a theorem that will aid us in answering this question: Consider a switch graph in which every switch is replaced by an ordinary vertex of degree three. We will call this the *normalization* of the switch graph. Then, the theorem is the following:

Theorem 2.1. *If the normalization of a switch graph contains no bridges (no cut edges), then the switch graph contains a switch cycle.*

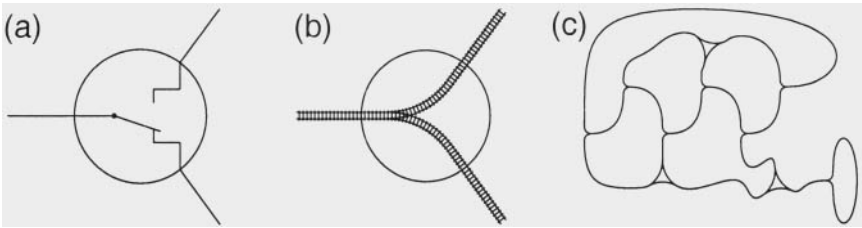

FIGURE 12 (a) A switch vertex, enlarged here to show hypothetical detail, can be thought of like a mechanical switch that connects its "preferred edge," shown here coming in on the left, to exactly one of the other two edges. (b) Another way to think of a switch is like a branching train track. Just as a train must use the track on the left, and may not go from the upper right track to the lower right track, a switch cycle's route must obey the same restriction. (c) The train track version is especially easy to draw and read, so we will use this approach when drawing a switch graph. At each branching, a cycle must pass in a smooth, rather than cuspy, manner. So the question is, is there a smooth loop?

We will accept the theorem without proof for now, postponing its proof to the end of this section.

A bridge (also known as a cut edge) is an edge of a graph that is "the last straw" connecting two parts of the graph, meaning that if the edge were to be removed, the graph would become disconnected. Clearly, such an edge cannot be part of a cycle, since there would be no way for the rest of the cycle to connect the two ends of the edge. So if the graph has any bridges, they may be removed without affecting the question of whether there is a cycle in the graph, or a switch cycle in the switch graph.

What the above theorem tells us, then, is that if there are no more bridges to remove, then the graph must contain a switch cycle! So there is a very simple algorithm for *Switch Cycle*: As long as there are bridges, remove them. If anything is left at the end, then there is a switch cycle.

When we remove an edge in a switch graph, we have to look at how it might have been connected to switches (for example, see figure 13). If it was the preferred edge of a switch, then without it, the other two edges of that switch cannot connect to anything, since they could only have connected to the preferred edge. So in this case, the other two edges should be removed as well. On the other hand, if it went to a switch but was not a preferred edge, then once it is gone, there is no reason for the switch not to connect the remaining two edges, so in this case, the other two edges should be merged into one long edge, removing the switch. (If the other two edges were, in fact, two ends of the same edge, then a cycle exists using that edge alone, and we are done.) So in either case, the switches at the ends of the original edge are removed, and possibly

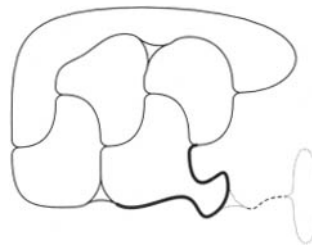

FIGURE 13 This graph contains a bridge, shown dashed, that we would like to remove. After removing it, its three dependent edges, shown in gray, must be removed as well. Then three of the remaining edges, shown in bold, must be merged into one long edge. The resulting graph does not have any more bridges, so our theorem tells us that it must contain a switch cycle.

other edges are removed as well ("dependent edges"), leading in turn to more switches and edges being removed. All in all, the total amount of time required is proportional to the number of edges that are removed.

Identifying the bridges in a graph is a well-known problem [5] that takes only $O(n)$ time using a simple depth-first search strategy. So repeatedly identifying the bridges and removing them (along with any dependents) can only take at most $O(n^2)$ time.

All that remains is to prove our useful theorem.

We will just consider graphs in which all vertices are switch vertices, since, as we saw at the end of section 2.1, we can convert any switch graph to such a form in $O(n)$ time without affecting whether the graph has bridges or has a switch cycle.

First of all, note that in a graph with no bridges, it cannot be the case that both ends of an edge meet at the same switch, since then the third edge going to that switch would be a bridge.

Now, in a graph with no bridges, we will say that a *free* edge is any edge which is not the preferred edge for either of the switches it touches. A free edge is called this because it does not have any dependents, and can be removed without forcing any other edges to be removed. (Its neighbor edges would merely be merged.)

Note that every switch graph must have a free edge, as a simple pigeonhole counting argument shows: If we look at the ends of all the edges, noting which are and are not preferred, we will of course find that only one third of the edge ends are preferred, since each switch prefers one of its three edge ends. Since two thirds of all edge ends are not preferred, there must be some edge for which both ends are not preferred, i.e., a free edge.

Now, suppose that the theorem is not true. Then there must be some switch graph which has no bridges, and yet does not contain a switch cycle. Let us

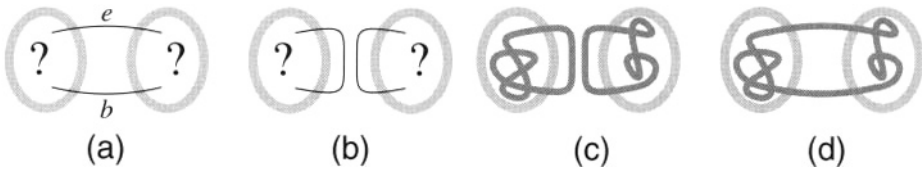

FIGURE 14 (a) We know that G can be drawn as two components connected only by e and b. (b) We can cut G up into two smaller graphs, each "short circuiting" the other component. (c) If only the smaller graphs, but not G, have switch cycles, then they must be using the short circuits. (d) But then we can reconnect the components and see that G has a switch cycle too.

consider such a graph G which is minimal, i.e., has at least as few edges as any other such graph. We will call this graph a minimal counterexample to the theorem.

Pick a free edge e. If we remove e, then one of two things must happen: Either we get a smaller switch graph, or, after merging the neighbor edge ends, we just get a loop with no switches in it at all. In the latter case, the loop is a cycle which also existed as a switch cycle in the graph before removing e, contradicting the assumption that G has no switch cycle. So it must be the case that removing e leads to a smaller switch graph, which we know must have a bridge b, since otherwise it would be a smaller counterexample.

So G must consist of two components which are connected only by e and b as shown in figure 14. If we look at just one of the components, and attach its dangling edges e and b to each other to become a single edge, then there cannot be any bridges, since they would also be bridges in G. Therefore, it must have a switch cycle, or else it would be a smaller counterexample. If the switch cycle does not use the merged e-b edge, then G contains the same switch cycle, contradicting our assumption, so the switch cycle must use the merged e-b edge.

The same must be true for the other component: It must have a switch cycle using its merged e-b edge. But this means that if we put the two components back together to form G, then we can merge the two smaller switch cycles along e and b to form one big switch cycle, again contradicting our assumption that G has none.

Since the assumption of a counterexample leads in all cases to a contradiction, there must not be any such counterexample, and so the theorem must be true.

If we want an algorithm to actually find a switch cycle if there is one, rather than just determining that there must be one, we can use the above proof as a recursive algorithm: Where the proof appealed to smaller counterexamples, the algorithm would recurse, finding and using switch cycles present in the smaller graphs to exhibit an explicit switch cycle in G, again in only $O(n^2)$ time.

3 A MORE INTUITIVE DEFINITION

The conventional definition given in section 1 for a pseudo still life requires that the pattern be decomposable into exactly two sets, each of which is stable alone. But it turns out that there are many patterns which do not fit this definition, even though they are distinctly composed of a collection of still lifes! In the last figure of section 1, we saw some instances of such patterns: The "sleigh," the "fragile four," and the "two snakes, two hats, and a block" are all examples of patterns which cannot be divided into exactly two stable groups of islands, even though they can be divided into three or four stable groups. We would naturally like to think of these patterns as being pseudo still lifes, just like the "quad block," since they are close collections of stable islands. But the conventional definition classified them as strict still lifes, since it only considered partitions into exactly two groups.

A more natural definition immediately suggests itself, namely:

Definition 2 (Natural Pseudo Still Life). *A natural pseudo still life is any stable pattern whose islands can be partitioned into two or more nonempty sets, each of which is stable on its own.*

This definition captures the idea that a stable collection of stable objects should be considered a pseudo still life, and only indivisible collections of islands should constitute strict still lifes.

One can see that the "sleigh" needs to be divided into three parts in order for each part to be stable, and the "fragile four" needs to be divided into four parts. Is there some pattern that requires five or more parts? Perhaps surprisingly, the answer is no. For suppose you can divide a pattern into five or more stable subsets. Then the following process will rearrange the islands into just four stable subsets:

First, enlarge each island by half a cell, so that islands can border each other. (Any cell with no live neighbors remains "international waters.") Call each group of contiguous islands of the same set a "country." (Where two enlarged islands touch only at a point, we do not consider them to be contiguous.) Note that each country is independently stable. Note also that if two countries meet just at a point, then they will each be stable regardless of whether their islands are in the same group or not.

Now we have a map to which the four-color theorem [1, 9] applies. The four-color theorem guarantees us that we can color the countries with just four colors so that any two bordering countries are different colors. Given such a four coloring, we can put each island into a set according to the color of its country.

This means that where two abutting enlarged islands have been in the same set (and therefore the same country), they will still be in the same set (since they will both be in the set for that country's color), and where they have been

in different sets (and thus in different countries), they will still be in different sets (since neighboring countries get different colors in the four coloring).

In places where the enlarged islands of two countries touch at just a point, they will be stable regardless of whether the countries wind up having the same color or not, since the rules for Life are such that it makes no difference to stability whether diagonally opposite islands are assigned to the same set or not.

So the four-color theorem is sufficient for showing that allowing divisions of the islands into any number of stable sets is no different from allowing divisions into at most four stable sets.

So an equivalent form of our more natural definition would be:

Definition 3 (Natural Pseudo Still Life, again). *A natural pseudo still life is any stable pattern whose islands can be partitioned into two, three, or four nonempty sets, each of which is stable on its own.*

The complexity of detecting whether patterns are natural pseudo still lifes will be left until section 5.

4 NP-COMPLETENESS OF THE "THREE SET" DEFINITION

In the last section, we proposed a definition that concerned partitionability into two, three, or four proper subsets. The standard definition concerns itself only with partitionability into exactly two proper subsets. Clearly we could also have an intermediate definition which considers partitionability into *two or three* proper subsets.

Although neither conventional nor intuitive, this intermediate definition does have an interesting property: We are able to show that the complexity of determining whether a pattern is a strict still life according to this definition is NP-complete. In other words, this determination requires[6] exponential time to compute.

We will devote the remainder of this section to this proof.

First of all, the problem is clearly in NP, since if we are given a pattern which is already marked with a proposed partitioning of the islands into two or three sets, we can easily verify in polynomial time whether the proposed partitions would be stable. So a nondeterministic computer could, in effect, try all possible partitioning schemes in parallel (or use an oracle to just try the best one), and in polynomial time it would know whether there is a stable partition.

To show that the problem is not just in NP, but NP-complete, we will show that if we could solve this problem in polynomial time, then we could also solve a known NP-complete problem, *CNF Satisfiability*, in polynomial time.

[6] Given the current (and, in most people's opinion, any future) state of the art in computational complexity theory.

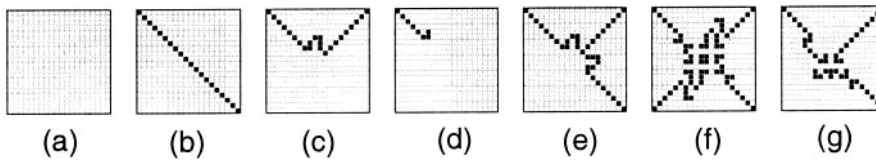

FIGURE 15 (a) Empty space; (b) Straight wire; (c) Turning wire; (d) Wire end; (e) Wire junction; (f) Wires crossing; (g) Simple switch.

CNF Satisfiability is a well-known problem which asks, given an expression in "Conjunctive Normal Form" (e.g., $(a \vee b \vee \overline{d}) \wedge (\overline{a} \vee \overline{c}) \wedge (\overline{b} \vee e \vee \overline{f} \vee \overline{g}) \wedge \cdots$, where the whole expression is a conjunction of terms, each of which is a disjunction of variables or their negations), does there exists a set of Boolean values for the variables which results in the whole expression being true?

Given a *CNF Satisfiability* problem, we will construct a pattern which is either a pseudo or strict still life, depending on whether there is a solution to the *CNF Satisfiability* problem or not.

We will use simple 20×20 building blocks to build our stable Life pattern. These building blocks will have "wires" of cells, and the "value" of a wire will be the set that it belongs to. If we like, we can think of each of the (at most) three sets of islands as having a distinct voltage level, constant along wires and where switches connect them.

In figure 15 we recognize the last diagram as being the same kind of switch connection we discussed in section 2.2. In it, the wire arriving from the lower right must be in the same set as (i.e., connected to) either the upper right or the upper left, with the other upper wire being free to belong to any set (i.e., disconnected).

The second-to-last diagram, of two wires crossing, needs some explanation. In it, the wires at opposite corners must belong to the same set, and to which set one pair belongs is independent of to which set the other pair belongs.

To see this, the first notice that if any wire belongs to the same set as the center block, then the next wire clockwise must also belong to the same set, in order to stabilize a cell diagonally adjacent to the center block. This, in turn, will force the next wire after that to belong to the same set, too, and in the end all four wires together with the center block must belong to the same set.

The other possibility is that the center block does not belong to the same set as any of the wires. In this case, we notice that two adjacent wires may not belong to the same set, since then a cell diagonally adjacent to the center block would be unstable. Since there are only three possible sets, this means that the center cell must belong to one set, and the wires must then alternate between the two other sets, so that opposite wires belong to the same set.

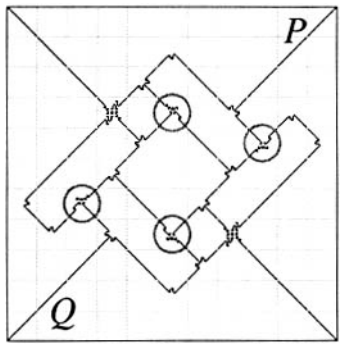

FIGURE 16 A symmetric arrangement. The simple switches are circled. For each of the four wires coming from the corners, there is a switch that forces it to be in the same group as at least one of the two corner wires perpendicular to it. For example, the simple switch on the right says that at least one of P or Q must be in the same group as the lower right wire.

So in summary we see that no matter what, opposite wires must belong to the same set. Furthermore, if one pair of opposite wires belongs to set i, and the other pair belongs to set j, then we can have either $i = j$ (in which case the center block belongs to the same set, too), or $i \neq j$ (in which case the center block belongs to the third set, not i or j). So we see that the diagram does indeed correspond to two independent crossing wires.

What's great about these components is that we can use them to build up a switch graph. In section 2 the switch graph represented possible routes for a dividing boundary, but here the switch graph directly represents the islands of the pattern. Now, if a switch connects two wires of the pattern, it means that those wires must be assigned to the same stable partition. So for the pattern to be a pseudo still life, we must be able to set the switches so that the pattern becomes disconnected, consisting of disjoint pieces not connected by any switch settings. The question of whether a switch graph can have its switches set so that it becomes disconnected is a problem that we will call *Switch Disconnected.*

The first thing we will build out of our 20×20 building blocks is the following symmetrical arrangement of four simple switches (see figure 16):

Say the upper right wire is in set P and the lower left wire is in set Q.

If P and Q are the same set, then the left simple switch forces the upper left wire to be in this set, too, and the right simple switch forces the lower right wire to also be in this set, so all four wires must be in the same set.

Now suppose P is different from Q. Consider the upper left and lower right wires. The upper simple switch says that one of them must be P, while the lower simple switch says that one of them must be Q. So they must be different, one

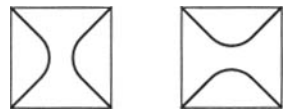

FIGURE 17 The two possible effective connectivities for a Fancy Switch.

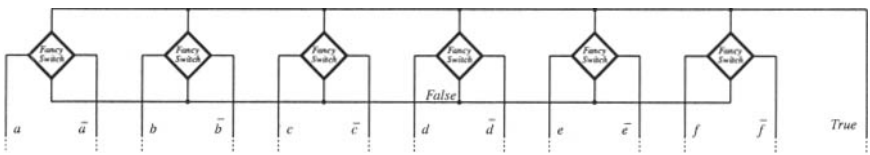

FIGURE 18 This diagram is rotated 45 degrees from the previous orientation.

being P and one being Q. But which is which? One can easily check that the simple switches allow either possibility.

So this is almost the opposite of the "wires crossing" block, in that this forms *non*-crossing connections. This is a essentially a fancy kind of switch, that forces the four wires to have at least one of the two possible connectivities shown in figure 17.

Now, with this Fancy Switch, we are ready to start constructing a pattern for the *CNF Satisfiability* problem.

We will start by having a row of Fancy Switches (see figure 18), one for each variable. A bunch of wires will dangle down, where we will add things onto them.

We will name the upper wire "*True*" and the horizontal wire underneath all the Fancy Switches "*False.*" *True* and *False* might wind up in the same set, or they might wind up in different sets, when we partition the pattern into stable sets.

The two wires dropping out from the sides of each Fancy Switch will be called x and $\overline{x}$ for the Fancy Switch corresponding to variable x.

If *True* and *False* belong to the same set, then all the x and $\overline{x}$ wires will also belong to this set. If *True* and *False* belong to different sets, then the Fancy Switch for x either puts x in *True*'s set and $\overline{x}$ in *False*'s set, or else it puts x in *False*'s set and $\overline{x}$ in *True*'s set.

Now we need to represent the terms of the big conjunctive expression.

This turns out to be very easy: We can use simple switches to build up each term. As an example, we'll look at the term $(a \vee b \vee \overline{d} \vee e)$ (see figure 19).

The rightmost simple switch says that either e must be in *True*'s set, or else... the next switch says that either $\overline{d}$ must be in *True*'s set or else... either b must be in *True*'s set or a must be in *True*'s set.

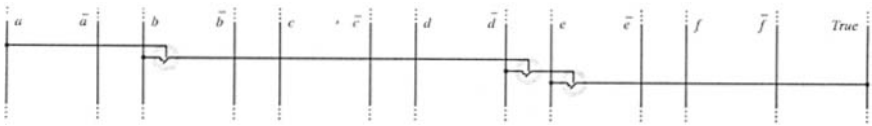

FIGURE 19

In other words, supposing *True* and *False* are in different sets, we can think of each Boolean variable x as being true iff its wire is in the same set as *True* (which is the same as the $\overline{x}$ wire being in the same set as *False*). So the previous paragraph is just saying that $(a \vee b \vee \overline{d} \vee e)$ must be true, and this is exactly the term we wanted to represent.

We continue down with all the other terms of the big conjunctive expression, representing them in this way, one after the other, until at the end we just terminate all the dangling wires, at which point we are done making the pattern.

If we continue supposing that *True* and *False* are in different sets, then we see that a stable partition corresponds perfectly to a solution of the *CNF Satisfiability* problem.

If *True* and *False* are in the same set, then one can quickly verify that all the wires in the diagram, and therefore all the islands in the diagram, must belong to this same set. (Recall that the only islands that are not wires are the blocks at the middle of wire crossings.) So there is no way to stably partition the islands into two or three proper subsets if *True* and *False* are in the same set.

So, for the definition proposed at the beginning of this section, the pattern we have constructed is a pseudo still life exactly when there is a solution to the *CNF Satisfiability* problem, and it is a strict still life exactly when the *CNF Satisfiability* problem has no solution.

And, as a bonus, our method of proving this has also shown that *Switch Disconnected* is an NP-complete problem.

5 NP-COMPLETENESS OF THE NATURAL DEFINITION

In this section, we will show that the problem of determining whether a pattern is a pseudo still life according to the natural definition proposed in section 3 is an NP-complete problem.

We will use techniques similar to those of section 2 to convert the problem into a switch graph problem, that of detecting whether a specific layout of a switch graph in the plane can contain a simple loop that does not cross itself. We will call this problem *Switch-Simple Loop*.

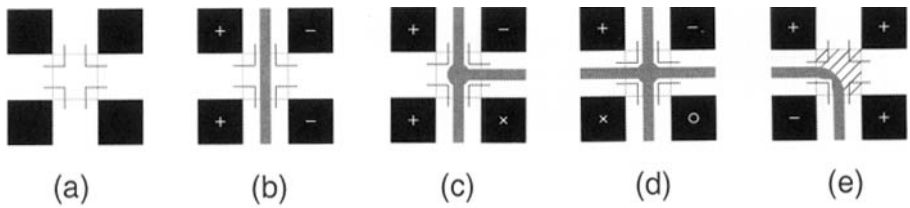

FIGURE 20 (a) This is how we will draw a bend preventer. They will go where aquaducts went. (b) A bend preventer allows a boundary to go straight through it. (c) It also allows a T junction joining three boundaries. (d) Four boundaries can also meet at a bend preventer. (e) Bend preventers do not allow a boundary to bend, creating an unstable set.

Then, using ideas similar to those in section 4, and based on ideas of Mazzoni and Watkins,[7] we will show that *Switch-Simple Loop* is NP-complete.

5.1 BOUNDARIES BETWEEN STABLE GROUPS

We are faced with the question of whether a given stable pattern can have its islands partitioned into stable sets. As in section 2, we will look at the boundaries between islands that are in different sets, and see how these boundaries can be connected. Since there can be as many sets as we want, rather than just two, we will be searching not just for a cycle, but for a more general network of boundaries, dividing the plane into countries like the ones discussed in section 3. Since countries do not cross each other, our network of boundaries will not need to have any boundaries crossing any other boundaries, and so instead of the crossing aquaducts of section 2, in their place we will have a special kind of infrastructure that allows three or four boundaries to meet, or one to pass through, but will not let a single boundary make a turn. We will call this piece of infrastructure a "bend preventer"; see figure 20.

As in section 2, we can determine the seas, making a graph in which bend preventers and locks are connected to one another via connecting seas. Since we no longer have crossing aquaducts, the graph will be planar.

Our task will be to see whether our graph can contain a network of boundaries subject to the simple constraint that the network must not contain any bridges (cut edges). The reason bridges are prohibited from the network of

[7] Dominic Mazzoni and Kevin Watkins wrote a proof that the problem of deciding whether a position in the game of Twixt is a winning position or not is NP-complete, currently available at: `http://www.mathematik.uni-bielefeld.de/~sillke/PROBLEMS/Twixt_Proof_Draft`. Their proof effectively showed that the problem of determining whether there is a simple non-crossing path between two given points in a planar layout of a graph is NP-complete. Their proof very directly inspired the proof in section 5 of the NP-completeness of *Switch-Simple Loop*.

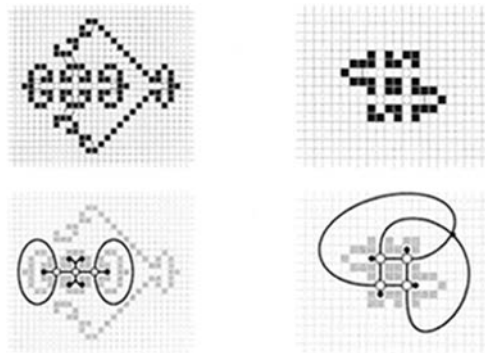

FIGURE 21 (Left) Here is a pattern with some bend preventers, and a graph of sea connections between the bend preventers. All of the edges except for the circular loops are bridge edges, and so they cannot be used by a network of boundaries, since they would be bridges in the boundary network as well, and the land on both sides of a bridge must belong to the same set. (Right) The pattern of two snakes, two hats, and a block is just four bend preventers arranged like this. If any of the sides of the square are used by a boundary, then the next bend preventer going counter-clockwise must contain a T junction of boundaries. Continuing around the square, we see that all sides and departing edges must be boundaries. Therefore, if any of the emanating edges is a boundary, then every one of them must be a boundary.

boundaries is because both sides of the bridge would correspond to the same country, which means there should not be a boundary there (see figure 21).

Figure 22 shows a wonderful configuration which permits boundaries to come out of it either along one set of diagonally opposite edges, or along the other set of diagonally opposite edges, but not in any other way. If we think of this as two paths crossing, we see that at most one of the two paths may be used. We will call this a "blocking crossing," since, if it is crossed in one direction, that effectively blocks its use in the other direction.

Given any planar graph containing only blocking crossings and switches, it should be clear that we could construct a big stable Life pattern corresponding to it. Such graphs are like switch graphs, except that they have a specific layout in the plane, and where edges cross each other, they do it with blocking crossings. We will call this kind of graph a "switch graph layout"; see figure 23.

We will call the problem of finding a loop in a switch graph layout *Switch-Simple Loop*. In section 5.2, we will show that *Switch-Simple Loop* is NP-complete by designing a big switch graph layout corresponding to a given *CNF Satisfiability* problem so that, if there is a solution to the *CNF Satisfiability* problem, then there is a loop in the switch graph layout, but, if there is no solution to the *CNF Satisfiability* problem, then there is no loop and no stable network of boundaries in the switch graph layout.

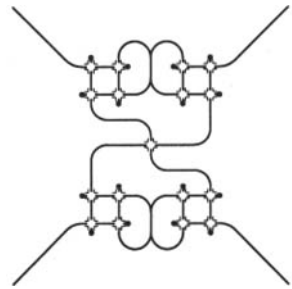

FIGURE 22 Each of the four edges entering this arrangement can be a boundary if all four edges emanating from the square it leads to are boundaries. The two upper squares are connected by a pair of switches that ensures that only one of the squares may be used by boundaries. The same is true for the two lower squares, so the central bend preventer cannot have more than two boundaries coming to it, and so it will only allow boundaries to go straight across it. This means that of the four edges entering the arrangement, only two opposite edges may have a boundary. We will call this arrangement a "blocking crossing."

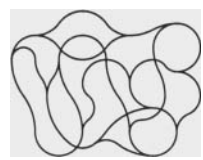

FIGURE 23 Here is an example of a switch graph layout. The question we are interested in, given such a graph, is whether it contains a loop that does not cross itself. In the next section, we will show that this is an NP-complete problem.

As it turns out, if there is a boundary loop in the switch graph layout, then the islands will be partitionable into three stable sets, and a fourth is not needed. The two main sets are the islands inside and outside the loop, and a third set is sufficient for keeping blocking crossings stable where they are traversed by the boundary loop. This fact will not affect the proof, but it means that regarding still lifes, this proof is stronger than the one in section 4.

5.2 SWITCH-SIMPLE LOOP IS NP-COMPLETE

The basic approach of this section is inspired by a proof of Mazzoni and Watkins (see footnote 7).

We will show that given a *CNF Satisfiability* problem, we can make a *Switch-Simple Loop* problem which has a solution exactly when there is a solution to the *CNF Satisfiability* problem.

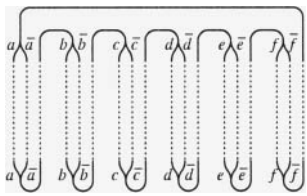

FIGURE 24 We will construct a Switch-Simple Loop problem corresponding to the CNF Satisfiability problem by creating a switch graph layout whose overall form will be as shown here. The resulting Switch-Simple Loop problem will only be able to have a loop if the loop follows this overall path, effectively picking a value for each variable. We will modify the central dotted portion so as to represent the terms of the CNF expression.

Our *Switch-Simple Loop* problem will take the following overall form, in which choosing a value for a variable is represented by choosing one of two vertical paths stretching from the top to the bottom of figure 24.

In order to represent the terms of the CNF expression, we will need to be able to "push" the values of variables horizontally out to the right, where we can make them interact according to the CNF terms. We can do this by making the vertical paths protrude slightly out to the right so as to interfere with neighboring columns. A pair of protrusions will propogate a variable's value: The upper protrusion will be part of the loop if the variable is true, while the lower protrusion will be part of the loop if the variable is false.

Once the values of the variables for a CNF term have been propogated all the way to the right side of figure 25, we can implement the CNF term by ending those protrusions corresponding to the values appearing in the CNF term, leaving the negations of those values to play a game of musical chairs, where if there are n such negations, then they must share $n - 1$ positions. This is impossible if all of the negations are part of the loop, but, if any of them are not (that is, if any of the values in the CNF term are true), then the rest can "lean toward" the unused negation, and there will be room for all of them to coexist.

In this way, we can represent each CNF term, one after the next, down through the switch graph layout (see figure 26). When we are done, we have a switch graph layout with the property that if any part of it is used as a boundary, then the boundary must follow the overall form that we intended, effectively choosing values for the variables that solve the *CNF Satisfiability* problem.

Therefore the *Switch-Simple Loop* problem has a solution exactly when the *CNF Satisfiability* problem has a solution, and these questions are the same as the problem of determining whether the pattern corresponding to the switch graph layout is a natural pseudo or strict still life.

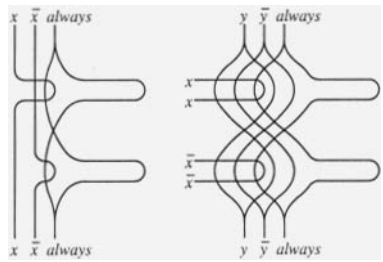

FIGURE 25 (Left) This shows how we can alter the three vertical paths corresponding to a variable in the previous diagram so that the variable's value is pushed out to the right. The third vertical path, which is always used by the loop, must use one of two branches here. If the variable is true, then the loop must use the branch with the upper right protrusion, since the other branch crosses the first path, which is used by the loop when the variable is true. Similarly, if the variable is false, the loop must use the lower protrusion on the right. (Right) This shows how we can keep pushing a variable's value to the right. If the upper protrusion coming from the left is used by the loop, then the vertical paths used here will have to swing out to the right to avoid it, and swing left across the unused protrusion. Since the third vertical path is always used, the upper right protrusion must be used. Similarly, if the lower protrusion from the left is used, then the lower protrusion to the right must be used.

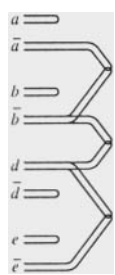

FIGURE 26 This shows how we can implement a disjunction of the CNF expression, such as "a or b or not d or e," after pushing all the relevant variables out to the right. If one of the terms, say b, is true, then only the stubby path will be used there, and so, if any of a or not d or e is false, they may use the long path that leans toward b, and they will not cross each other. But if none of the terms is true, then there will not be enough room for all the falsehoods to be able to coexist without crossing each other.

Since both *Switch-Simple Loop* and the natural pseudo still life question are problems for which a positive solution can be easily verified, we have proved that these problems are NP-complete.

6 CONCLUSION

What started out as a practical problem, creating an efficient "object analyzer" for stable regions in the Game of Life, turned out to be quite a complex problem, giving rise to new questions, new methods, and, in general, a new area of investigation: Still Life Theory.

The various *Switch-Graph* problems are also interesting in their own right, and one can easily come up with new and interesting problems in this area.

We found that testing patterns under the conventional definition for strict still lifes turns out to have quadratic complexity rather than exponential complexity as has been assumed by workers in the field. We proposed a more intuitive definition, but unfortunately it turned out to make testing patterns an NP-complete problem, thus making it unappealing for very large patterns. Since the definition is trying to capture a distinction that was originally thought to be obvious, one hopes that, if not quite obvious, the distinction should at least be a tractable problem! An intermediate definition also resulted in pattern testing being NP-complete, so we see that the conventional definition is in fact uniquely amenable to use for testing large complicated patterns.

In a broader perspective, these results show that, in general, questions about decomposability of stable patterns for a cellular automaton can vary widely in complexity, and are likely to be very sensitive to the particular details of what kind of decomposition is required. And, as we have seen, the solutions to such questions can be very intriguing.

As a final remark, let us note that stable Life patterns are just one instance of a kind of two-dimensional language called *local lattice languages* [7], which are characterized by having a finite set of locally allowable configurations. Conversely, for any local lattice language, we can easily construct a cellular automaton whose stable patterns are exactly the members of the local lattice language, so we see that speaking about the still lifes of a cellular automaton is equivalent to speaking about a local lattice language.

REFERENCES

[1] Appel, K., W. Haken, and J. Koch. "Every Planar Map is Four Colorable." *Illinois J. Math* **21** (1977): 429–567.

[2] Conway, John. *Winning Ways For Your Mathematical Plays*, edited by E. Berlekamp, J. Conway, and R. Guy, Vol. 2. New York: Academic Press, 1982.

[3] Flammenkamp, Achim. Statistics about the objects produced from a random initial state can be found at:
⟨`http://www.uni-bielefeld.de/cgi-bin/php/~achim/gol.html`⟩.

[4] Koenig, H. Lists of possible small still lifes with constructions for many, as well as their frequencies of occurrence from a random initial state, can be found at:
⟨`http:/www.pentadecathlon.com/LifeInfo/LifeInfo.html`⟩.

[5] Leiserson, C., and Rivest, eds. *Introduction to Algorithms*, Ch. 23, problem 23–2.

[6] Li, C. L., ed. *Introduction to Combinatorial Mathematics.*

[7] Lindgren, K., C. Moore, and M. Nordahl. "Complexity of Two-Dimensional Patterns." Working Paper 97-03-023, Santa Fe Institute, Santa Fe, New Mexico, 1997.

[8] Niemiec, Mark D. Lists of possible small still lifes as well as methods for creating many of them from gliders can be found at:
⟨`http:/home.interserv.com/~mniemiec/lifepage.htm`⟩.

[9] Robertson, N., D. P. Sanders, P. D. Seymour, and R. Thomas. "A New Proof of the Four-Color Theorem." *Electron. Res. Announc. Amer. Math. Soc.* **2** (1996): 17–25.

[10] Silver, Stephen. Currently maintains the Life Lexicon, which lists most of the Life terms you're likely to see, at:
⟨`http:/www.cs.jhu.edu/~callahan/lexiconf.htm`⟩.

Many of the items here refer to web pages, which unlike items published on paper and stored in libraries, can easily change or disappear without notice. To help ameliorate this, I will maintain a page of links to the web pages cited above at:
⟨`http:/www.paradise.caltech.edu/~cook/Warehouse/StillLifeLinks.html`⟩.

Replicators and Larger-than-Life Examples

Kellie Michele Evans

After watching a substantial number of cellular automaton dynamics generated by rules containing suitable ingredients, eventually a particular time-dependent pattern catches the eye. A configuration of occupied sites makes copies of itself, then the copies make copies of themselves, and these copies move toward one another and also toward the boundaries of the evolution. This continues as long as there is room for the evolution. When the innermost copies collide, they annihilate one another. Meanwhile, the outermost copies continue to reproduce, provided that no occupied sites from the outside impede. The pattern repeats, *ad infinitum*.

We first saw this kind of evolution, which we call a *replicator*, in our studies of the *Larger-than-Life (LtL)* family of cellular automaton (CA) rules. The first replicators we found were all in the same region of *LtL space*. We thought an intrinsic property of this specific region was necessary for the existence of a replicator. However, we began seeing similar configurations, with slight variations, in many different subregions of LtL space. Then we learned of the range 1 *HighLife* replicator (which we call *bow tie pasta*), a very intriguing example that has since

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

become quite famous. We saw more examples on Christopher Langton's computer at the Santa Fe Institute in 1995; this convinced us that the behavior was not exclusive to LtL-like rules. Since then, new replicators have been discovered for a variety of CA rules.

In this chapter, we define a replicator using an axiomatic approach and prove various theorems that follow from the axioms. We also present a collection of Larger-than-Life replicator examples, HighLife's famous example, and propositions that generalize several of the LtL examples.

We will begin by presenting a collection of Larger-than-Life replicator examples, but first let us define the family of Larger-than-Life update rules.

1 DEFINITION OF LARGER-THAN-LIFE

Larger-than-Life (LtL) is a four-parameter family of *two-state* cellular automaton rules. The four parameters are the upper and lower bounds of the *birth* and *survival* intervals. At each time t, each site $x \in \mathbb{Z}^d$ is either *live* or *dead*. We think of a live site as being in state 1 and a dead site as being in state 0. At each time step, each site updates (meaning it switches state or not) according to the number of 1s in its neighborhood. Let us define the rule precisely.

- Let $\mathcal{N}$, a finite subset of $\mathbb{Z}^d$, be the neighborhood of the origin so that the translate $x + \mathcal{N}$ is the neighborhood of the site $x \in \mathbb{Z}^d$.
- Let $\mathcal{T}$ denote the CA rule. That is, $\mathcal{T} : \{0,1\}^{\mathbb{Z}^d} \to \{0,1\}^{\mathbb{Z}^d}$.
- Let $\xi_t(x) \in \{0,1\}$ denote the state of the site $x = (x_1, x_2, \ldots, x_d) \in \mathbb{Z}^d$ at time t.
- Let ξ_t represent the system at time t. The collection of 1s in ξ_t comprises some set Λ, which is contained in $\mathbb{Z}^d$. As is customary in this area, we will confound this configuration, consisting of all 1s on Λ, with the set Λ itself. Hence, if $\Lambda = \{x \in \mathbb{Z}^d : \xi_t(x) = 1\}$, we write $\xi_t = \Lambda \subset \mathbb{Z}^d$. We use $\xi_t^\Lambda = \mathcal{T}^t(\Lambda) = B$ to mean that starting with $\xi_0 = \Lambda$ and updating t time steps yields a set of 1s that lies on the set B.
- The update rule for Larger than Life is given by:

$$\xi_{t+1}(x) = \begin{cases} 1 & \text{if } \xi_t(x) = 0 \text{ and } |(x+\mathcal{N}) \cap \xi_t| \in [\beta_1, \beta_2] \text{ or} \\ & \text{if } \xi_t(x) = 1 \text{ and } |(x+\mathcal{N}) \cap \xi_t| \in [\delta_1, \delta_2]\,; \\ 0 & \text{otherwise}\,. \end{cases}$$

Translated into words, if a dead site sees between β_1 and β_2 live sites in its neighborhood at time t, it will become live at time $t+1$. Otherwise, it will remain dead at time $t+1$. If a live site sees between δ_1 and δ_2 live sites (including itself) in its neighborhood at time t, it will remain live at time $t+1$. Otherwise, it will become dead at time $t+1$. Thus, if $\Lambda \subset \mathbb{Z}^d$ is a set of 1s (on a background of

0s), then the mapping $\mathcal{T}$ is defined by

$$\mathcal{T}(\Lambda) = \{x \in \Lambda^c : \beta_1 \leq |(x+\mathcal{N}) \cap \Lambda| \leq \beta_2\} \cup \{x \in \Lambda : \delta_1 \leq |(x+\mathcal{N}) \cap \Lambda| \leq \delta_2\}.$$

Starting from an initial set $\Lambda \subset \mathbb{Z}^d$ of 1s and iterating $\mathcal{T}^{t+1}(\Lambda) = \mathcal{T}(\mathcal{T}^t(\Lambda))$ generates *LtL dynamics*. To reiterate, we denote the CA mapping from one time step to the next by $\mathcal{T}$, and use ξ_t^Λ or $\mathcal{T}^t(\Lambda)$ to denote an LtL rule that has updated t time steps starting from $\xi_0 = \Lambda$.

The LtL cellular automata form a four-parameter family of rules indexed by the endpoints of the intervals which determine each rule: β_1, β_2, δ_1, and δ_2. As such LtL can be viewed as a subset of a four-dimensional hyperspace with points $(\beta_1, \beta_2, \delta_1, \delta_2)$ representing *d-dimensional* cellular automaton rules.

Most of the examples that follow are *two-dimensional* LtL rules with *range* ρ *box neighborhoods*. That is, the rules are defined on $\mathbb{Z}^2$ with $\mathcal{N} = \{y \in \mathbb{Z}^2 : ||y||_\infty \leq \rho\}$ ($\rho \in \mathbb{N}$) (i.e., the neighborhood is a box with side length $2\rho + 1$). In this framework, *The Game of Life* is a range 1 LtL rule with parameters

$$(\beta_1, \beta_2, \delta_1, \delta_2) = (3, 3, 3, 4).$$

We note that each range $\rho \in \mathbb{N}$ determines a family of LtL rules. To specify the range in which the rule exists, we include the range when specifying the rule. Thus, the range ρ rule with parameters β_1, β_2, δ_1, and δ_2 is denoted by $(\rho, \beta_1, \beta_2, \delta_1, \delta_2)$. For example, the Game of Life is the LtL rule $(1, 3, 3, 3, 4)$.

2 LARGER-THAN-LIFE REPLICATOR EXAMPLES: SPACE-TIME DIAGRAMS

The space-time diagram of a replicator suggests a relationship with Pascal's Triangle Mod 2 in an appropriate dimension. Before getting into the technical details, we present a variety of LtL examples that illustrate this connection.

All of the replicators in this section are admitted by range 5 LtL rules. We present these because range 5 is large enough to provide a glimpse of even larger ranges yet small enough to allow for graphics that are not too unwieldy. We have many examples of replicators in other ranges that we save for later sections.

For computer simulations, the states are represented by colors. In what follows we use *black* for state 1 and *white* for state 0.

EXAMPLE 2.1

The space-time diagram depicted in figure 1 suggests a relationship with the two-dimensional version of Pascal's Triangle Mod 2. That is, every fifth time step, the rectangles evolve in a manner that is reminiscent of the evolution of the triangle's 1s. To illustrate this connection, we present the space-time diagram for only times that are multiples of five (see fig. 2).

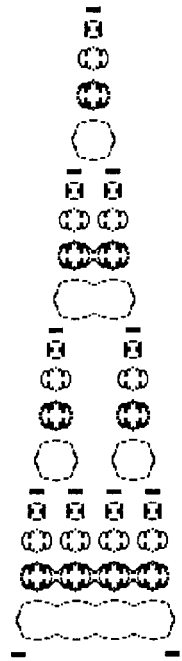

FIGURE 1 Space-time diagram for LtL rule $(5, 15, 18, 15, 26)$ with $\xi_0 = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 8,\ 0 \leq z_2 \leq 2\}$. Time moves downward and times $0, 1, 2, \ldots, 20$ are depicted.

Of course, rather than focus on the evolution of the rectangles in this example, we could have watched the evolutions of the configurations that appear at time 1 or 2. Those also replicate every five time steps. It takes ten time steps for the configurations that appear at times 3 and 4 to replicate. This will be the case in all of the examples that we present; that is, given a configuration that replicates under a CA rule, one may find a set of other configurations that also replicate, perhaps taking longer to do so.

To illustrate that the above is not unique to that example, we provide the space-time diagrams of several more examples below.

Example 2.2. In figure 3, it takes 11 time steps for the initial seed (depicted in the top row) to replicate.

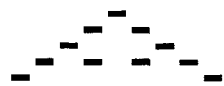

FIGURE 2 Space-time diagram for $(5, 15, 18, 15, 26)$ with $\xi_0 = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 8 \text{ and } 0 \leq z_2 \leq 2\}$ depicted in the top row. Time moves downward and times $0, 5, 10, 15$, and 20 are depicted.

FIGURE 3 Space-time diagram for LtL rule $(5, 27, 32, 26, 44)$ with $\xi_0 = \{(z_1, z_2) \in \mathbb{Z}^2 : 2 \leq z_1 \leq 4 \text{ and } z_2 = 0 \text{ or } z_2 = 4\} \cup \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 6 \text{ and } 1 \leq z_2 \leq 3\}$. Time moves downward and times $0, 1, 2, \ldots, 11$ are depicted.

Example 2.3. The initial seed in figure 4, which is not symmetrical about the vertical axis, replicates after eight time steps. One of the replicas is rotated 180 degrees.

Example 2.4. This example (see fig. 5) is distinct from examples 2.1–2.3 because its space-time diagram is symmetric about the diagonal, rather than the vertical axis. The example is generalized to other ranges in proposition 7.1.

Example 2.5. This example (see fig. 6), and the next one, have three-dimensional space-time diagrams. Their dynamics are similar to the previous examples; however, the duplication occurs in directions perpendicular to the direction of the initial seed. This provides a 90-degree rotation each time duplication occurs and thus enables the seed to propagate in two directions rather than one. In each case, we depict various cross sections of the space-time diagram.

FIGURE 4 Space-time diagram for LtL rule $(5, 11, 13, 11, 14)$ with $\xi_0 = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 6 \text{ and } 0 \leq z_2 \leq 3\} \cup \{(z_1, z_2) \in \mathbb{Z}^2 : 7 \leq z_1 \leq 8 \text{ and } 1 \leq z_2 \leq 2\}$ depicted in the top row. Time moves downward and times $0, 1, 2, \ldots, 8$ are depicted.

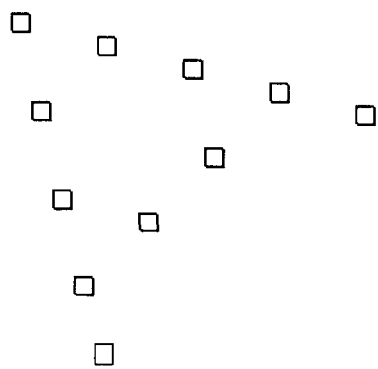

FIGURE 5 Space-time diagram for LtL rule $(5, 3, 3, 3, 3)$. Time moves downward along the diagonal and times $0, 4, 8, 12$, and 16 are depicted.

We are interested in the even times since, as will be illustrated in the next section, their space-time diagrams have a relationship with the three-dimensional version of Pascal's Triangle Mod 2. As we discussed in example 2.1, this relationship is not unique to the configuration that appears at time 0. For example, times $1 + 4k$, and times $2 + 4k$, $k = 0, 1, 2, \ldots$ also have a relationship with the three-dimensional version of Pascal's Triangle.

Example 2.6. This is the most intriguing replicator we have found. Its evolution takes up a lot of space so we depict only times 0, 35, 140, and 175. As illustrated in figure 7, after just 35 time steps, two replicas that are perpendicular to the

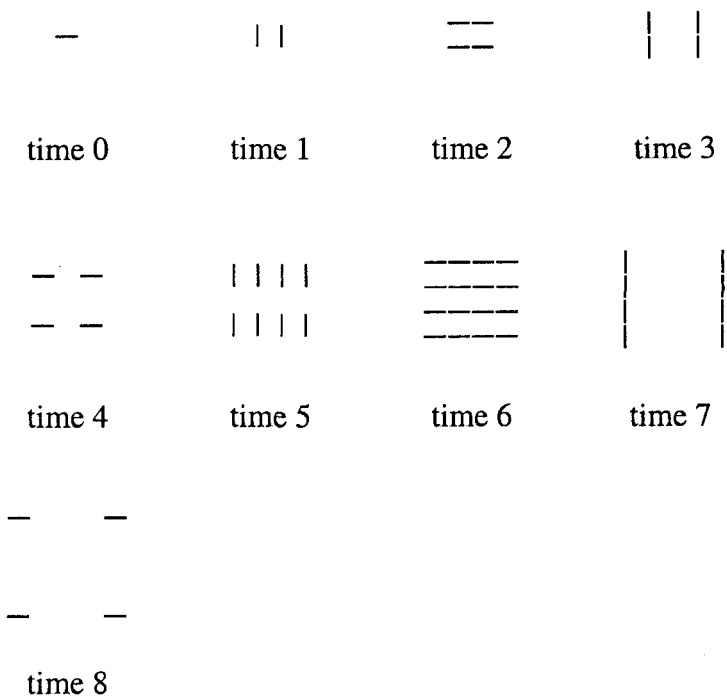

FIGURE 6 Cross sections of the space-time diagram for LtL rule $(5,5,5,5,5)$ with $\xi_0 = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 10 \text{ and } z_2 = 0\}$.

original appear. However, it is not until time 140 that exactly four copies appear. At times $140t$ the spatial orientation is just like that of example 2.5 at times $2t$, $t = 0, 1, 2, 3, \ldots$, with larger distances between copies. And at times $140t+35$, the spatial orientation is just like that of example 2.5 at times $2t+1$, $t = 0, 1, 2, 3, \ldots$. We will discuss this example further in section 5.

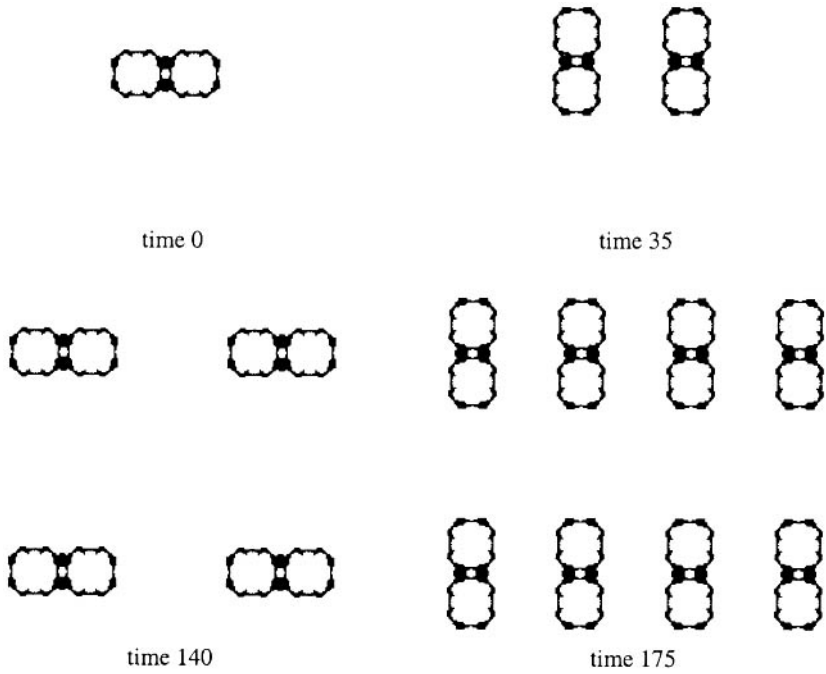

FIGURE 7 Times 0, 35, 140, and 175 for LtL rule $(5, 26, 36, 26, 41)$.

3 PASCAL-GENERATING CA RULE

Before getting into the details of the connection between the replicators' space-time diagrams and Pascal's Triangle in an appropriate dimension, let us explicitly state what we mean by the d-dimensional version of Pascal's Triangle Mod 2. First, we need the following definition.

Definition 3.1. The d-dimensional *Pascal-generating CA* rule, $\mathcal{F} : \{0,1\}^{\mathbb{Z}^d} \to \{0,1\}^{\mathbb{Z}^d}$ has neighbor set $\mathcal{N}$. The state of the site $x = (x_1, x_2, \ldots, x_d) \in \mathbb{Z}^d$ at time t is denoted by $\zeta_t(z) \in \{0,1\}$. The space-time diagram of $\mathcal{F}$ is a generalization (that depends on $\mathcal{N}$) of Pascal's Triangle Mod 2, provided the initial configuration is $\zeta_0 = \vec{0}$ (i.e., there is a single 1 at the origin and 0s elsewhere). Let us define $\mathcal{F}$ and relate it to Pascal's Triangle, following the $d = 1$ example that Durrett constructed [2, section 5d]. Let $\mathcal{N}$ be the neighborhood of $\vec{0} \in \mathbb{Z}^d$. Define

$$\zeta_{t+1}(x) = \sum_{z \in x + \mathcal{N}} \zeta_t(x + z) \bmod 2 .$$

TABLE 1 From example 3.1, $d = 1$, $\mathcal{N} = \{\pm 1\}$.

time																				
0									...	0	1	0	...							
1								...	0	1	0	1	0	...						
2							...	0	1	0	2	0	1	0	...					
3						...	0	1	0	3	0	3	0	1	0	...				
4					...	0	1	0	4	0	6	0	4	0	1	0	...			
5				...	0	1	0	5	0	10	0	10	0	5	0	1	0	...		
6			...	0	1	0	6	0	15	0	20	0	15	0	6	0	1	0	...	
7		...	0	1	0	7	0	21	0	35	0	35	0	21	0	7	0	1	0	...
8	...	0	1	0	8	0	28	0	46	0	70	0	46	0	28	0	8	0	1	0	...

For $x, y \in \mathbb{Z}^d$ define

$$f_0(y) = \begin{cases} 1 & \text{if } y = \vec{0}, \\ 0 & \text{otherwise,} \end{cases} \quad \text{and} \quad f_{t+1}(x) = \sum_{z \in x + \mathcal{N}} f_t(x + z).$$

Then f generates the $(d+1)$-dimensional generalization of Pascal's Triangle, with neighbor set $\mathcal{N}$, as long as we ignore the 0s. Also,

$$\zeta_t^{\{\vec{0}\}}(x) = f_t(x) \bmod 2.$$

When we refer to the space-time diagram of the d-dimensional Pascal-generating CA starting from $\zeta_0 = \vec{0}$, we mean the above $(d+1)$-dimensional generalization of Pascal's Triangle Mod 2. Thus, in visualizing it, we assume that the 0s have been ignored, prior to modding out by 2. Doing this acts like a shift of the state space at alternate times. Hence, the space-time diagram consists of the space where ζ_t lives at even times, and the same space shifted at odd times. For example, in dimension d with

$$\mathcal{N} = \{(z_1, z_2, \ldots, z_d) : z_i \in \{-1, 1\}, \quad i = 1, 2, \ldots, d\},$$

this is equivalent to omitting the values of $\zeta_t^{\{\vec{0}\}}(z)$ for all $z = (z_1, z_2, \ldots z_d)$ such that $t + z_i$ is odd (for some $i \in \{1, 2, \ldots, d\}$), or $|z| > t$ ($|z| = |z_1| + \ldots + |z_d|$). Hence, $\zeta_{2t} \subset \mathbb{Z}^d$ and $\zeta_{2t+1} \subset (\mathbb{Z} + 1/2)^d$ for $t = 0, 1, 2, \ldots$. This is illustrated in the examples below.

Example 3.1. See table 1; $d = 1$, $\mathcal{N} = \{\pm 1\}$. A few time steps of the action of f (empty sites have not yet seen any occupied sites in their neighborhoods) are depicted in table 1. The modded out version, which is Pascal's Triangle Mod 2, is illustrated in table 2.

TABLE 2 Constructed from table 1 by ignoring the 0s and then modding out by 2.

time																	
0								1									$\subset \mathbb{Z}$
1							1		1								$\subset \mathbb{Z}+1/2$
2						1		0		1							$\subset \mathbb{Z}$
3					1		1		1		1						$\subset \mathbb{Z}+1/2$
4				1		0		0		0		1					$\subset \mathbb{Z}$
5			1		1		0		0		1		1				$\subset \mathbb{Z}+1/2$
6		1		0		1		0		1		0		1			$\subset \mathbb{Z}$
7	1		1		1		1		1		1		1		1		$\subset \mathbb{Z}+1/2$
8	1	0		0		0		0		0		0		0		1	$\subset \mathbb{Z}$

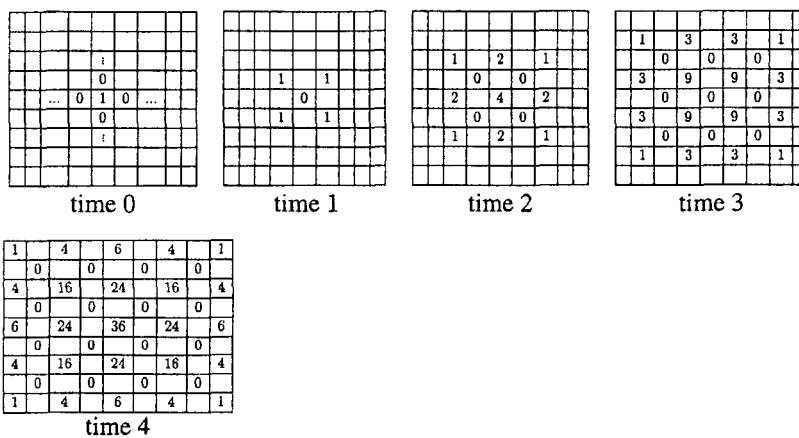

FIGURE 8 From example 3.2, $d = 2$, $\mathcal{N} = \{(z_1, z_2) \in \mathbb{Z}^2 : z_i \in \{1, -1\}, i = 1, 2\}$.

Example 3.2. Here $d = 2$, $\mathcal{N} = \{(z_1, z_2) \in \mathbb{Z}^2 : z_i \in \{1, -1\}, i = 1, 2\}$. A few time steps of the action of f (empty sites have not yet seen any occupied sites in their neighborhoods) are depicted in figure 8 and the modular version appears in figure 9.

Now let us make the connection between the Pascal-generating CAs and our replicator examples. To do this, we must find an appropriate *tiling* of $\mathbb{Z}^2$. We imagine placing a transparency (that represents $\mathbb{R}^2$) with tiles drawn on it (using lines to represent the boundaries of the tiles) over the space-time diagram in such a way that at a fixed time each replica is contained inside a distinct tile. For instance, a *rectangular* tile satisfies our condition for the replicator in example 2.1. Since that replicator propagates along only one axis and replicas appear every five time steps, we may illustrate this using a strip of the tiling at

$$1 \subset \mathbb{Z}^2 \qquad \begin{matrix} 1 & 1 \\ 1 & 1 \end{matrix} \subset (\mathbb{Z}+1/2)^2 \qquad \begin{matrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{matrix} \subset \mathbb{Z}^2$$

time 0 time 1 time 2

$$\begin{matrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{matrix} \subset (\mathbb{Z}+1/2)^2 \qquad \begin{matrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{matrix} \subset \mathbb{Z}^2$$

time 3 time 4

FIGURE 9 Figure 8 with deletions and mod 2.

each time that is a multiple of five. Let us depict two such strips of the tiling, one at time 0 and another at time 5.

After five more time steps, the replicas are inside a *shift* of the tiling:

As can be seen in the space-time diagram, every five time steps the set of live sites will be contained inside either the tiling or a shift of the tiling.

Once we find the desired tiling, we identify the replicators with the centroids of the tiles in which they lie and imagine the CA map acting on the "centroids" of the replicators. This is the sense in which the CA behaves like "addition mod 2." That is, the CA places a replica inside a shifted tile next time if there are an odd number of replicas inside the tiles surrounding it this time; otherwise, it places all 0s in the shifted tile.

To further illustrate this idea, let us construct the desired tilings for two more examples from section 2. Example 2.4 is a one-dimensional replicator, but propagates along the diagonal. Tiles in the shape of parallelograms satisfy our requirements. A strip of tiles at time 0 and its shift at time 4 are depicted in figure 10.

Finally, let us illustrate the relevant tiling for example 2.5, which is a two-dimensional replicator. In that case, the desired tiles are squares; see figure 11.

We note that a requirement for our tiling is that at any given time either *none* of the replicas intersect a boundary of the tiling or *all* of the replicas intersect a boundary of the tiling (i.e., they all lie inside a tile, or they all lie inside a *shifted* tile). We are only going to use the fact that the tiling is actually

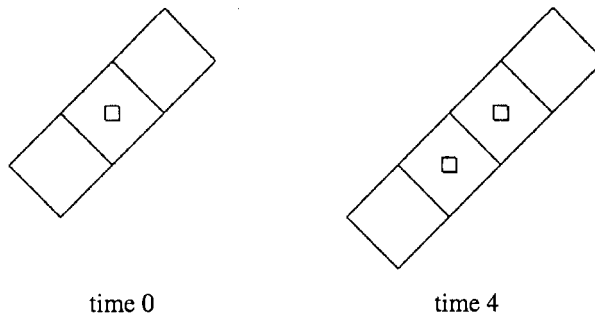

FIGURE 10 Strip of tiles at time 0 and its shift at time 4.

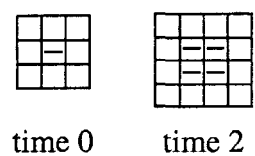

FIGURE 11 Tiling at time 0 and its shift at time 2 for the two-dimensional replicator from example 2.5.

of $\mathbb{R}^2$ to illustrate the partitioning of space and to keep track of the centroids of the tiles in which the replicas lie.

4 REPLICATOR AXIOMS

In this section we define a replicator using an axiomatic approach. That is, we list a set of axioms that one may check to confirm the discovery of a replicator. Following the axioms are two theorems that connect the axioms with Pascal's Triangle Mod 2 in an appropriate dimension. Additionally, a theorem that simplifies the work required to check axioms 5 and 6 is provided.

Before stating the axioms, we need a few definitions. The definitions are necessary to construct a *tiling* of $\mathbb{Z}^d$ that will be used to show that the CA rule of a replicator behaves like "addition mod 2" when restricted to the tiled space that has been painted with appropriately oriented copies of the replicator and 0s.

Definition 4.1. Let $\mathcal{I} \subset \mathbb{R}^d$ be a set with dimensions $\sigma_1 \times \sigma_2 \times \ldots \times \sigma_d$ in d mutually orthogonal directions and centroid placed on the origin. Let $\mathcal{A}$ be the set consisting of $\mathcal{I}$ and its disjoint translates. If $\cup_{X \in A} X = \mathbb{R}^d$, then $\mathcal{A}$ forms a *partition* for $\mathbb{R}^d$ and we may *tile* $\mathbb{R}^d$ with $\mathcal{I}$ so that all tiles are identically oriented and so that the initial tile's centroid coincides with a specified point, $(x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$.

In order to make the connection with the Pascal-generating CA rule, we focus on the *centroids* of the tiles and their images under the CA rule, $\mathcal{T}$. This requires more definitions.

Definition 4.2. The *centroids* of the tiles in definition 4.1 are contained in some set that we will denote by $\mathbb{Z}^d_{\mathcal{I}}$. Let $(\mathbb{Z} + 1/2)^d_{\mathcal{I}}$ denote the set containing the centroids of the *shifted* tiles. Let $\{0, \Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ denote the *tiled* space where each translate of $\mathcal{I}$ is inscribed with either Λ sharing its centroid, or all 0s. Similarly, let $\{0, \Lambda\}^{(\mathbb{Z}+1/2)^d_{\mathcal{I}}}$ denote the *shifted tiling* where each translate of $\mathcal{I}$ is inscribed with either Λ sharing its centroid, or all 0s. Let $\mathcal{I}^\Lambda \in \{0, \Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ denote a tile in which Λ is inscribed and $\mathcal{I}^0$ a tile containing all 0s.

In what follows, we confound the *space*, consisting of tiles inscribed with copies of Λ or all 0s, with the *set*, consisting of the centroids of the tiles inscribed with letters, $l \in \{0, \Lambda\}$ to denote whether the tile is inscribed with Λ or all 0s. We must do this because the CA rule, $\mathcal{T}$, is a map on the space while the Pascal-generating CA rule, $\mathcal{F}$, is a map on the image of the set of centroids. We use the notation $\{0, \Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ for both cases, the meaning of which will be implied via context.

Definition 4.3. A Λ^{l_ω}*-string*, $\mathcal{L}^\Lambda \subset \{0, \Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$, is a subset that consists of l_i disjoint translates of $\mathcal{I}^m$, $m \in \{0, \Lambda\}$ in direction i, $l_i \in \mathbb{Z}^+$, $i = 1, 2, \ldots, \omega$ $(\omega \leq d)$. In other words, it is an $l_1 \times l_2 \times \ldots \times l_\omega$ configuration of tiles in ω *mutually orthogonal* directions. Similarly, $\mathcal{L}^\Lambda_{1/2} \subset \{0, \Lambda\}^{(\mathbb{Z}+1/2)^d_{\mathcal{I}}}$ is a subset that consists of l_i disjoint translates of $\mathcal{I}^m$, $m \in \{0, \Lambda\}$ in direction i, $l_i \in \mathbb{Z}^+$, $i = 1, 2, \ldots, \omega$ $(\omega \leq d)$. If the configuration consists of infinitely many translates in each of the ω directions, then $l_i \to \infty$ for each i and we call this a Λ^{∞_ω}*-string*. Since each tile in the string may be one of two patterns, there are $2^{l_1 l_2 \ldots l_\omega}$ distinct Λ^{l_ω}-strings.

Definition 4.4 (Replicator Axioms). A *ν-dimensional replicator* under the d-dimensional CA rule $\mathcal{T}$ is a quadruplet $(\Lambda, \mathcal{I}, \tau, \nu)$ such that

1. Λ is a finite configuration of 1s.
2. $\nu \in \mathbb{N}_d$, $\tau \in \mathbb{N}$, $|\mathcal{T}^t(\Lambda)| < \infty$ for $t = 1, 2, \ldots, \tau$ and $\mathcal{T}^\tau(\Lambda)$ consists of exactly 2^ν *or* 2ν copies of Λ that propagate in directions $1, 2, \ldots, \nu$.
3. $\mathcal{I}$ is a set that may be used to partition $\mathcal{T}^\tau(\Lambda)$ in such a way that each copy of Λ is contained in a disjoint and adjacent translate of $\mathcal{I}$.

4. There exist bijections, $\phi : \{0,\Lambda\}^{\mathbb{Z}_I^d} \to \{0,1\}^{\mathbb{Z}^d}$ and $\phi_{1/2} : \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d} \to \{0,1\}^{(\mathbb{Z}+1/2)^d}$, where the sets $\{0,\Lambda\}^{\mathbb{Z}_I^d}$ and $\{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$ represent the *centroids* of the tiles.
5. If $\mathcal{L}^\Lambda \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$, then $\mathcal{T}^\tau(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$ and $\phi_{1/2}(\mathcal{T}^\tau(\mathcal{L}^\Lambda)) = \mathcal{F}(\phi(\mathcal{L}^\Lambda))$, where $\mathcal{F}$ is the ν-dimensional Pascal-generating CA rule from definition 3.1 with a suitable neighborhood, $\mathcal{N}$.
6. If $\mathcal{L}^\Lambda_{1/2} \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$, then $\mathcal{T}^\tau(\mathcal{L}^\Lambda_{1/2}) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$ and $\phi(\mathcal{T}^\tau(\mathcal{L}^\Lambda_{1/2})) = \mathcal{F}(\phi_{1/2}(\mathcal{L}^\Lambda_{1/2}))$, where $\mathcal{F}$ is the ν-dimensional Pascal-generating CA rule from axiom 5.

We must prove that axioms 5 and 6 are sufficient to show the same for all times that are multiples of τ. We do this in theorems 4.1 and 4.2.

Theorem 4.1. *If $\mathcal{L}^\Lambda \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$, then for $k = 1, 2, \ldots$*

1. $\mathcal{T}^{(2k-1)\tau}(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$ *and*
2. $\mathcal{T}^{2k\tau}(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$.

Proof. First, we use induction to prove part 1. The base case, $k = 1$, is given in axiom 5. Let $n \geq 1$ be a positive integer. We must show that if part 1 holds for $k = n$, then it holds for $k = n + 1$.

$$\mathcal{T}^{(2(n+1)-1)\tau}(\mathcal{L}^\Lambda) = \mathcal{T}^{2\tau}(\mathcal{T}^{(2n-1)\tau}(\mathcal{L}^\Lambda)) = \mathcal{T}^{2\tau}(\mathcal{J}^\Lambda_{1/2})$$

where $\mathcal{J}^\Lambda_{1/2} = \mathcal{T}^{(2n-1)\tau}(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$ by the induction hypothesis. Now

$$\mathcal{T}^{2\tau}(\mathcal{J}^\Lambda_{1/2}) = \mathcal{T}^{\tau}(\mathcal{T}^{\tau}(\mathcal{J}^\Lambda_{1/2})) = \mathcal{T}^{\tau}(\mathcal{Q}^\Lambda)$$

where by axiom 6, $\mathcal{Q}^\Lambda = \mathcal{T}^\tau(\mathcal{J}^\Lambda_{1/2}) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$. Finally, by axiom 5, $\mathcal{T}^\tau(\mathcal{Q}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$. ■

To prove part 2 using induction, first let us check the $k = 1$ case: $\mathcal{T}^{2\tau}(\mathcal{L}^\Lambda) = \mathcal{T}^\tau(\mathcal{T}^\tau(\mathcal{L}^\Lambda)) = \mathcal{T}^\tau(\mathcal{Q}_{1/2})$, where

$$\mathcal{Q}^\Lambda_{1/2} = \mathcal{T}^\tau(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$$

by axiom 5 and axiom 6 gives $\mathcal{T}^\tau(\mathcal{Q}_{1/2}) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$, as desired. Let $n \geq 1$ be a positive integer and assume that part 2 holds for $k = n$. Then it also holds for $k = n+1$ since $\mathcal{T}^{2(n+1)\tau}(\mathcal{L}^\Lambda) = \mathcal{T}^{2\tau}(\mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda)) = \mathcal{T}^{2\tau}(\mathcal{J}^\Lambda)$ where $\mathcal{J}^\Lambda = \mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$, by the induction hypothesis. Finally, we already showed that $\mathcal{T}^{2\tau}(\mathcal{J}^\Lambda) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$. ■

Theorem 4.2. *If $\mathcal{L}^\Lambda \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$ and $\mathcal{F}$ is the ν-dimensional Pascal-generating CA rule from definition 2.1 with a suitable neighborhood, $\mathcal{N}$, then for $k = 1, 2, \ldots$,*

1. $\phi(\mathcal{T}^{2k\tau}(\mathcal{L}^\Lambda)) = \mathcal{F}^{2k}(\phi(\mathcal{L}^\Lambda))$ *and*
2. $\phi_{1/2}(\mathcal{T}^{(2k-1)\tau}(\mathcal{L}^\Lambda)) = \mathcal{F}^{2k-1}(\phi(\mathcal{L}^\Lambda))$.

Proof. First, we prove part 1 using induction. Let us check the basis case: $\phi(\mathcal{T}^{2\tau}(\mathcal{L}^\Lambda)) = \phi(\mathcal{T}^\tau(\mathcal{T}^\tau(\mathcal{L}^\Lambda))) = \phi(\mathcal{T}^\tau(\mathcal{J}_{1/2})$, where by axiom 5, $\mathcal{J}_{1/2} = \mathcal{T}^\tau(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$ and axiom 6 gives $\phi(\mathcal{T}^\tau(\mathcal{J}_{1/2})) = \mathcal{F}(\phi_{1/2}(\mathcal{J}_{1/2}))$. Substitution and axiom 5 yield

$$\mathcal{F}(\phi_{1/2}(\mathcal{J}_{1/2})) = \mathcal{F}(\phi_{1/2}(\mathcal{T}^\tau(\mathcal{L}^\Lambda))) = \mathcal{F}(\mathcal{F}(\phi(\mathcal{L}^\Lambda))) = \mathcal{F}^2(\phi(\mathcal{L}^\Lambda)),$$

as desired. Now let $n \geq 1$ be a positive integer.

$$\phi(\mathcal{T}^{2(n+1)\tau}(\mathcal{L}^\Lambda)) = \phi(\mathcal{T}^{2\tau}(\mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda))) = \phi(\mathcal{T}^{2\tau}(\mathcal{J}^\Lambda)),$$

where $\mathcal{J}^\Lambda = \mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$ by theorem 4.1, part 2. By the basis case, $\phi(\mathcal{T}^{2\tau}(\mathcal{J}^\Lambda)) = \mathcal{F}^2(\phi(\mathcal{J}^\Lambda))$. Substitution and the induction hypothesis yield

$$\mathcal{F}^2(\phi(\mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda))) = \mathcal{F}^2(\mathcal{F}^{2n}(\phi(\mathcal{L}^\Lambda))) = \mathcal{F}^{2n+2}(\phi(\mathcal{L}^\Lambda)),$$

as desired. ∎

Proof. Now let us do an inductive proof of part 2. The basis case, $k = 1$, is given in axiom 5. Let $n \geq 1$ be a positive integer.

$$\phi_{1/2}(\mathcal{T}^{(2(n+1)-1)\tau}(\mathcal{L}^\Lambda)) = \phi_{1/2}(\mathcal{T}^{2\tau}(\mathcal{T}^{(2n-1)\tau}(\mathcal{L}^\Lambda))) = \phi_{1/2}(\mathcal{T}^{2\tau}(\mathcal{J}^\Lambda_{1/2})),$$

where $\mathcal{J}^\Lambda_{1/2} = \mathcal{T}^{(2n-1)\tau}(\mathcal{L}^\Lambda) \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)_I^d}$ by theorem 4.1, part 1. Now $\phi_{1/2}(\mathcal{T}^\tau(\mathcal{T}^\tau(\mathcal{J}^\Lambda_{1/2}))) = \phi_{1/2}(\mathcal{T}^\tau(\mathcal{Q}^\Lambda))$ where $\mathcal{Q}^\Lambda = \mathcal{T}^\tau(\mathcal{J}^\Lambda_{1/2}) \subset \{0,\Lambda\}^{\mathbb{Z}_I^d}$ by axiom 6. Thus, by axiom 5 $\phi_{1/2}(\mathcal{T}^\tau(\mathcal{Q}^\Lambda)) = \mathcal{F}(\phi(\mathcal{Q}^\Lambda))$. Substitution yields

$$\begin{aligned}\mathcal{F}(\phi(\mathcal{Q}^\Lambda)) &= \mathcal{F}(\phi(\mathcal{T}^\tau(\mathcal{J}^\Lambda_{1/2}))) = \mathcal{F}(\phi(\mathcal{T}^\tau(\mathcal{T}^{(2n-1)\tau}(\mathcal{L}^\Lambda))))\\ &= \mathcal{F}(\phi(\mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda))).\end{aligned}$$

By part 1 of this theorem,

$$\mathcal{F}(\phi(\mathcal{T}^{2n\tau}(\mathcal{L}^\Lambda))) = \mathcal{F}(\mathcal{F}^{2n}(\phi(\mathcal{L}^\Lambda))) = \mathcal{F}^{2n+1}(\phi(\mathcal{L}^\Lambda)),$$

as desired. ∎

Corollary 4.1.

1. If the Pascal-generating CA rule ζ_t has $\zeta_0 = \vec{0}$ and neighborhood $\mathcal{N} = \{(z_1, z_2, \ldots, z_d) : z_i \in \{-1, 1\},\ i = 1, 2, \ldots, d\}$, then
   - starting from $\xi_0 = \mathcal{I}_{\vec{0}}^{\Lambda}$, $(\Lambda, \mathcal{I}, \tau, \nu)$ is a sawtooth pattern with fixed low value $2^{\nu}|\Lambda|$. This value is obtained at times $2^n\tau$, $n = 0, 1, 2, \ldots$.
   - The space-time diagram of $(\Lambda, \mathcal{I}, \tau, \nu)$ is self-similar and has fractal dimension $\log_2(2^{\nu} + 1)$.
2. If the Pascal-generating CA rule has $\mathcal{N} = \{\pm e_i :\ i = 1, 2, \ldots, d\}$, then
   - starting from $\xi_0 = \mathcal{I}_{\vec{0}}^{\Lambda}$, $(\Lambda, \mathcal{I}, \tau, \nu)$ is a sawtooth pattern with fixed low value $2\nu|\Lambda|$. This value is obtained at times $2^n\tau$, $n = 0, 1, 2, \ldots$.
   - The space-time diagram of $(\Lambda, \mathcal{I}, \tau, \nu)$ is self-similar and has fractal dimension $\log_2(2\nu + 1)$.

Proof. The first parts of both 1 and 2 follow from ϕ and $\phi_{1/2}$ together with the proof provided in Lind [8]. For the second parts, see Durrett [2, section 5d] or Wolfram [9, p. 454].

The next theorem shows that the work we must do to check axioms 5 and 6 may be simplified. In fact, rather than check Λ^{l_ν}-strings that are arbitrarily long in each direction, we need only check that the $\mathcal{I}^{\Lambda}$s in strings of length $2k_i + 3$ in the $i = 1, 2, \ldots, \nu$ mutually orthogonal directions do not destroy one another's evolutions after τ time steps. Before stating the theorem we need a definition.

Definition 4.5. Constructed as follows, the *M-lightcone* of a site x contains $\xi_t^{\{x\}}$ for all $t \in \{0, 1, \ldots, M\}$. Let $\mathcal{N}$ be the neighborhood for the CA rule, ξ_t. Fix $i \in \{1, \ldots, d\}$ and let

$$s_i = \max_{x \in \mathcal{N}} |x_i| .$$

Then an occupied site can propagate outward at most s_i sites after one time step in each of the i *mutually orthogonal* directions. We call s_i the *speed of light* in direction i.

Fix $x = (x_1, x_2, \ldots, x_d)$. For each $t \in \{0, 1, \ldots, M\}$, let $\mathcal{H}^t$ be the "hyperrectangle" with side of length $2ts_i + 1$ in direction i and whose edges are determined by the sites $x_1 \pm ts_i$, $i = 1, 2, \ldots, d$. Construct the $(d + 1)$-dimensional configuration that consists of a copy of $\mathcal{H}^t$ at each time $t \in \{0, 1, \ldots, M\}$. This is the M-lightcone of x. Observe that after M time steps, the only sites that might feel the effect from the evolution of x (meaning that their neighbor sets might contain 1s generated by x), are contained in the M-lightcone. Thus, the set of occupied sites in the space-time diagram generated by x is contained in the M-lightcone of x.

If Ω is a finite configuration of occupied sites, define its M-lightcone to be the union of the M-lightcones of all of its elements.

Example 4.1. $\mathcal{T}$ is a d-dimensional LtL rule and $\mathcal{N}$ is the range ρ box neighborhood. Then ρ is the speed of light in every direction (so $s_i = \rho\ \forall i$) and $\mathcal{H}^t$ is the d-dimensional hyper cube with side length $2t\rho + 1$ and edges determined by $x_i \pm t\rho$, $i = 1, 2, \ldots, d$.

Theorem 4.3. *Suppose we are given the d-dimensional CA rule, $\mathcal{T}$, and the quadruplet, $(\Lambda, \mathcal{I}, \tau, \nu)$, $\nu \leq d$. Let λ_i be the number of sites in direction i required for the smallest hyperrectangle in which Λ may be inscribed. Let σ_i be the maximum number of sites in direction i that $\mathcal{I}$ comprises and let s_i be the speed of light in direction i, $i = 1, 2, \ldots, \nu$ (the ν directions must be mutually orthogonal). Let*

$$k_i = \left\lfloor \frac{2s_i\tau - (\sigma_i - \lambda_i)}{\sigma_i} \right\rfloor, \quad i = 1, 2, \ldots, \nu\,.$$

Let $\mathcal{F}$ be the ν-dimensional Pascal-generating CA rule with neighborhood $\mathcal{N}$. If

1. *$\phi_{1/2}(\mathcal{T}^\tau(\mathcal{L}^\Lambda)) = \mathcal{F}(\phi(\mathcal{L}^\lambda))$, for each distinct $\Lambda^{2k_\nu+3}$-string, $\mathcal{L}^\Lambda \subset \{0, \Lambda\}^{\mathbb{Z}_\mathcal{I}^d}$ (on a background of 0s) and*
2. *$\phi(\mathcal{T}^\tau(\mathcal{L}^\Lambda_{1/2})) = \mathcal{F}(\phi_{1/2}(\mathcal{L}^\Lambda_{1/2}))$, for each distinct $\Lambda^{2k_\nu+3}$-string, $\mathcal{L}^\Lambda_{1/2} \subset \{0, \Lambda\}^{(\mathbb{Z}+1/2)_\mathcal{I}^d}$ (on a background of 0s),*

then axioms 5 and 6 hold. In other words, instead of checking all *strings, $\mathcal{L}^\Lambda$ and $\mathcal{L}^\Lambda_{1/2}$, it suffices to check only strings of length k_i, in directions $i = 1, 2, \ldots, \nu$, k_i defined above.*

Proof. In order for axioms 5 and 6 to hold, parts 1 and 2 must be satisfied for Λ^{∞_ν}-strings. We must show that we may restrict our attention to $\Lambda^{2k_\nu+3}$-strings.

We begin by showing part 1. Fix $\mathcal{I}^\Lambda \in \{0, \Lambda\}^{\mathbb{Z}_\mathcal{I}^d}$, a translate of $\mathcal{I}$ that is inscribed with Λ sharing its centroid. Compute the maximum number, k_i, of translates of $\mathcal{I}$ painted with all 0s, that may be placed between $\mathcal{I}^\Lambda$ and another translate of $\mathcal{I}$ that is inscribed with Λ in direction i, so that the evolutions of the copies of Λ might intersect by time τ. Since $\mathcal{I}$ comprises σ_i sites in direction i, there will be at least $(k_i + 1)\sigma_i - \Lambda_i$ 0s in direction i between these copies of Λ.

The evolutions in direction i of the Λ's might interact by time τ if their respective τ-lightcones intersect in direction i. The fastest each evolution can propagate is the speed of light, or $s_i\tau$ sites every τ time steps. Since these evolutions are moving toward one another, the only way they can interact by time τ is if they were separated by fewer than or equal to $2s_i\tau$ 0s, at time 0. Thus, the τ-lightcones will intersect if and only if $2s_i\tau \geq (k_i + 1)\sigma_i - \lambda_i$. Solving the inequality yields $[2s_i\tau - (\sigma_i - \lambda_i)]/\sigma_i \geq k_i$ and,

since $k_i \in \{0, 1, 2, \ldots\}$, take

$$k_i = \left\lfloor \frac{2s_i\tau - (\sigma_i - \lambda_i)}{\sigma_i} \right\rfloor .$$

By symmetry, the same number of tiles painted with all 0s can be placed between the original $\mathcal{I}^\Lambda$ and another copy, on the opposite side, in direction i. Thus, we may restrict our attention to subsets of $\{0, \Lambda\}^{\mathbb{Z}_\mathcal{I}^d}$ that consist of $2k_i+3$ disjoint translates of $\mathcal{I}$ in direction i in which are inscribed with copies of Λ or all 0s. For each such configuration in $\{0, \Lambda\}^{\mathbb{Z}_\mathcal{I}^d}$, we must run the rule for τ time steps and check that part 1 holds. This will ensure that the $\mathcal{I}^\Lambda$s do not destroy one another's evolutions. Since the rule and neighborhood are symmetric, we need only check unique cases modulo translation and symmetry.

The argument for part 2 is identical with $\{0, \Lambda\}^{\mathbb{Z}_\mathcal{I}^d}$ replaced by $\{0, \Lambda\}^{(\mathbb{Z}+1/2)_\mathcal{I}^d}$. ■

5 CONSTRUCTION OF THE SET, A TILING, AND THE BIJECTIONS FOR A SPECIFIC CLASS OF REPLICATORS

In this section we find the set $\mathcal{I}$ and construct the bijections, ϕ and $\phi_{1/2}$, in axiom 4 for replicators for which the neighborhood, $\mathcal{N}$, of the relevant Pascal-generating CA rule given in axioms 5 and 6 is $\mathcal{N} = \{(z_1, z_2, \ldots, z_d) : z_i \in \{-1, 1\},\ i = 1, 2, \ldots, d\}$. For these replicators 2^ν copies of Λ appear at time τ. The entire discussion generalizes to replicators that have different neighborhoods provided an appropriate set $\mathcal{I}$ and mappings can be found. We discuss one such example in section 6.

In order to find $\mathcal{I}$ and construct the bijections, we must define precise parameters λ_k, γ_k, and σ_k, $k = 1, 2, \ldots, d$ that we will use to tile and map the space accordingly. We begin with λ_k, but first need the following.

Since Λ is a finite and discrete set, $\Lambda = \{z_1, z_2, \ldots, z_n\}$ for some $n < \infty$, where $z_j = (z_{j1}, z_{j2}, z_{j3}, \ldots, z_{jd})$, $j = 1, 2, 3, \ldots, n$. By axiom 2, $|\mathcal{T}^\tau(\Lambda)| < \infty$ for $t = 0, 1, \ldots, \tau$; thus, after $t \leq \tau$ time steps, the configuration of 1s that represents the image of Λ is $\mathcal{T}^t(\Lambda) = \{z_1^t, z_2^t, \ldots, z_{n_t}^t\}$, for some $n_t < \infty$.

- Let

$$\lambda_k = \begin{cases} \max_{0 \leq i,j \leq d} |z_{ik} - z_{jk}| + 1, & \text{if } k = 1, 2, 3, \ldots, \nu; \\ \max_{0 \leq t \leq \tau} (\max_{0 \leq i,j \leq d} |z_{ik}^t - z_{jk}^t|) + 1, & \text{if } k = \nu+1, \nu+2, \ldots, d. \end{cases}$$

Translated into words, for $k = 1, 2, 3, \ldots, \nu$, λ_k represents the number of sites in direction k required for the smallest hyperrectangle $\mathcal{S} \subset ((1/2)\mathbb{Z})^d$ in which Λ may be inscribed. For $k = \nu+1, \nu+2, \ldots, d$, λ_k represents the number

of sites in direction k required for the smallest hyperrectangle $S \subset ((1/2)\mathbb{Z})^d$ in which $\mathcal{T}^t(\Lambda)$ may be inscribed, where t $(0 \leq t \leq \tau)$ represents the time for which this number is the largest.

- Let $S \subset ((1/2)\mathbb{Z})^d$ be a $\lambda_1 \times \lambda_2 \times \lambda_3 \times \ldots \times \lambda_d$ hyperrectangle oriented so that it has one vertex at the origin and lies in the region of $((1/2)\mathbb{Z})^d$ where all of the coordinates are greater than or equal to zero. That is, $\mathcal{S} = \{(z_1, z_2, \ldots, z_d) \in ((1/2)\mathbb{Z})^d : 0 \leq z_k \leq \lambda_k - 1, k = 1, 2, \ldots, d\}$. Inscribe Λ in S so that the centroid of Λ (which does not necessarily lie on Λ) coincides with the centroid, $((\lambda_1 - 1)/2, (\lambda_2 - 1)/2, (\lambda_3 - 1)/2, \ldots, (\lambda_d - 1)/2)$, of $\mathcal{S}$.

For instance, in example 2.1, $\Lambda = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 8, 0 \leq z_2 \leq 2\}$ so $\lambda_1 = 9$. To compute λ_2 we use the space-time diagram which shows that the length in direction 2 is largest when $t = 4$. The maximum length in direction 2 of the rectangle required to contain $\mathcal{T}^4(\Lambda)$ is $\lambda_2 = 25$. Thus, $\mathcal{S} = \{(z_1, z_2) \in ((1/2)\mathbb{Z})^2 : 0 \leq x_1 \leq 8$ and $0 \leq x_2 \leq 24\}$ and we embed Λ in $\mathcal{S}$ by placing the centroid, $(4, 1)$, of Λ on the centroid, $(4, 12)$, of S.

Let γ_k be the number of sites in direction k, $k = 1, 2, \ldots, \nu$ between the 2^ν copies of Λ that axiom 2 gives at time τ. The 2^ν copies of Λ are translations (and possibly rotations) of $\Lambda = \{z_1, z_2, \ldots, z_n\}$ and propagate in directions $1, 2, \ldots, \nu$. Write each of the 2^ν copies as $\Lambda_m = \{z_{1,m}, z_{2,m}, \ldots, z_{n,m}\}$, $m = 1, 2, \ldots, 2^\nu$, where $z_{j,m} = (z_{j1,m}, z_{j2,m}, z_{j3,m}, \ldots, j_{jd,m})$, $j = 1, 2, 3, \ldots, n$. Then

$$\begin{aligned} \mathcal{T}^\tau(\Lambda) &= \{\Lambda_1, \Lambda_2, \Lambda_3, \ldots, \Lambda_{2^\nu}\} \\ &= \{z_{1,1}, z_{2,1}, \ldots, z_{n,1}, z_{1,2}, z_{2,2}, \ldots, z_{n,2}, \ldots, z_{1,2^\nu}, z_{2,2^\nu}, \ldots z_{n,2^\nu}\}. \end{aligned}$$

Let us compute γ_k explicitly.

- Let

$$\gamma_k = \begin{cases} \max_{1 \leq l,m \leq 2^\nu, l \neq m} (\min_{0 \leq i,j \leq d} |z_{ik,l} - z_{jk,m}|) - 1, & k = 1, 2, \ldots, \nu; \\ 2\rho + 1, k = \nu + 1, \nu + 2, \ldots, d. \end{cases}$$

In directions $\nu+1, \nu+2, \ldots, d$, there are no additional copies of Λ (since it is a ν-dimensional replicator) so we use $\gamma_k = 2\rho + 1$. This provides a band of 0s of width $\rho + (1/2)$ on each side of S in the additional directions. The band of 0s is included to prevent interaction among disjoint replicators in adjacent subspaces. It is necessary because axiom 4 requires that ϕ and $\phi_{1/2}$ be defined on $\{0, \Lambda\}^{\mathbb{Z}_I^d}$ and $\{0, \Lambda\}^{(\mathbb{Z}+(1/2))_I^d}$, respectively but the ν-dimensional replicator propagates in only ν directions.

- Let $\sigma_k = \lambda_k + \gamma_k$, $k = 1, 2, \ldots, d$.

For instance, in example 2.1, $\nu = 1$ and $\mathcal{T}^5(\Lambda) = \{(z_1, z_2) \in \mathbb{Z}^2 : -12 \leq z_1 \leq -4, 0 \leq z_2 \leq 2\} \cup \{(z_1, z_2) \in \mathbb{Z}^2 : 12 \leq z_1 \leq 20, 0 \leq z_2 \leq 2\}$. Thus, $\gamma_1 = 15$, $\sigma_1 = 24$, and $\sigma_2 = 36$.

- Let $\mathcal{I} \subset \mathbb{R}^d$ be a set that comprises σ_k sites of $\mathbb{Z}^d$ in direction k.

Tile $\mathbb{R}^d$ with $\mathcal{I}$ (as described in definition 4.1) so that all of the tiles are identically oriented and the initial tile's centroid coincides with the centroid $((\lambda_1 - 1)/2, (\lambda_2 - 1)/2, \ldots, (\lambda_d - 1)/2)$ of S. Then each tile's centroid is in the set $\mathbb{Z}^d_{\mathcal{I}} = \sigma_1\mathbb{Z} \times \sigma_2\mathbb{Z} \times \ldots \times \sigma_d\mathbb{Z} + ((\lambda_1 - 1)/2, (\lambda_2 - 1)/2, \ldots, (\lambda_d - 1)/2)$. The centroids of the shifted tiles are in the set $(\mathbb{Z} + (1/2))^d_{\mathcal{I}} = \sigma_1(\mathbb{Z} + (1/2)) \times \sigma_2(\mathbb{Z} + (1/2)) \times \ldots \times \sigma_d(\mathbb{Z} + (1/2)) + ((\lambda_1 - 1)/2, (\lambda_2 - 1)/2, \ldots, (\lambda_d - 1)/2)$.

- Since each tile's centroid has the form $(\sigma_1 n_1 + (\lambda_1 - 1)/2, \sigma_2 n_2 + (\lambda_2 - 1)/2, \ldots, \sigma_d n_d + (\lambda_d - 1)/2) \in \mathbb{Z}^d_{\mathcal{I}} \cup (\mathbb{Z} + (1/2))^d_{\mathcal{I}}$, let $\mathcal{I}^l_{(n_1,n_2,\ldots,n_d)} \in \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}} \cup \{0,\Lambda\}^{(\mathbb{Z}+(1/2))^d_{\mathcal{I}}}$ denote *either* the tile or its centroid, whichever is appropriate in the given context, inscribed with $l \in \{0,\Lambda\}$ ($\{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ and $\{0,\Lambda\}^{(\mathbb{Z}+(1/2))^d_{\mathcal{I}}}$ are defined in definition 4.1).
- If $\mathcal{I}^l_{(n_1,n_2,\ldots,n_d)} \in \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$, then by theorem 4.1 the image of the tile under the CA rule after τt time steps, $\mathcal{T}^{\tau t}(\mathcal{I}^l_{(n_1,n_2,\ldots,n_d)})$ will be in $\{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ if t is even and in $\{0,\Lambda\}^{(\mathbb{Z}+(1/2))^d_{\mathcal{I}}}$ if t is odd.
- Define $\phi : \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}} \to \{0,1\}^{\mathbb{Z}^d}$ by $\phi(\mathcal{I}^l_{(n_1,n_2,\ldots,n_d)}) = (n_1, n_2, \ldots, n_d)^{\phi(l)}$, where

$$\phi(l) = \begin{cases} 1 & \text{if } l = \Lambda; \\ 0 & \text{if } l = 0. \end{cases}$$

- Define $\phi_{1/2} : \{0,\Lambda\}^{(\mathbb{Z}+(1/2))^d_{\mathcal{I}}} \to \{0,1\}^{(\mathbb{Z}+(1/2))^d}$ by $\phi_{1/2}(\mathcal{I}^l_{(n_1,n_2,\ldots,n_d)}) = (n_1, n_2, \ldots, n_d)^{\phi_{1/2}(l)}$, where

$$\phi_{1/2}(l) = \begin{cases} 1 & \text{if } l = \Lambda; \\ 0 & \text{if } l = 0. \end{cases}$$

We prove that ϕ and $\phi_{1/2}$ are bijections in the appendix.

Definition 5.1. The *projection maps*,

$$\pi : \{0,1\}^{\mathbb{Z}^d} \to \{0,1\}^{\mathbb{Z}^\nu} \text{ and } \pi_{1/2} : \{0,1\}^{(\mathbb{Z}+(1/2))^d} \to \{0,1\}^{(\mathbb{Z}+(1/2))^\nu}$$

are defined by

$$\begin{aligned} \pi((n_1, n_2, \ldots, n_d)^l) &= (n_1, n_2, \ldots, n_\nu)^l \text{ and} \\ \pi((m_1, m_2, \ldots, m_d)^l) &= (m_1, m_2, \ldots, m_\nu)^l, \end{aligned}$$

respectively, where $l \in \{0,1\}$.

Proposition 5.1. Suppose the quadruplet $(\Lambda, \mathcal{I}, \tau, \nu)$ is a replicator under the CA rule, $\mathcal{T}$, with relevant ν-dimensional Pascal-generating CA, $\mathcal{F}$, that has neighborhood $\mathcal{N} = \{(z_1, z_2, \ldots, z_\nu) : z_i \in \{-1, 1\}, i = 1, 2, \ldots, \nu\}$ and $\mathcal{I}$, ϕ, and $\phi_{1/2}$ constructed as above and π and $\pi_{1/2}$ from definition 5.1.

1. If $\nu = 1$ and $\mathcal{I}$ comprises σ_1 sites in the direction along which Λ propagates, then the following diagram illustrates the locations of the centroids of the tiles under the map $\mathcal{T}^\tau$ and the connection via ϕ and $\phi_{1/2}$ with $\mathcal{F}$, which generates Pascal's Triangle Mod 2.

$\mathcal{I}^\Lambda_{(0,0)}$	$\overset{\pi\circ\phi}{\rightarrow}$	$(0)^1$
$\mathcal{T}^\tau \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^\Lambda_{((-\sigma_1/2),0)}, \mathcal{I}^\Lambda_{((\sigma_1/2),0)}\}$	$\overset{\pi_{1/2}\circ\phi_{1/2}}{\rightarrow}$	$\{(-1/2)^1, (1/2)^1\}$
$\mathcal{T}^\tau \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^\Lambda_{(-\sigma_1,0)}, \mathcal{I}^0_{(0,0)}, \mathcal{I}^\Lambda_{(\sigma_1,0)}\}$	$\overset{\pi\circ\phi}{\rightarrow}$	$\{(-1)^1, (0)^0, (1)^1\}$

2. If $\nu = 2$ and $\mathcal{I}$ comprises σ_1 sites in direction 1 and σ_2 sites in direction 2 (where Λ propagates along these mutually orthogonal directions), then the following diagram illustrates the locations of the centroids of the tiles under the map $\mathcal{T}^\tau$ and the connection with $\mathcal{F}$, which generates the three-dimensional version of Pascal's Triangle Mod 2.

$\mathcal{I}^\Lambda_{(0,0)}$	$\overset{\phi}{\rightarrow}$	$(0,0)^1$
$\mathcal{T}^\tau \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^\Lambda_{(\frac{-\sigma_1}{2},\frac{-\sigma_2}{2})}, \mathcal{I}^\Lambda_{(\frac{-\sigma_1}{2},\frac{\sigma_2}{2})}, \mathcal{I}^\Lambda_{(\frac{\sigma_1}{2},\frac{\sigma_2}{2})}, \mathcal{I}^\Lambda_{(\frac{\sigma_1}{2},\frac{-\sigma_2}{2})}\}$	$\overset{\phi_{1/2}}{\rightarrow}$	$\{(\frac{-1}{2},\frac{-1}{2})^1, (\frac{-1}{2},\frac{1}{2})^1, (\frac{1}{2},\frac{1}{2})^1, (\frac{1}{2},\frac{-1}{2})^1\}$
$\mathcal{T}^\tau \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^\Lambda_{(-\sigma_1,-\sigma_2)}, \mathcal{I}^0_{(-\sigma_1,0)}, \mathcal{I}^\Lambda_{(-\sigma_1,\sigma_2)}, \mathcal{I}^0_{(0,\sigma_2)}, \mathcal{I}^\Lambda_{(\sigma_1,\sigma_2)}, \mathcal{I}^0_{(\sigma_1,0)}, \mathcal{I}^\Lambda_{(\sigma_1,-\sigma_2)}, \mathcal{I}^0_{(0,-\sigma_2)}\mathcal{I}^0_{(0,0)}\}$	$\overset{\phi}{\rightarrow}$	$\{(-1,-1)^1, (-1,0)^0, (-1,1)^1, (0,1)^0, (1,1)^1, (1,0)^0, (1,-1)^1, (0,-1)^0, (0,0)^0\}$

Proof. Both parts 1 and 2 follow by construction of $\mathcal{I}$, ϕ, and $\phi_{1/2}$ and axioms 5 and 6.

Revisiting Example 2.6. We may use part 2 of proposition 5.1 with $\sigma_1 = \sigma_2 = 144$, and $\tau = 140$ to compute the distances between the copies of the replicator at each time that is a multiple of 140 (also note that $\lambda_1 = 72$, $\lambda_2 = 32$).

Corollary 5.1. For $k = 0, 1, 2, \ldots, \phi$ and $\phi_{1/2}$ restricted to ν dimensions and followed by the projection maps π and $\pi_{1/2}$ from definition 5.1 (i.e.,

$$\begin{aligned}\pi \circ \phi \quad &: \{0,\Lambda\}^{\mathbb{Z}_{\mathcal{I}}^{\nu} \times \left[\frac{\lambda_{\nu+1}-1}{2} - \frac{\sigma_{\nu+1}}{2}, \frac{\lambda_{\nu+1}-1}{2} + \frac{\sigma_{\nu+1}}{2}\right] \times \ldots \times \left[\frac{\lambda_d - 1}{2} - \frac{\sigma_d}{2}, \frac{\lambda_d - 1}{2} + \frac{\sigma_d}{2}\right]} \\ &\to \{0,1\}^{\mathbb{Z}^{\nu}}\end{aligned}$$

and

$$\begin{aligned}\pi_{\frac{1}{2}} \circ \phi_{\frac{1}{2}} \quad &: \{0,\Lambda\}^{(\mathbb{Z}+\frac{1}{2})_{\mathcal{I}}^{\nu} \times \left[\frac{\lambda_{\nu+1}-1}{2} - \frac{\sigma_{\nu+1}}{2}, \frac{\lambda_{\nu+1}-1}{2} + \frac{\sigma_{\nu+1}}{2}\right] \times \ldots \times \left[\frac{\lambda_d - 1}{2} - \frac{\sigma_d}{2}, \frac{\lambda_d - 1}{2} + \frac{\sigma_d}{2}\right]} \\ &\to \{0,1\}^{(\mathbb{Z}+\frac{1}{2})^{\nu}}),\end{aligned}$$

1. $\pi(\phi(\mathcal{T}^{2k\tau}(\mathcal{I}_{\vec{0}}^{\Lambda}))) = \mathcal{F}^{2k}(\phi(\mathcal{I}_{\vec{0}}^{\Lambda})) \subset \{0,1\}^{\mathbb{Z}^{\nu}}$ and
2. $\pi_{1/2}(\phi_{1/2}(\mathcal{T}^{(2k+1)\tau}(\mathcal{I}_{\vec{0}}^{\Lambda}))) = \mathcal{F}^{2k+1}(\phi(\mathcal{I}_{\vec{0}}^{\Lambda})) \subset \{0,1\}^{(\mathcal{Z}+(1/2))^{\nu}}$.

Translated into words, there is a one-to-one correspondence between the space-time diagram of the replicator starting on a background of 0s and the space-time diagram of the Pascal-generating CA rule starting with a single live site at the origin, provided we ignore the zeros prior to modding out in the latter case (see definition 3.1).

Proof. Both parts 1 and 2 follow from proposition 5.1 and theorems 4.1 and 4.2.

Revisiting Example 2.1. Let us show that example 2.1 is a one-dimensional replicator. We have already shown that axioms 1–4 hold. Recall that the LtL rule is $(5, 15, 18, 15, 26)$ and Λ is a 9×3 rectangle. In section 5 we computed $\lambda_1 = 9$, $\lambda_2 = 25$, and $\tau = 5$. We also found $\sigma_1 = 24$ and $\sigma_2 = 36$, the dimensions of the rectangle $\mathcal{I}$ used for the tiling depicted in section 2 and we constructed the maps ϕ and $\phi_{1/2}$.

Begin with a single copy of Λ inscribed in the tile $\mathcal{I}$ whose centroid, $(23/2, 35/2)$ is placed on the "center" $(4, 12)$ of the evolution of Λ. Denote this by $(0,0)^{\Lambda}$. By construction of $\mathcal{I}$ we may apply proposition 5.1, part 1, to see that the first parts of axioms 5 and 6 hold.

To check the remaining parts of axioms 5 and 6, we need only check Λ^5-strings since theorem 4.3 gives

$$k_1 = \left\lfloor \frac{2(5)(5) - (24 - 9)}{24} \right\rfloor = 1\,.$$

Let us do the required checking. Since relevant configurations are strings of length 5, there are at most 2^5 cases that must be checked. However, in the set of such possibilities many of the cases are irrelevant or redundant. First of all, $k_1 = 1$ means that tiles inscribed with copies of Λ which are separated by 2 or more

FIGURE 12 The relevant Λ^5-strings for example 2.1.

other tiles consisting only of 0s will not feel the effects of one another's evolutions by time 5. This restriction eliminates six of the cases. Identifying configurations which are equivalent, either mod translation or via a 180-degree rotation, eliminates another 16 cases. Hence, only the ten cases depected in figure 12 require checking. The checking consists of showing that each configuration (assuming there are tiles consisting of only 0s surrounding those that are depicted) satisfies parts 1 and 2 of theorem 4.3.

In order to do this checking, we ran the above rule on each of the first six distinct initial states, for $\tau = 5$ time steps each. See figure 13 for the resulting diagrams, shown actual size.

The last space-time diagram is actually the union of two copies of the first since the copies do not interact by time 5. This shows that the evolutions of the copies of Λ do not interact unless they are contained in adjacent tiles. Thus, the remaining four cases evolve as disjoint unions of the cases already checked and, hence, it is not necessary to recheck them.

We have already seen that the CA map, $\mathcal{T}^5$, shifts the locations of the centroids of the tiles. And since the replicators' dynamics will be the same on the shifted tiling as they were on the original tiling (since the replicators themselves remain in $\mathbb{Z}^2$), we may use the work done to check axiom 5 to determine that axiom 6 holds as well.

6 MORE REPLICATOR EXAMPLES

In this section we present a variety of examples for which the constructions in section 5 apply. In each case, we present the replicator as a quadruplet, $(\Lambda, \mathcal{I}, \nu, \tau)$, along with its CA rule. Proposition 5.1, part 1, applies to all but the last two examples in this section. We leave it to the reader to verify that the other parts of axioms 5 and 6 also hold. This may be done with the assistance of WinCA which may be downloaded from Griffeath's site [7].

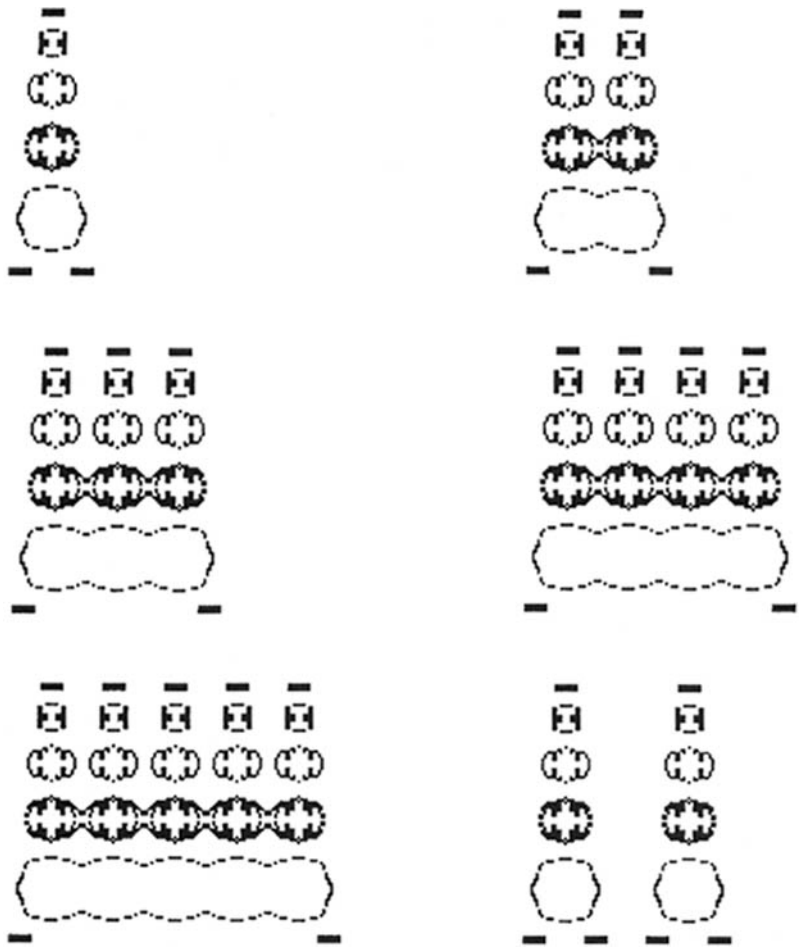

FIGURE 13 Space-time diagrams for the LtL rule $(5, 15, 18, 15, 26)$.

We begin with the famous *HighLife* replicator (which is a range 1, non-LtL rule that was discovered by Nathan Thompson in 1994). Each of the other examples is admitted by an LtL rule.

Example 6.1. *Bow tie pasta* $(\Lambda, \mathcal{I}, 12, 1)$, where Λ is the configuration in figure 14(a) and $\mathcal{I}$ is a rectangular tile oriented along the diagonal that comprises $\sigma_1 = 4$ and $\sigma_2 = 10$ sites (see fig. 14(b)). This replicator is admitted by HighLife, which is a variant of Life (see Bell [1]). As in the Game of Life, 0s become 1s next time if they see three 1s in their range 1 box neighborhood this time, and

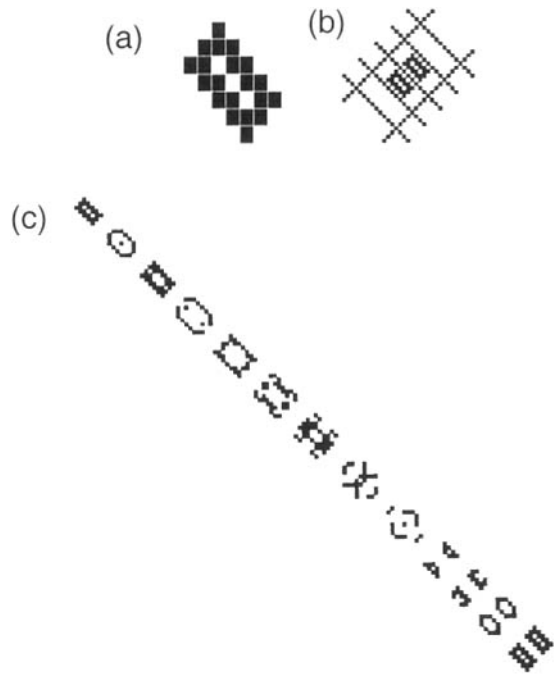

FIGURE 14 Bow tie pasta $(\Lambda, \mathcal{I}, 12, 1)$. (a) Lambda configuration. (b) A portion of the tiled space, with evolution along the diagonal. (c) Space-time diagram of replicator for HighLife rule. Time moves downward along the diagonal and times $0, 1, 2, \ldots, 12$ are depicted.

1s remain 1s if they see three or four 1s (including themselves) in their neighborhoods this time. However, in HighLife, a 0 can *also* become a 1 next time if it sees six 1s in its neighborhood this time, and, hence, HighLife is not an LtL rule. The additional possibility for birth yields the bow tie pasta, which is *not* admitted by Life.

As in example 2.4 the bow tie pasta's evolution occurs along the diagonal. This is depicted in figure 14(c).

Example 6.2. $(\Lambda, \mathcal{I}, 14, 1,)$ admitted by LtL rule $(2, 5, 6, 5, 7)$, where Λ is the configuration in figure 15(a), and $\mathcal{I}$ is a rectangle with parameters $\sigma_1 = 24$ and $\sigma_2 = 23$. Let us present 32 rows of the space-time diagram (fig. 15(b)) so that the reader may see the "bigger picture."

The next example is admitted by an LtL rule that is "Life-like" in the sense that when started from a random initial configuration, various local configura-

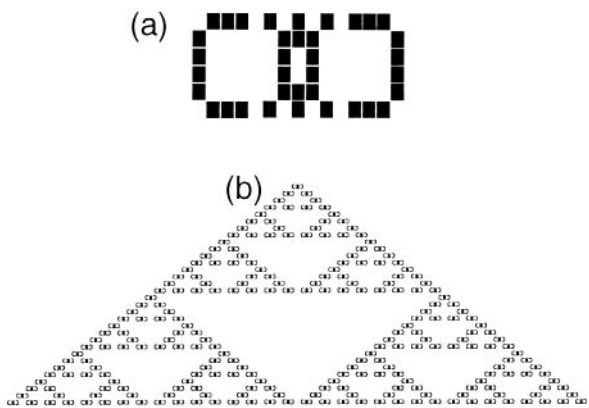

FIGURE 15 $(\Lambda, \mathcal{I}, 14, 1,)$ admitted by LtL rule $(2, 5, 6, 5, 7)$. (a) Lambda configuration. (b) Space-time diagram of replicator. Times $14k$, $k = 0, 1, 2, \ldots, 31$ are depicted.

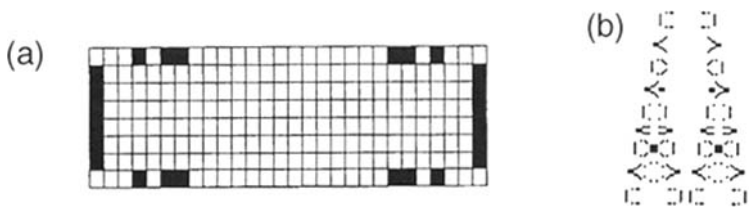

FIGURE 16 $(\Lambda, \mathcal{I}, 8, 1)$ admitted by LtL rule $(3, 6, 6, 6, 6)$. (a) Lambda configuration. (b) Space-time diagram. Times $0, 1, 2, \ldots, 8$ are depicted.

tions such as *bugs*, which are analogs to Life's *gliders*, and *bug makers*, analogs to Life's *glider guns*, emerge and are viable, while aperiodicity is not viable (see Evans [4, 5]). The replicator also emerges from a random initial state. There are no such replicators known for the Game-of-Life rule.

Example 6.3. $(\Lambda, \mathcal{I}, 8, 1)$ admitted by LtL rule $(3, 6, 6, 6, 6)$, where Λ is the configuration in figure 16(a), and $\mathcal{I}$ is a rectangle with parameters $\sigma_1 = 32$ and $\sigma_2 = 17$. Let us depict times 0–8 of the space-time diagram generated by Λ (see fig. 16(b)).

Example 6.4. $(\Lambda, \mathcal{I}, 14, 1)$ admitted by LtL rule $(4, 8, 9, 8, 12)$, where $\mathcal{I}$ is a rectangle with parameters $\sigma_1 = 68$ and $\sigma_2 = 51$. While Λ is too large to depict here,

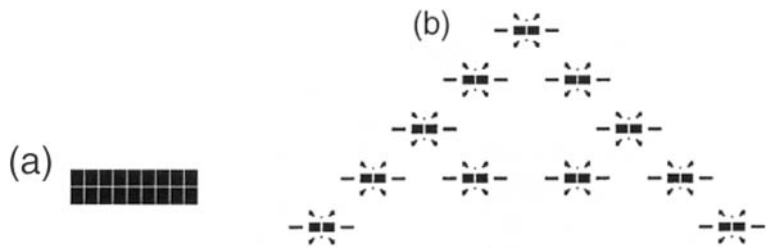

FIGURE 17 $(\Lambda, \mathcal{I}, 14, 1)$ admitted by LtL rule $(4, 8, 9, 8, 12)$. (a) Lambda configuration. (b) Space-time diagram. Times $0, 14, 28, 42$, and 56 are depicted.

it can be generated by running the rule for seven time steps starting from the seed in figure 17(a).

The next example illustrates that in the region of LtL space where the rule's parameters, β_1, β_2, δ_1, and δ_2, are near ρ, it is easy to construct new replicators from those that are known. This is also the region in which replicators seem to be most abundant.

Example 6.5. *Arrow Replicator* $(\Lambda, \mathcal{I}, 4, 1)$ admitted by LtL rule $(5, 3, 3, 3, 3)$, where Λ is the configuration in figure 18(a), and $\mathcal{I}$ is a rectangular tile oriented along the diagonal that comprises $\sigma_1 = 32$ and $\sigma_2 = 53$ sites. This range 5 replicator combines copies of that from example 2.4 (and generalized in proposition 7.1) to form a new replicator that also propagates along the diagonal axis (see fig. 18(b)).

The replicator in example 6.6 (see fig. 19) is generalized to other ranges in the next section (proposition 7.2).

Example 6.6. $(\Lambda, \mathcal{I}, 4, 1)$ admitted by LtL rule $(6, 12, 13, 12, 13)$, where Λ is the configuration in figure 19(a), and $\mathcal{I}$ is a square with parameters $\sigma_1 = \sigma_2 = 28$.

Example 6.7. $(\Lambda, \mathcal{I}, 22, 1)$ admitted by LtL rule $(7, 14, 14, 14, 14)$, where Λ is the configuration in figure 20, and $\mathcal{I}$ is a rectangle with parameters $\sigma_1 = 154$ and $\sigma_2 = 35$.

The next example is a range 7 *replicator-maker*. That is, it is a finite configuration that periodically generates a copy of the seed for the replicator from example 6.7.

Example 6.8. LtL rule $(7, 14, 14, 14, 14)$. This rule admits the previous example and the replicator-maker, depicted in figure 21 at time 0, that generates a copy of the replicator every 32 time steps.

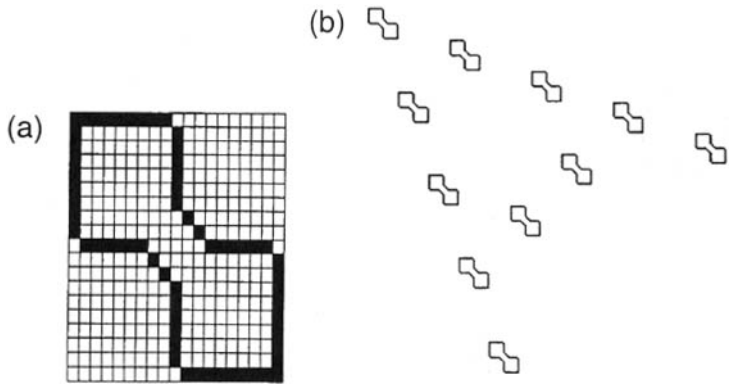

FIGURE 18 Arrow Replicator. $(\Lambda, \mathcal{I}, 4, 1)$ admitted by LtL rule $(5, 3, 3, 3, 3)$. (a) Lambda configuration. (b) Space-time diagram. Time moves downward along the diagonal and times $0, 4, 8, 12$, and 16, are depicted.

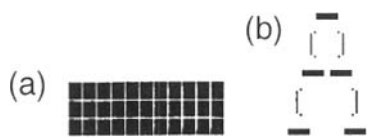

FIGURE 19 $(\Lambda, \mathcal{I}, 4, 1)$ admitted by LtL rule (6, 12, 13, 12, 13). (a) Lambda configuration. (b) Space-time diagram. Times $0, 1, 2, 3$, and 4 are depicted.

FIGURE 20 Lambda configuration for $(\Lambda, \mathcal{I}, 22, 1)$ admitted by LtL rule $(7, 14, 14, 14, 14)$.

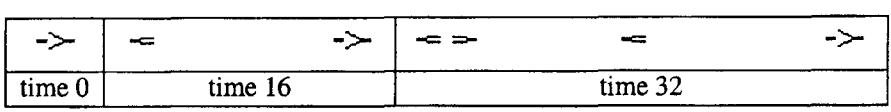

FIGURE 21 LtL rule $(7, 14, 14, 14, 14)$, at time 0.

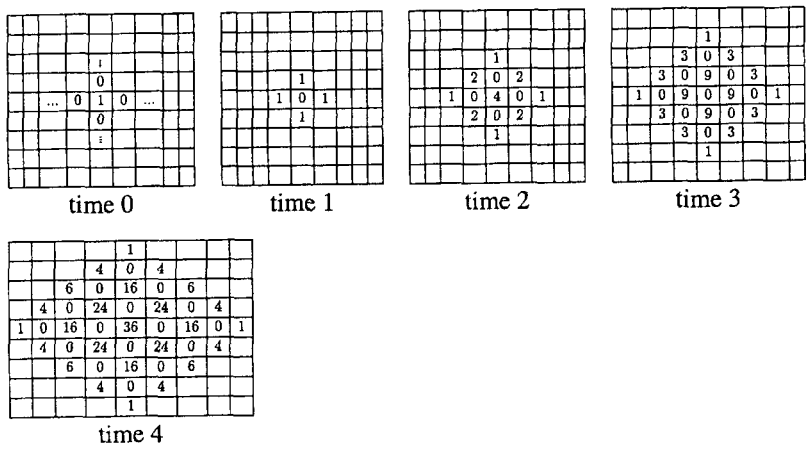

FIGURE 22 Space-time diagram of example 6.9, showing action of f.

The last LtL example is a two-dimensional replicator from range 3 that propagates along the diagonals. We present this example because the requisite Pascal-generating CA in axioms 5 and 6 has neighborhood $\mathcal{N} = \{\pm e_i : i = 1, 2\}$, even though the LtL rule has a box neighborhood. This is different from the other examples we have presented. Let us present that Pascal-generating CA before the LtL example.

Example 6.9. This is the Pascal-generating CA rule with $d = 2$ and $\mathcal{N} = \{\pm e_i : i = 1, 2\}$. The space-time diagram (fig. 22) of this CA is another three-dimensional generalization of Pascal's Triangle Mod 2. A few time steps of the action of f (empty sites have not yet seen any occupied sites in their neighborhoods) are depicted in figure 22 and the modular version is figure 23.

$\mathcal{I}^{\Lambda}_{(0,0)}$	$\stackrel{\phi}{\rightarrow}$	$(0,0)^1$
$\mathcal{T}^4 \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^{\Lambda}_{(-\frac{\sigma_1}{2},0)}, \mathcal{I}^{\Lambda}_{(\frac{\sigma_1}{2},0)}, \mathcal{I}^{\Lambda}_{(0,-\frac{\sigma_2}{2})}, \mathcal{I}^{\Lambda}_{(0,\frac{\sigma_2}{2})}\}$	$\stackrel{\phi_{1/2}}{\rightarrow}$	$\{(-\frac{1}{2},0)^1, (\frac{1}{2},0)^1, (0,-\frac{1}{2})^1, (0,\frac{1}{2})^1\}$
$\mathcal{T}^4 \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^{\Lambda}_{(-\sigma_1,0)}, \mathcal{I}^{\Lambda}_{(\sigma_1,0)}, \mathcal{I}^{\Lambda}_{(0,-\sigma_2)}, \mathcal{I}^{\Lambda}_{(0,\sigma_2)}\}$	$\stackrel{\phi}{\rightarrow}$	$\{(-1,0)^1, (1,0)^1, (0,-1)^1, (0,1)^1\}$

Example 6.10. $(\Lambda, \mathcal{I}, 4, 2)$ admitted by LtL rule $(3, 3, 3, 5, 5)$, where Λ is the configuration in figure 24(a). The cross sections of this replicator's three-dimensional space-time diagram are depicted in figure 24(b).

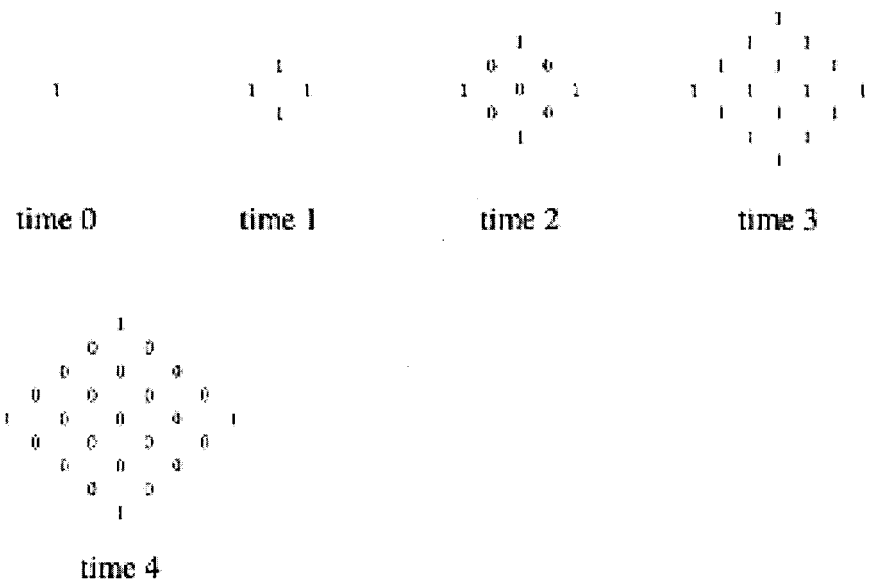

FIGURE 23 Constructed from figure 22 by ignoring the 0s and modding out by 2.

The tile for this replicator is a rhombus with dimensions $\sigma_1 = \sigma_2 = 16$ in the directions along which it propagates. Times $0, 4, 8, 12$, and 16 of the replicator in the tiled space are depicted in figure 25. As usual, every four time steps the replicas are contained inside a shift of the tiled space. For this CA and tiling depicted in figure 25, the centroids are in the set $(\sigma_1\mathbb{Z}\times\sigma_2\mathbb{Z})\cup((\sigma_1/2)\mathbb{Z}\times(\sigma_2/2)\mathbb{Z})$ at even times and in the shifted set, $((\sigma_1/2)\mathbb{Z} \times \sigma_2\mathbb{Z}) \cup (\sigma_1\mathbb{Z} \times (\sigma_2/2)\mathbb{Z})$ at odd times. Note that here the shifting occurs in each direction separately since that is how the Pascal-generating CA neighborhood is defined. ϕ and $\phi_{1/2}$ are defined as in section 5; however, proposition 5.1 does not apply. Instead, the diagram on page 146 applies.

We note that we have many more LtL replicator examples in a variety of ranges. These are best introduced in an animation fashion so we refer the reader to the collection of replicator experiments at Evans [6].

7 GENERALIZED REPLICATORS

In this section we show that for LtL rules with box neighborhoods, there exist one-dimensional replicators for all ranges $\rho \geq 3$ and two-dimensional replicators for ranges $\rho \geq 1$. We do this by generalizing several replicators in the following propositions.

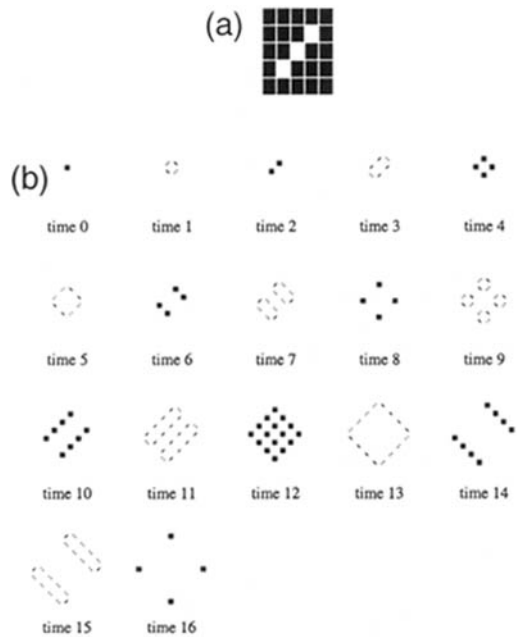

FIGURE 24 $(\Lambda, \mathcal{I}, 4, 2)$ admitted by LtL rule (3,3,3,5,5). (a) Lambda configuration. (b) Cross sections of the space-time diagram for this replicator.

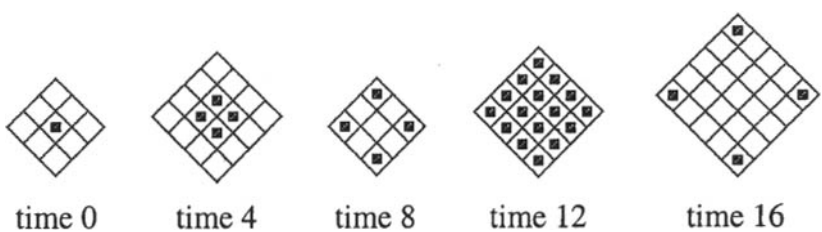

FIGURE 25 $(\Lambda, \mathcal{I}, 4, 2)$ depicted in the tiled space at times $0, 4, 8, 12, 16$.

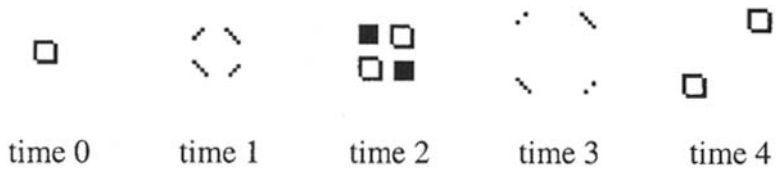

FIGURE 26 $(\Lambda, \mathcal{I}, \tau, 1)$ admitted by LtL rule $(\rho, 3, 3, 3, 3)$.

In the previous section we showed that there exists a one-dimensional replicator admitted by a range 2 LtL rule. We have not yet found a one-dimensional replicator admitted by a range 1 LtL rule (though we do most of our work in ranges larger than 1). We have, however, found many two-dimensional replicators admitted by range 1 LtL rules. We have also learned of many one-dimensional replicators admitted by range 1 non-LtL rules [3].

The range 5 version of the following generalization appears in example 2.4.

Proposition 7.1. For two-dimensional LtL rules with range ρ box neighborhoods such that $\rho = 3 + i$, $i = 0, 1, 2, \ldots, \Lambda$ consists of the sites on the boundary of a $2\rho \times 2\rho$ box with the two sites at the northeast and southwest corners deleted, tile $\mathcal{I}$ that comprises $\sigma_1 = 16 + 8i$, and $\sigma_2 = 24 + 9i$ sites along the diagonal, and $\tau = 4$, $(\Lambda, \mathcal{I}, \tau, 1)$ is a *one-dimensional replicator* with respect to the rule with parameters $\beta_1 = \beta_2 = \delta_1 = \delta_2 = 3$.

Proof. This family of replicators propagate along the diagonal as shown in figure 26. For each distinct range $\rho \in \{3 + i : i = 0, 1, 2, \ldots\}$, the spatial configurations of live sites are identical, except the blocks that appear at time 2 and the distances along the diagonal between the configurations of live sites vary as the range does. These variations are as follows.

The side length of the blocks that appear in the northwest and southeast corners at time 2 is $5 + 2i$.

The number of dead sites along the diagonals between the configurations of live sites is: $4 + 2i$ at time 0; $8 + 4i$ at time 1; $4 + 2i$ at time 2; $16 + 8i$ at time 3; and $12 + 6i$ at time 4. We use this information to construct the tile, $\mathcal{I}$, with $\sigma_1 = 16 + 8i$ representing the number of sites it comprises in direction 1 along the diagonal where propagation occurs and $\sigma_2 = 24 + 9i$ its analog in the mutually orthogonal direction 2. $\mathbb{Z}^2$ is tiled as illustrated in figure 27.

Since $\mathcal{I}$ is a rectangular tile, we may use the constructions from section 5 to yield the following.

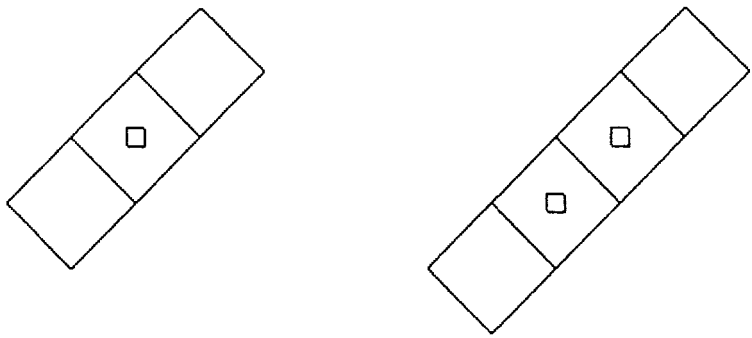

FIGURE 27 $\mathbb{Z}^2$ is tiled at times 0 and 4.

$\mathcal{I}^{\Lambda}_{(0,0)}$	$\overset{\pi\circ\phi}{\rightarrow}$	$(0)^1$
$\mathcal{T}^4 \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^{\Lambda}_{(-\sigma_1/2,-\sigma_1/2)}, \mathcal{I}^{\Lambda}_{(\sigma_1/2,\sigma_1/2)}\}$	$\overset{\pi_{1/2}\circ\phi_{1/2}}{\rightarrow}$	$\{(-1/2)^1, (1/2)^1\}$
$\mathcal{T}^4 \downarrow$		$\mathcal{F} \downarrow$
$\{\mathcal{I}^{\Lambda}_{(-\sigma_1,-\sigma_1)}, \mathcal{I}^{0}_{(0,0)}, \mathcal{I}^{\Lambda}_{(\sigma_1,\sigma_1)}\}$	$\overset{\pi\circ\phi}{\rightarrow}$	$\{(-1)^1, (0)^0, (1)^1\}$

This table shows that the first parts of axioms 5 and 6 hold; therefore, all that must be checked are properties 1 and 2 of theorem 4.3, where

$$k_1 = \left\lfloor \frac{2s_i\tau - (16+8i) + 2(3+i)}{(16+8i)} \right\rfloor = \left\lfloor \frac{14+2i}{(16+8i)} \right\rfloor = 0\,.$$

There are eight distinct strings of length 3. However, we already checked the case consisting of exactly one tile inscribed with Λ. The case consisting of all 0s is trivial and, since $k_1 = 0$ and two of the remaining four cases are equivalent mod translation, we need only check the two cases in figure 28, consisting of two and three adjacent tiles inscribed with Λ, respectively. (Distances between sites scale with the range as described above.) ■

The range 6 version of the following generalization appeared in example 6.6.

Proposition 7.2. For two-dimensional LtL rules with range ρ box neighborhoods such that $\rho \in \cup_{l=3}^{\infty}\{3l-3, 3l-2\}$, $\Lambda = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \le z_1 \le 2\rho - (l-1)$ and $0 \le z_2 \le 2\}$, rectangular tile $\mathcal{I}$ with dimensions $\sigma_1 = 2\rho + 2$ and $\sigma_2 = 4\rho + 4$,

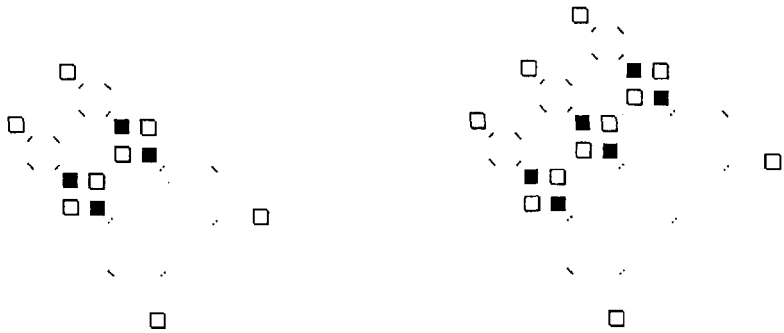

FIGURE 28 Space-time diagrams starting from 2 and 3 copies of Λ in adjacent tiles, respectively. Time moves downward along the diagonal and times $0, 1, 2, 3, 4$ are depicted.

and $\tau = 4$, $(\Lambda, \mathcal{I}, 2, 1,)$ is a *one-dimensional replicator* with respect to the rule with parameters $\beta_1 = \delta_1 = 2\rho$, $\beta_2 = 2\rho + 1 \leq \delta_2 \leq 3\rho + 2$.

Proof. *Case 1:* $\rho = 3l - 3$. Let us check that axioms 1–6 hold. We do this by illustrating the evolution of Λ under $\mathcal{T}$. In all of the illustrations in figures 29 and 30, Λ has height 3, width $5l - 4$, and the centroid of the evolution is the centroid of Λ.

Figure 29 shows that $\nu = 1$ and axiom 2 will be satisfied with either $\tau = 2$, or $\tau = 4$. Axioms 5 and 6 will yield $\tau = 4$ so let us use that. We may use the construction of the tile $\mathcal{I}$ and the maps ϕ and $\phi_{1/2}$ from section 5. Thus, $\sigma_1 = 12l - 8$ and $\sigma_2 = 12l - 10$. Since the speed of light in direction 1 is $\rho = 3l - 3$, theorem 4.3 gives

$$k_1 = \left\lfloor \frac{8(3l-3) - (12l-8) + 5l - 4}{12l - 8} \right\rfloor = 1 .$$

As discussed before, this gives us ten distinct cases to check, one of which we just did. Let us check the remaining cases. We begin with two copies of Λ separated by a tile painted with all 0s, illustrated in figure 30(a).

In figure 30(a), the numbers of live sites are as in the case with only one copy of Λ (depicted in gory detail in fig. 29). The number of dead sites between the copies at time 0 is $19l - 12$; at time 1, $17l - 12$; at time 2, $13l - 8$; at time 3, $11l - 8$; and at time 4, $7l - 4$. Thus, at time 4 all four copies are separated by the same number of dead sites, as required. Observe that the copies of Λ do not interact by time 4. Thus, as in example 2.1, we have only four cases left to check. These are illustrated in figure 30(b).

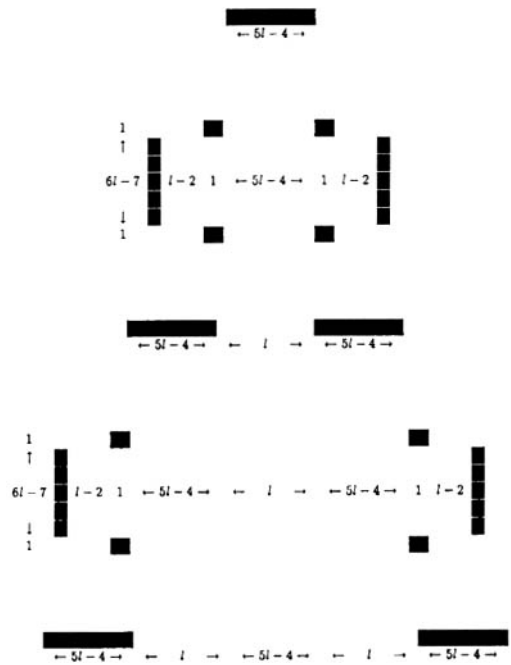

FIGURE 29 Space-time diagram illustrating in detail times $0, 1, 2, 3$, and 4, starting from Λ.

In order to describe the numbers of dead sites between the various configurations of live sites at each time step, we depict in detail (fig. 31) the space-time diagram starting from three adjacent tiles inscribed with copies of Λ. In that case, for axiom 5 to hold, two copies of Λ must appear at time 4 with two tiles inscribed with all 0s between them. Thus, there *should* be $31l - 20$ dead sites between the copies of Λ.
At time 4, we see that there are $31l - 20$ dead sites between the copies, as required. Figure 31 captures the scale of other space-time diagrams as well. Thus each of the required configurations satisfy axiom 5. The checking for axiom 6 is identical and, thus, by construction of the tile, $\mathcal{I}$, the replicator satisfies the axioms. ■

Case 2: $\rho = 3l - 2$. This example is just like the previous example, with a slight variation. We illustrate in figure 32 the variation in the first four time steps and leave it to the reader to fill in the remaining details.

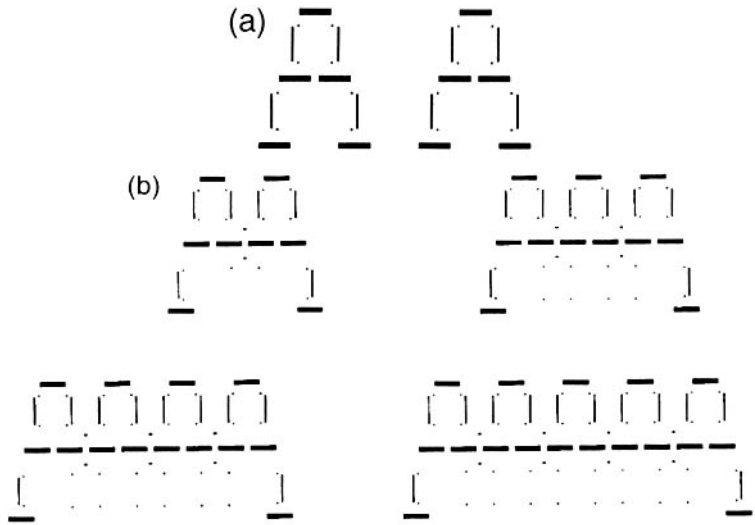

FIGURE 30 (a) Two copies of Λ separated by a tile painted with all 0s. (b) Space-time diagrams starting from $2, 3, 4$, and 5 copies of Λ in adjacent tiles.

Proposition 7.3. For two-dimensional LtL rules with range ρ box neighborhoods, $\Lambda = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 2\rho \text{ and } z_2 = 0\}$ square tile $\mathcal{I}$ with side length $2(\rho + k + 1)$ and $\tau = 2$, $(\Lambda, \mathcal{I}, \tau, 2)$ is a *two-dimensional replicator* with respect to the *exactly* θ rule $(\rho, \theta, \theta, \theta, \theta)$, for $\theta = \rho - k$, $k \in \{0, 1, 2, \ldots, \rho - 1\}$.

Proof. We leave it to the reader to check that axioms (1)–(6) from definition 4.4 hold.

Proposition 7.4. For two-dimensional LtL rules with range ρ box neighborhoods, $\Lambda = \{(z_1, z_2) \in \mathbb{Z}^2 : 0 \leq z_1 \leq 2\rho - ((\rho+1) \bmod 2) \text{ and } z_2 = 0\}$ $\mathcal{I}$ is a square with side length $\sigma_1 = \sigma_2 = [2 + 2(\rho \bmod 2)]\rho$, and $\tau = 2 + 2(\rho \bmod 2)$ $(\Lambda, \mathcal{I}, \tau, 2)$ is a *two-dimensional replicator* with respect to the exactly theta rule, $(\rho, \theta, \theta, \theta, \theta)$ for $\theta = \rho + 1$.

Proof. We leave it to the reader to check that axioms (1)–(6) from definition 4.4 hold.

There are many other generalizations of replicators as well; we provide only a few to illustrate that there are numerous examples.

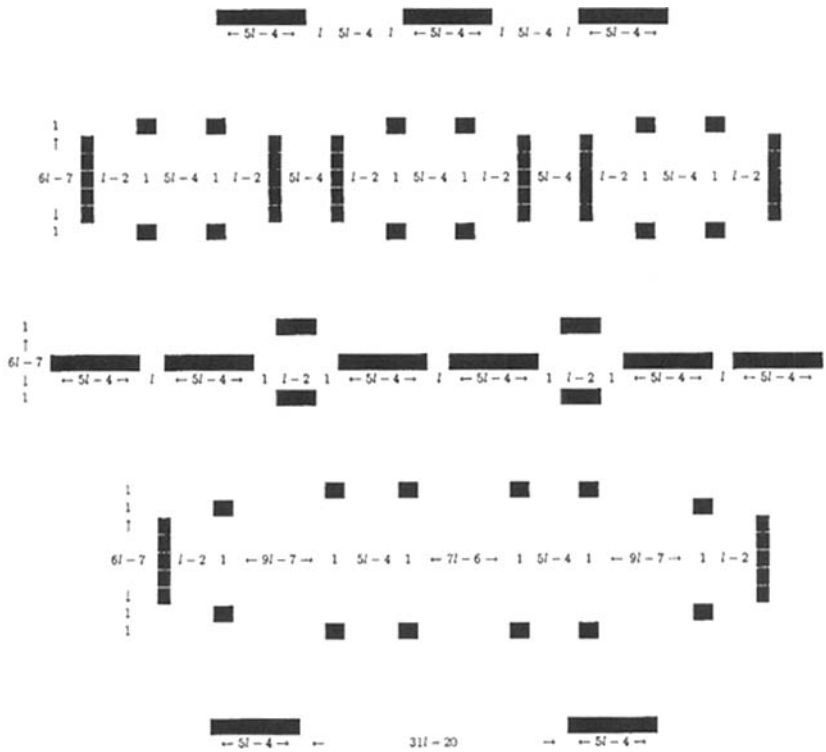

FIGURE 31 Space-time diagram illustrating in detail times $0, 1, 2, 3$, and 4, starting from three copies of Λ in adjacent tiles.

8 APPLICATIONS TO THE INFINITE SYSTEM

Up to now we have focused on the *local* aspects of replicators. Do replicators impact the *global* dynamics of the rules that admit them? We will show that, when restricted to particular initial states, the rules have well-understood stationary distributions. To do this, we use the ideas in Durrett [2, section 5d]. He does this for the case of the one-dimensional Pascal-generating CA with $\mathcal{N} = \{\pm 1\}$.

Let $\mathcal{T}$ be a d-dimensional CA rule and assume that it admits the replicator, $(\Lambda, \mathcal{I}, \tau, \nu)$. Let $\tilde{\mathcal{T}}^{\tau}$ be the restriction of $\mathcal{T}$ to Λ^{∞_ν}-strings (i.e., configurations that consist of infinitely many disjoint translates of $\mathcal{I}^{\Lambda}$ in each of the i mutually orthogonal directions ($i = 1, 2, \ldots, \nu$) with each translate inscribed with a copy of Λ or all 0s) at times $t\tau$, $t = 0, 1, 2, \ldots$.

Let η_θ = product measure with density θ. That is, at time 0 each tile is independently inscribed with a copy of Λ with probability θ, and all 0s otherwise.

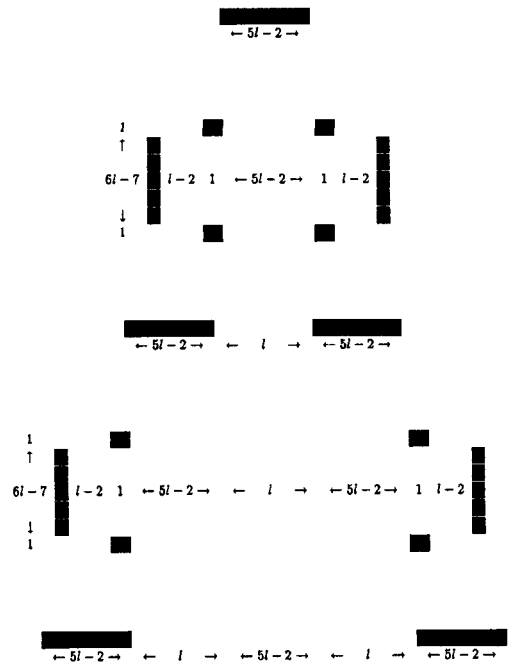

FIGURE 32 Space-time diagram illustrating in detail times 0, 1, 2, 3, and 4, starting from Λ.

Proposition 8.1. The measure $\eta_{1/2}$ is a stationary distribution for $\tilde{T}^{\tau}$. That is, starting from $\eta_{1/2}$, the $\mathcal{I}^{\Lambda}$'s in the image under $\tilde{T}^{\tau}$ are independent and identically distributed.

Proof. Let $\eta_{1/2}$ be the initial configuration. It suffices to show that the measure that gives equal mass to every configuration is invariant under $\tilde{T}^{\tau}$. In other words, it suffices to show that the mapping

$$\tilde{T}^{\tau} : \left\{\Lambda^{\infty_\nu}\text{–strings} \subset \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}\right\} \to \left\{\Lambda^{\infty_\nu}\text{–strings} \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)^d_{\mathcal{I}}}\right\}$$

is one to one and onto. And, since the infinite system is the limit of processes on strings of finite length, it suffices to show that the mapping

$$\tilde{T}^{\tau} : \left\{\Lambda^{(n+1)_\nu}\text{–strings} \subset \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}\right\} \to \left\{\Lambda^{(n)_\nu}\text{–strings} \subset \{0,\Lambda\}^{(\mathbb{Z}+1/2)^d_{\mathcal{I}}}\right\}$$

is $2^{[(n+1)^\nu - n^\nu]}$ to 1 and onto, where $n \in \mathbb{Z}^+$. This will yield the desired outcome since there are $2^{(n+1)^\nu}$ possible preimages and only 2^{n^ν} images.

Let

$$s \in \left\{ \Lambda^{(n)_\nu}\text{–strings} \subset \{0,\lambda\}^{(\mathbb{Z}+1/2)^d_I} \right\}$$

so that s is a string that has length n in each of the ν directions. Consider the possible preimages for s. The sites in ν of the hyperfaces (these are edges in two dimensions and faces in three dimensions) of the preimage can be whatever we like and then each of the remaining sites in the configuration can take on only one possible value in order for the image to be s. Thus, we need only count the number of possible configurations for the hyperfaces. There are $(n+1)^{\nu-1}$ sites in the first hyperface, $n(n+1)^{\nu-2}$ sites in the second (that have not already been counted), $n^2(n+1)^{\nu-3}$ in the third, and so on. Thus, the total number of sites in all of the relevant hyperfaces is

$$\sum_{k=0}^{\nu-1} n^{\nu-1-k}(n+1)^k.$$

Thus, s has $2^{\sum_{k=0}^{\nu-1} n^{\nu-1-k}(n+1)^k}$ possible preimages. Now let us show that

$$\sum_{k=0}^{\nu-1} n^{\nu-1-k}(n+1)^k = (n+1)^\nu - n^\nu.$$

First, use the Binomial Theorem to get

$$(n+1)^\nu - n^\nu = \sum_{k=0}^{\nu-1} \binom{\nu}{k} n^k.$$

Now we induct on ν so want to show that

$$\sum_{k=0}^{\nu} \binom{\nu+1}{k} n^k = \sum_{k=0}^{\nu} n^{\nu-k}(n+1)^k.$$

Since

$$\binom{\nu+1}{k} = \binom{\nu}{k-1} + \binom{\nu}{k},$$

then

$$\sum_{k=0}^{\nu} \binom{\nu+1}{k} n^k = \sum_{k=0}^{\nu-1} \binom{\nu}{k} n^{k+1} + \sum_{k=0}^{\nu} \binom{\nu}{k} n^k.$$

Using the inductive hypothesis and the Binomial Theorem, the above is equal to

$$\sum_{k=0}^{\nu-1} n^{\nu-k}(n+1)^k + (n+1)^\nu = \sum_{k=0}^{\nu} n^{\nu-k}(n+1)^k,$$

as desired. ■

Corollary 8.1. If a CA rule, $\mathcal{T}$, admits a ν-dimensional replicator, then it has a nontrivial invariant measure.

Proof. By definition, we have a tile, $\mathcal{I}$, and a finite configuration of 1s, Λ. Now let us construct an invariant measure. We work in the tiled space, $\{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ (see definition 4.1). With probability 1/2, place a copy of Λ inside each tile. Call the resulting tiling $\tilde{\mathcal{I}}$. There are $|\mathcal{I}|$ distinct shifts of $\tilde{\mathcal{I}}$. Form the average over all of the shifts,

$$\mu = \frac{1}{|\mathcal{I}|}\sum_{v\in\mathcal{I}} \theta^v(\tilde{\mathcal{I}})$$

where v is a vector in $\mathcal{I}$, and θ^v is the shift operator which translates the entire tiling by v. Then, by construction, μ is translation invariant. Also μ consists of independent copies of the Λ^{∞_ν}-strings described in proposition 8.1. Thus, the probability that any tile in μ contains a copy of Λ is 1/2. Therefore, μ is a nontrivial invariant measure for $\mathcal{T}$. ■

What happens starting from product measure with density θ if $\theta \neq 1/2$? The reader is referred to Dur [2, section 5d] for this, as well as other properties for the $\nu = 1$ case. As D. Lind (whose approach Durrett follows in section 5d) says, "Extensions to higher ranges should be clear" (see Lind [8, p. 36]).

APPENDIX

Let us prove that the maps ϕ and $\phi_{1/2}$ defined in section 5 are bijections.

The map ϕ takes $\{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ *onto* $\{0,1\}^{\mathbb{Z}^d}$ since, given $y = (n_1, n_2, \ldots, n_d)^m \in \{0,1\}^{\mathbb{Z}^d}$, $(m \in \{0,1\})$, there exists an $x = \mathcal{I}^l_{(n_1,n_2,\ldots,n_d)} \in \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ such that $\phi(x) = (n_1, n_2, \ldots, n_d)^{\phi(l)} = (n_1, n_2, \ldots, n_d)^m$ by taking

$$l = \phi^{-1}(m) = \begin{cases} \Lambda, & m = 1\,; \\ 0, & m = 0\,. \end{cases}$$

The proof that $\phi_{1/2}$ maps $\{0,\Lambda\}^{(\mathbb{Z}+1/2)^d_{\mathcal{I}}}$ *onto* $\{0,1\}^{(\mathbb{Z}+1/2)^d}$ is identical with ϕ replaced by $\phi_{1/2}$, $\mathbb{Z}^d_{\mathcal{I}}$ replaced by $(\mathbb{Z}+1/2)^d_{\mathcal{I}}$, and $\mathbb{Z}^d$ replaced by $(\mathbb{Z}+1/2)^d$.

In order to prove that ϕ is one to one, let $a, b \in \{0,\Lambda\}^{\mathbb{Z}^d_{\mathcal{I}}}$ with $a \neq b$. Then $a = \mathcal{I}^j_{(n_1,n_2,\ldots,n_d)}$ and $b = \mathcal{I}^l_{(m_1,m_2,\ldots,m_d)}$ for some $(n_1, n_2, \ldots, n_d), (m_1, m_2, \ldots, m_d) \in \mathbb{Z}^d$, $j, l \in \{0,\Lambda\}$. Since $a \neq b$, either $n_k \neq m_k$ for some $k \in \{1, 2, \ldots, d\}$ or $j \neq l$. Now $\phi(a) = (n_1, n_2, \ldots, n_d)^{\phi(j)}$ and $\phi(b) = (m_1, m_2, \ldots, m_d)^{\phi(l)}$ so in either case $\phi(a) \neq \phi(b)$, as desired.

The proof that $\phi_{1/2}$ is one to one, is identical with ϕ replaced by $\phi_{1/2}$, $\mathbb{Z}^d_{\mathcal{I}}$ replaced by $(\mathbb{Z}+1/2)^d_{\mathcal{I}}$, and $\mathbb{Z}^d$ replaced by $(\mathbb{Z}+1/2)^d$. ■

REFERENCES

[1] Bell, D. "HighLife—An Interesting Variant of Life," 1994. ⟨http://www.tip.net.au/~dbell/⟩.

[2] Durrett, R. *Lecture Notes on Particle Systems and Percolation.* Pacific Grove, CA: Wadsworth & Brooks Cole, 1988.

[3] Eppstein, D. Personal communication, 1999. ⟨http://www.ics.uci.edu/~eppstein/ca/replicators⟩.

[4] Evans, K. "Larger than Life: It's So Nonlinear." Ph.D. diss., University of Wisconsin-Madison, Ann Arbor, MI, 1996.

[5] Evans, K. "Larger-than-Life: Digital Creatures in a Family of Two-Dimensional Cellular Automata." *Discrete Math. & Theor. Comp. Sci.* (2001). ⟨http://dmtcs.loria.fr⟩.

[6] Evans, K. "Replicators and Larger-than-Life Examples—Experiments." 1999. ⟨http:/v/www.csun.edu/~kme52026/replicators.html⟩.

[7] Griffeath, D. "Primordial Soup Kitchen." ⟨http://psoup.math.wisc.edu/kitchen.html⟩.

[8] Lind, D. A. "Applications of Ergodic Theory and Sofic Systems to Cellular Automata." *Physica D* **10** (1984): 36–44.

[9] Wolfram, S. *Cellular Automata and Complexity.* Menlo Park, CA: Addison-Wesley, 1994.

This page intentionally left blank

Growth Phenomena in Cellular Automata

Janko Gravner

We illustrate growth phenomena in two-dimensional cellular automata (CA) by four case studies. The first CA, which we call *Obstacle Course*, describes the effect that obstacles have on such features of simple growth models as linear expansion and coherent asymptotic shape. Our next CA is random-walk-based *Internal Diffusion Limited Aggregation*, which spreads sublinearly, but with a shape which can be explicitly computed due to hydrodynamic effects. Then we propose a simple scheme for characterizing CA according to their growth properties, as indicated by two *Larger than Life* examples. Finally, a very simple case of *Spatial Prisoner's Dilemma* illustrates nucleation analysis of CA.

1 INTRODUCTION

In essence, analysis of growth models is an attempt to study properties of physical systems far from equilibrium (e.g., Meakin [34] and more than 1300 references cited in the latter). Cellular automata (CA) growth models, by virtue of their

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

simplicity and amenability to computer experimentation [25], have become particularly popular in the last 20 years, especially in physics research literature [40, 42]. Needless to say, precise mathematical results are hard to come by, and many basic questions remain completely open at the rigorous level. The purpose of this chapter, then, is to outline some successes of the mathematical approach and to identify some fundamental difficulties.

We will mainly address three themes which can be summarized by the terms: aggregation, nucleation, and constraint-expansion transition. These themes also provide opportunities to touch on the roles of randomness, monotonicity, and linearity in CA investigations. We choose to illustrate these issues by particular CA rules, with little attempt to formulate a general theory. Simplicity is often, and rightly, touted as an important selling point of cellular automata. We have, therefore, tried to choose the simplest models which, while being amenable to some mathematical analysis, raise a host of intriguing unanswered questions. The next few paragraphs outline subsequent sections of this chapter.

Aggregation models typically study properties of growth from a small initial seed. Arguably, the simplest dynamics are obtained by adding sites on the boundary in a uniform fashion. In fact, such examples were among the first studied. It soon became clear that they expand linearly in time and, properly rescaled, obtain a characteristic limiting shape. What if the space over which such growth spreads is not uniform, but instead contains a field of obstacles? Stationary obstacles do not complicate the analysis much, but the situation becomes murkier once one allows the obstacles to move. In fact, the literature appears to contain conflicting claims in the case of moving obstacles. Section 2 presents a detailed discussion of this class of models, including some rigorous results and conjectures.

Properties of asymptotic shape for linearly spreading growth can be notoriously hard to elucidate. By contrast, symmetric random walks progress through space more slowly (diffusing as the square root of time), and have an isotropic continuum space-time limit. For these reasons, growth models based on such walks often yield sublinear growth and circular asymptotic shape. One such example is presented in section 3.

Section 4 is more theoretical in nature. It proposes a general classification scheme which, simply put, provides a precise way to divide CA into those which grow and those which do not. This taxonomy may be viewed as a simple alternative to Langton's approach based on the frequency of transitions to nonquiescent states (the λ parameter, see Langton [31]). Especially for CA which depend on a parameter, it provides a strategy to search for complex rules on the boundary between qualitatively distinct regimes. Section 5 then provides two illustrative examples from a four-parameter rule space of general Life-like nonmonotone rules.

If the initial state is disordered, how do droplets which generate persistent growth emerge from random "soup" and with what frequency? Nucleation analysis addresses such questions. Nucleation effects can sometimes be very tricky to discern by computer, but we will show in section 6 how a mathematical analysis

with an essential experimental component aids in understanding self-organization of a simple competition model.

2 OBSTACLE COURSE

Before describing our first models, let us emphasize that in this section and the next, the neighborhood of a site consists of its nearest four points. To define the Obstacle Course (OC) CA, start by assuming that the state of every site in $\mathbf{Z}^2$ can be either 0 (empty), 1 (occupied), or 2 (an obstacle). In the simplest version of the OC rule, called *static OC*, 1s never change, a point in state 0 changes to 1 as soon as some neighbor is in state 1, and finally a site in state 2 with a neighboring 1 changes to 0 with a fixed probability $q \in [0,1]$. (In epidemics terms, 1s, 0s, and 2s could be interpreted as infected, and more and less susceptible individuals, respectively.) This rule is applied synchronously and independently at all sites in $\mathbf{Z}^2$ at every step of discrete time $t = 0, 1, 2, \ldots$. As for the initial state, we assume that the origin contains the only 1, while every other site is independently 2 with probability p and 0 with probability $1-p$. Our attention will focus on the set A_t of 1s at time t.

Say that $L_{p,q}$ is the *(linear) asymptotic shape* of A_t if

$$\frac{A_t}{t} \to L_{p,q} \tag{1}$$

as $t \to \infty$. It is easiest to define this convergence in terms of the *Hausdorff metric*. That is, define the ϵ-fattening of a set $B \subset \mathbf{R}^2$ to be $B^\epsilon = B + B_2(0,\epsilon) = \cup_{x \in B} B_2(x,\epsilon)$. Then say that eq. (1) holds if, for any $\epsilon > 0$,

$$L_{p,q} \subset \left(\frac{A_t}{t}\right)^\epsilon \quad \text{and} \quad \frac{A_t}{t} \subset L_{p,q}^\epsilon ,$$

for a large enough t.

The case $p = 0$ is simple: A_t is merely the diamond $\{(x,y) \in \mathbf{Z}^2 : |x| + |y| \le t\}$. Therefore, we can explicitly compute

$$L_{0,q} = \{(x,y) \in \mathbf{R}^2 : |x| + |y| \le 1\} .$$

Equally clearly, $L_{1,1} = L_{0,q}/2$.

Assume now that $p > 0$ and $q \in (0,1]$. This model fits into a general class of dynamics known as *first passage percolation (FPP)*. To explain the correspondence, we assign to every site $x \in \mathbf{Z}^2$ an independent random variable ξ_x with $P(\xi_x = 1) = 1 - p$ and $P(\xi_x = k) = p(1-q)^{k-1}q$ for $k = 2, 3, \ldots$. Assuming only x is initially occupied by a 1, the time $T_{x,y}$ when y becomes occupied is given by

$$\inf\{\sum_{i=0}^{n} \xi_{x_i} : n \ge 1 \text{ and } x = x_0, x_1, \ldots, x_n = y \text{ is a nearest-neighbor path}\} .$$

If ξ_x is interpreted as the time needed for x to become 1 after it has a neighboring 1, then a short induction gives $A_t = \{x : T_{0,x} \le t\}$. The next crucial observation is *subadditivity*: $T_{x,y} \le T_{x,z} + T_{z,y}$. A fair amount of mathematical theory and technical machinery [8, 29] then yields existence of a deterministic convex set $L_{p,q} \subset L_{0,q}$ with nonempty interior, such that $A_t/t \to L_{p,q}$ almost surely.

The $q = 0$ case is similar, but we need to allow for the possibility that the set A_∞, consisting of the origin and any sites with state 0 to which the origin is connected (by a nearest-neighbor path), is finite. This happens with probability 1 if $p \ge 1 - p_c \approx 0.407$ and otherwise with probability strictly less than 1. (Here, p_c is the critical density for site percolation in the plane.) Thus, $L_{p,0}$ is a random set if $p < 1 - p_c$: there exists a nontrivial deterministic convex set $L'_{p,0}$ such that $L_{p,0}$ equals $\{0\}$ on $\{|A_\infty| < \infty\}$ and $L'_{p,0}$ on $\{|A_\infty| = \infty\}$. The top left frame of figure 1 provides an example with $p = 0.3$. Sites in A_t are gray and obstacles black in all four frames.

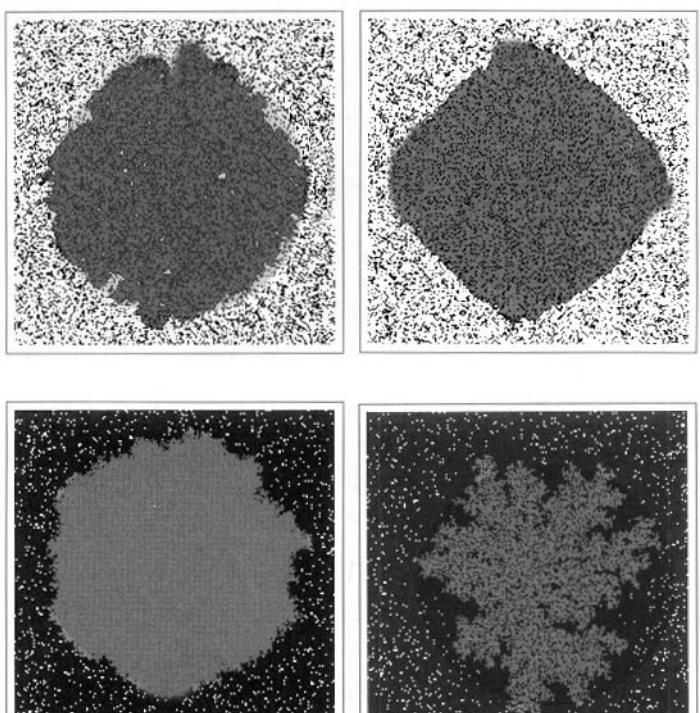

FIGURE 1 Growth in the static and moving OC dynamics.

Thus, the existence of $L_{p,q}$ is established, but what more can we say about these sets? It is possible to show that $L_{p,q} \to L_{0,q}$ as $p \to 0$, and that $L_{1,q} \to L_{1,1}$ as $q \to 1$ (using techniques from Durrett and Liggett [12]), but more detailed aspects of $L_{p,q}$ are not easy to discern. For instance, there are presently no rigorous methods available to show that $L_{p,q}$ is never a circle. More discussion on existence and properties of asymptotic shapes appears in Bohman and Gravner [5] and Gravner and Griffeath [17], while Gravner and McDonald [23] addresses a combination of bootstrap percolation [1] and OC rules.

A more complex CA called *moving OC* results if we allow the obstacles to diffuse. The easiest way to achieve this effect is to view 2s as particles which move freely on 0s and are forbidden to jump onto a 1. More precisely, the state of the CA consists of sites in state 1, sites in state 0, and sites containing one or more "2-particles." The following steps are then performed in succession:

1. Every 2-particle randomly and independently chooses a neighbor. If the chosen neighbor is not a 1, it jumps onto it. If the chosen neighbor is a 1, the jump is suppressed and the particle is killed (removed from the system) with probability q.
2. Every 0 with a neighboring 1 becomes a 1.

As before, start with a single 1 at the origin, surrounded by sites filled independently with a random number of particles from a distribution which is the same for all sites. We will also assume that this random number is bounded. Let p stand for the initial density of 2-particles, that is, the average number of 2s per site.

We should mention that, alternatively, one could restrict the number of 2s to at most 1. In this case, 2s would perform simple exclusion outside A_t, in which case synchronous dynamics would require a scheme such as Margolus neighborhood updating [40]. Phenomenologically, there should be little difference between our moving OC and the exclusion OC; this is easy to believe when p is small, while exclusion effects for $p \approx 1$ should correspond to those for $p \approx \infty$ in the version above. Furthermore, simple exclusion outside A_t ensures that 0s perform the same dynamics there as 2s, so the exclusion OC rule with $q = 0$ is a variant of *lattice gas DLA* [40, 41, 43]. One motivation for the agenda of this section is to take a step toward rigorous study of that still-mysterious rule. Additional details will appear in Gravner and Griffeath [21].

In what follows, we will say that a sequence A_t, $t = 0, 1, 2, \ldots$ of subsets of $\mathbf{Z}^2$ *expands linearly* if there exists an open ball $G \neq \emptyset$ and a $\rho_0 > 0$ such that, for every open ball $B \subset G$,

$$|A_t \cap (tB)| \geq \rho_0 t^2 \text{ area } (B) \text{ for all large enough } t\,; \tag{2}$$

hence, A_t/t covers G with density at least ρ_0. In practice, G is most often centered at the origin.

As with the static OC, let us start with the simpler case $q > 0$ and outline the proof that A_t expands linearly, almost surely. Linear expansion implies that the number of particles in A_t is on the order of the square of its diameter, eliminating the possibility that A_t is fractal (as extensive experimental evidence suggests to be the case in "ordinary" DLA, to which much of Meakin [34] is devoted). The top right frame of figure 1 is a snapshot of A_t with $q = 0.1$ and an initial set consisting of three 2-particles per site. In light of this picture, eq. (2) is no surprise. In fact, A_t/t seems to converge to a deterministic limit, but techniques for investigating this issue are completely lacking.

To prove eq. (2), attach to every 2-particle w a random variable ζ_w, which simply measures the number of times w attempts to jump onto A_t. Thus, for example, $\zeta_w = 1$ if the particle is killed on its first attempt. It is also clear that ζ_w are independent geometric random variables with mean $1/q$. Moreover, the position of w at any time t before extinction is, in ℓ^1-distance, at most ζ_w from where w would be if it moved freely. A standard computation with random walks then shows that the density (i.e., the expected number of 2-particles) is bounded uniformly in space and time. After a substantial extra argument, this property ultimately suffices to establish eq. (2). It is also worth noting that at every time t a particle w which has come in contact with A_t is either killed, in which case it contributes nothing to the total density, or else is still alive, in which case it contributes at most $2\zeta_w^2 + 2\zeta_w + 1$ to the density. Hence the density is always bounded by p if q is sufficiently close to 1.

What happens with eq. (2) in the most interesting case $q = 0$? For $p \in (0, 1)$, there could conceivably be three scenarios: either linear expansion persists for all values of p, or there is no linear expansion for any p, or there is a phase transition at some value of p. (Recall that this last is the state of affairs in the case of static OC.) The empirical literature on exclusion OC concurs that the second scenario is impossible, but there appear to be conflicting claims about linear expansion for high p [41, 43].

In fact, it seems natural to conjecture that, for very small p, the density of obstacles which remain active, in the sense that they are not captured within A_t, is uniformly bounded by p, and this property should imply linear expansion eq. (2). At present, a rigorous argument still appears elusive, so we simply illustrate the result by means of the bottom left frame of figure 1 (which has $p = 0.3$). If p is high, however, judging from computer simulations, the density of 2-particles at the boundary of A_t increases substantially above p, and the possibility that it increases without bounds cannot be eliminated. See Uwaha and Saito [41] for more discussion on this thorny issue, and the bottom right frame of figure 1, where $p = 3$, for an illustration.

3 INTERNAL DIFFUSION LIMITED AGGREGATION

The Internal Diffusion Limited Aggregation (IDLA) rule with continuous source, considered in this section, was introduced in Lawler et al. [32] where its basic asymptotic shape theory was developed. For a different perspective, see Moore and Machta [35] which contains an analysis of complexity properties of this rule. The synchronous version we present is specified by the *occupied set* A_t and the behavior of a collection of random walks. Initially, $A_0 = \{0\}$. At every time $t = 1, 2, \ldots$, each site in A_t contains one or more particles. To get A_{t+1}, together with a new particle configuration, execute (in succession) the following three steps:

1. One particle at each site is frozen, while the others execute one step of a symmetric nearest-neighbor random walk.
2. A_{t+1} is obtained by adjoining to A_t all sites which are being visited by a particle for the first time.
3. One particle is added at the origin.

To understand the behavior of this process, one approximates it by a *continuum-valued* CA on $\mathbf{Z}^2$. This CA, determined by $u_t : \mathbf{Z}^2 \to \mathbf{R}^+$, $t = 0, 1, \ldots$, is obtained by simply replacing the true particle configuration at time $t+1$ by the expected number of particles at every site. If we set $\lambda(u) = \max\{u - 1, 0\}$ and

$$\Delta_d(f)(x) = \sum_{y \in \partial\{x\}} (f(y) - f(x))$$

(where $\partial\{x\}$ is the set of nearest neighbors of x), we obtain

$$\begin{aligned} u_{t+1}(x) &= \max\{u_t(x), 1\} + \frac{1}{4} \sum_{y \in \partial\{x\}} \lambda(u_t(y)) + \mathbf{1}_{\{0\}} \\ &= u_t(x) + \frac{1}{4}\Delta_d(\lambda(u))(x) + \mathbf{1}_{\{0\}} \,. \end{aligned} \tag{3}$$

To get the *diffusion scaling* limit, one defines, for $x' \in \mathbf{R}^2$ and $t' \geq 0$, $u'(x', t') = u(\epsilon^{-1}x', \epsilon^{-2}t')$, writes eq. (3) in terms of the new variables t', x', u', then divides it by ϵ^2, and computes the limit as $\epsilon \to 0$ by (formal) Taylor expansion to obtain (omitting primes)

$$\frac{\partial u}{\partial t} = \frac{1}{4}\Delta\lambda(u) + \delta_0 \,. \tag{4}$$

In either eqs. (3) or (4), the occupied set is given simply by $\{u \geq 1\}$.

Although not obvious at first glance, it turns out that eq. (3) is equivalent (in the proper weak interpretation), to the famous *Stefan problem*, a model for ice melting in the presence of heat sources. It is also true that eq. (3) has an

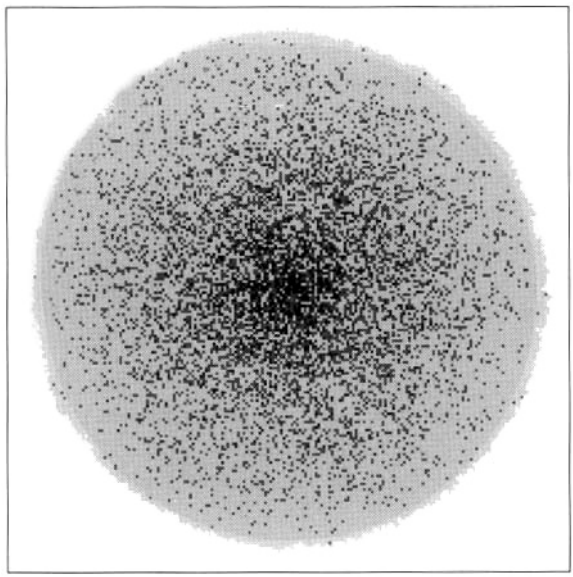

FIGURE 2 A snapshot of IDLA at a large time.

explicit unique solution given in polar coordinates by $\lambda(u) = v(rt^{-1/2})$, where

$$v(s) = \int_s^K \frac{2}{\pi\sigma} e^{-\sigma^2}\, d\sigma\,, \tag{5}$$

$$e^{-K^2} = \pi K^2\,. \tag{6}$$

Hence, $A_t/\sqrt{t}$ converges to a circle with radius $K \approx 0.498$, and the proportion of nonfrozen particles is $1 - \pi K^2 \approx 0.22$. Figure 2 gives a snapshot of an IDLA simulation at $t \approx 38,000$ on a 200×200 array. For many more details, and to see how the above heuristic can be turned into a proof for the very similar asynchronous version of this model, see Gravner and Quastel [27].

Assume now that one adds $c(t)$ particles at the origin at time t, where $c(t)$ is no longer 1, but an increasing function of t. An interesting question is how quickly c must increase for the shape of the occupied set to no longer be circular. Since the normal approximation for binomial probability $\binom{n}{k}2^{-n}$ holds up to $k = o(n)$, one would expect that the set A_t needs to expand linearly. Furthermore, during the time interval $[0, t]$, a random walk started at 0 will visit sites of distance order t from the origin with exponentially small probability. We, therefore, expect that $c(t)$ must increase exponentially fast to initiate the transition away from circular

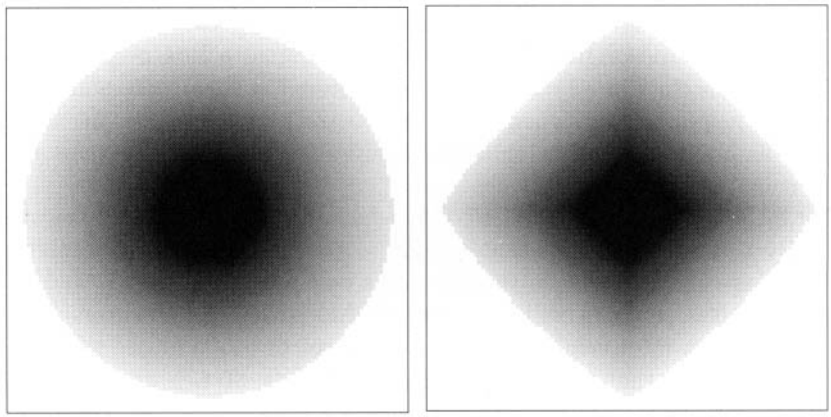

FIGURE 3 The continuum-valued CA eq. (5) with $\gamma = 0.1$ and $\gamma = 5$.

shape. More precisely, let us assume that $c(t) = e^{\gamma t}$, and write

$$L_\gamma = \lim_{t\to\infty} \frac{A_t}{t}.$$

Then we conjecture that L_γ exists almost surely, and approaches a circular shape as $\gamma \to 0$, while as $\gamma \to \infty$ it approaches the unit diamond $\{(x, y) : |x| + |y| \le 1\}$. In general, the shapes L_γ should be determined by large deviation rates; this model is, therefore, similar in spirit to branching random walks [3].

An exponentially increasing number of particles makes verification of the above conjecture by direct simulation of the particle system prohibitively slow. On the other hand, the continuum-valued CA eq. (3) simply becomes

$$u_{t+1}(x) = u_t(x) + \frac{1}{4}\Delta_d(\lambda(u))(x) + c(t)\mathbf{1}_{\{0\}}. \tag{7}$$

The two frames in figure 3 show the resulting growths for $\gamma = 0.1$ and $\gamma = 5$, both stopped when they reached a radius of about 50. (Gray shading is logarithmic to make the density profile visible.) A more general discussion on applications of continuum-valued CA may be found in Rucker [37].

4 GROWTH PROPERTIES OF CA: A GENERAL FRAMEWORK

The setup we now introduce is essentially the same as in Gravner and Griffeath [18]. Let us start by describing the neighborhoods we most often consider. The neighborhood for the origin will be a finite set denoted by $\mathcal{N}$, its translation

$x + \mathcal{N}$ then being the neighborhood of the point x. By convention, we assume that $\mathcal{N}$ contains the origin. Most typical is the range ρ Box neighborhood, in which case $\mathcal{N}$ is the $(2\rho+1) \times (2\rho+1)$ box centered at the origin, and the range ρ Diamond neighborhood, when $\mathcal{N} = \{(x, y) \in \mathbf{Z}^2 : |x| + |y| \leq \rho\}$. In particular, range 1 Diamond and Box neighborhoods are also known as von Neumann and Moore neighborhoods, respectively.

Let ξ_t be a general probabilistic CA, which, for simplicity, only has states 0 and 1. By this we mean first that $\xi_t : \mathbf{Z}^2 \to \{0, 1\}$ describes the configuration at time t; as usual, 1s will be thought of as occupied sites, and ξ_t and the set of occupied sites $\{\xi_t = 1\}$ will be identified. Moreover, the synchronous transition rule is given by a neighborhood $\mathcal{N}$ and a set of probabilities $\pi(S) \in [0, 1]$, $S \subset \mathcal{N}$, specifying that $\xi_{t+1}(x) = 1$ with probability $\pi((\xi_t - x) \cap \mathcal{N})$ independently at every time t and every spatial location x. Such CA rules are called *monotone* (or *attractive*) if $S_1 \subset S_2$ implies $\pi(S_1) \leq \pi(S_2)$. Deterministic CA, of course, have $\pi(S)$ only 0 or 1. Finally, in *solidification* CA every set S which contains 0 has $\pi(S) = 1$.

Assume that a two-state CA fixes the all 0s state, that is, $\pi(\emptyset) = 0$. Let B_n be the $(2n + 1) \times (2n + 1)$ box around the origin, and construct initial state ξ_0 by filling B_n with density $1/2$ product measure, and B_n^c with 0s.

We will call a CA *expansive* if $P(\xi_t$ expands linearly$) \to 1$ as $n \to \infty$ (recall the definition of linear expansion from section 2). On the other hand, we say that a CA is *constrained* if there exist positive constants c_1 and c_2 so that $P(B_{c_1 n}$ ever includes an occupied site$) \leq \exp(-c_2 n)$. Finally, we classify as *equivocal* those CA which do not fit into either previous category. (Various subcategories of equivocal may also be of interest, e.g., CA in which the number of occupied sites is likely to grow without limit, those in which linear spread occurs along a subsequence of times, and so on.)

The main motivation for these definitions comes from *oriented percolation*, in which $\mathcal{N}$ is, say, the von Neumann neighborhood, and the monotone rule declares that $\pi(S) = p > 0$ as soon as $S \neq \emptyset$. Then there exists a critical probability $p_c \in (0, 1)$ such that ξ_t is expansive in the supercritical regime (that is, when $p > p_c$). On the other hand, $P(\xi_t = \emptyset$ for some $t) = 1$ for every n as soon as $p \leq p_c$, and the subcritical ($p < p_c$) regime leads to a constrained dynamics, while the critical ($p = p_c$) oriented percolation is equivocal [2, 10].

Assume now that a CA rule is monotone and deterministic, and, for the sake of simplicity, that $\pi(S)$ does not change if S is reflected around either coordinate axis. It can then be proved that the model is expansive if and only if it enlarges every half-plane $H_u = \{x \in \mathbf{Z}^2 : \langle x, u\rangle \leq 0\}$ ($u \in \mathbf{R}^2$ is an arbitrary unit vector):

$$\xi_0 = H_u \Rightarrow \xi_0 \subset \xi_1 \text{ and } \xi_0 \neq \xi_1 .$$

One direction is easy: if this last condition is violated, then the CA is constrained, so, in fact, there are no equivocal CA in this class. (For much more on monotone CA growth, see Gravner and Griffeath [15, 17, 19, 22].)

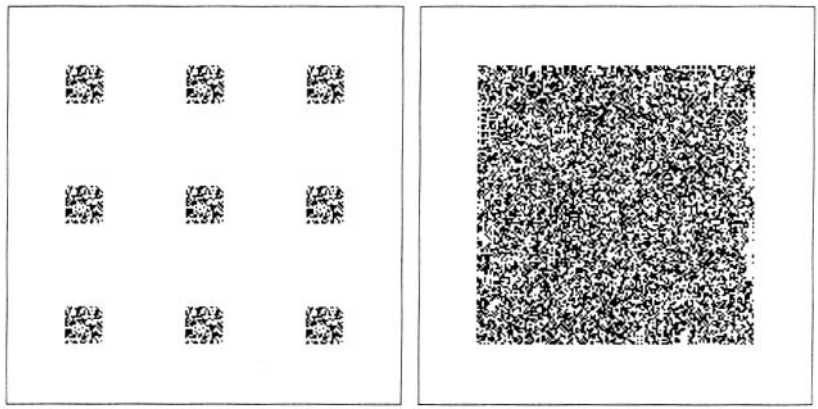

FIGURE 4 A linear CA and its random perturbation.

Outside the realm of monotonicity, there are very few available rigorous techniques [17], but it is worth mentioning a few. First, it is not hard to prove that any *linear* deterministic CA, where $\pi(S) = |S| \bmod 2$, is equivocal: it grows, but also repeatedly collapses to a set of $|\mathcal{N}| \cdot |\xi_0|$ points at exponentially spaced times. (For a discussion of the replication properties of such rules, see the July 15–21, 1996, recipe at Griffeath [25].) This example also illustrates how equivocal dynamics are typically fragile—by analogy with the one-dimensional case [7], one expects the random perturbation $\pi(S) = p \cdot (|S| \bmod 2)$ to be expansive for $p < 1$, as is strongly suggested by simulations. Figure 4 provides a range 1 Box example with $n = 20$ at time $t = 64$ with $p = 1$ and $p = 0.999$. By contrast, expansive dynamics typically seem to be robust with respect to small changes in p.

Some CA growth models can be analyzed by finding an embedded one-dimensional linear rule [20]. One such case is Exactly 1 solidification with von Neumann neighborhood $\mathcal{N}$ and $\pi(S) = 1_{|S|=1}$ if $0 \notin S$. To see how this works, fix a site x at distance exactly $t+1$ from the initial seed. (This distance is measured in "light speed," in this case via the ℓ^1, metric.) The state of x at time $t+1$ is obtained by an XOR of the states at time t of its two neighbors at distance t. While not immediate, it is possible to use this property to prove that this system is expansive, and, in fact, show that the final density of occupied sites is $2/3$ starting from any finite initial seed.

Finally, the "edge of the light cone" technique described above, in conjunction with analysis of one-dimensional random CA from Bramson and Neuhauser [7], shows that, for example, random Exactly 1 solidification, where $\pi(S) = p \cdot 1_{|S|=1}$ if $0 \neq S$, is expansive when $p < 1$ and $\mathcal{N}$ is either the von Neumann or Moore neighborhood.

For the vast majority of CA, however, one must resort to computer simulation to get an indication of their growth properties. Deterministic Moore neighborhood Exactly 1 solidification, for instance, is evidently expansive, although no rigorous argument for this is known at present.

In any case, one must be very cautious about conclusions from simulation, since it is always possible that a very large n (corresponding to a very large initial random set) is necessary before limiting behavior as $n \to \infty$ kicks in. For example, it is easy to convince oneself that Life is equivocal, as gliders are likely to emerge from initial soup, while the rest apparently settles into a periodic state. In fact, it seems more likely that Life is expansive, since a large box will likely contain space-filling structures on its boundary. But none of these structures is known to have appropriate self-defense properties against destabilizing influences from outside. Despite the spectacular advances in understanding the mechanisms of Life's growth, as described elsewhere in this volume, prospects for proving its expansiveness still seem quite remote.

Problems with simulation notwithstanding, one can use the computer to look for interesting CA rules on the border between "metastably" expansive and constrained cellular automata. If a CA depends on a parameter, and is apparently expansive for one value but constrained for another, then the values near the transition offer prospects not only for equivocal dynamics, but also for signature properties of complex dynamics. For example, one-dimensional objects such as *gliders*, *bugs*, and *ladders* (see next section for the meaning of last two terms) typically cannot persist in "robust" expansive rules, as they "explode" into growth in all directions. (See Bohman and Gravner [5] and Bohman [4] for some rigorous results in this direction.)

5 LARGER THAN LIFE

This rule was introduced in Griffeath [24] and studied in Evans [13]. Assume that the neighborhood $\mathcal{N}$ is a range ρ box. The deterministic Larger than Life (LtL) rule is given by

$$\pi(S) = 1_{0 \notin S, \beta_1 \le |S| \le \beta_2} + 1_{0 \in S, \delta_1 \le |S| \le \delta_2}.$$

In words, birth of a 1 occurs at a site if the number of occupied neighbors is between β_1 and β_2, while for survival of a 1 this number must be between δ_1 and δ_2. For example, Life is given by $\rho = 1$ and $(\beta_1, \beta_2, \delta_1, \delta_2) = (3, 3, 3, 4)$.

Of particular interest is the *threshold-range* regime, when ρ is large and $(\beta_1, \beta_2, \delta_1, \delta_2) = \rho^2 \cdot (\tilde{\beta}_1, \tilde{\beta}_2, \tilde{\delta}_1, \tilde{\delta}_2)$. Two reasons for significance of this regime are outlined below.

Assume that space is scaled by $1/\rho$. As $\rho \to \infty$, the LtL rule converges to an analogous Euclidean rule in which cardinalities are replaced by areas. This leads to limiting geometry of various objects of interest, such as *bugs*, which are finite

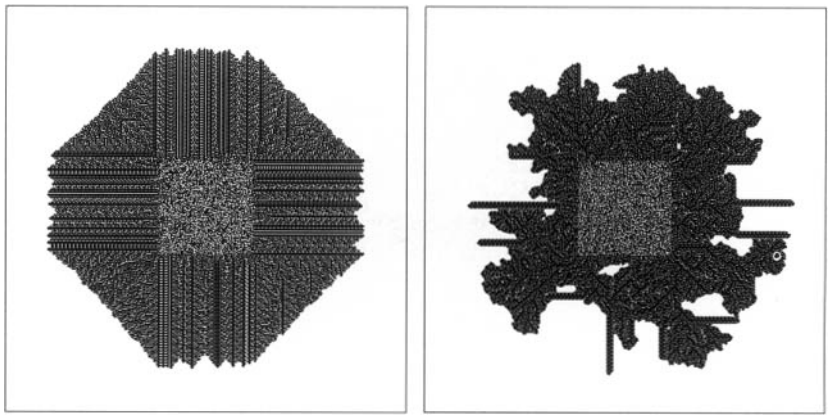

FIGURE 5 Exactly θ CA with $\theta = 2$ and $\theta = 3$.

sets with the property that the dynamics exactly translates them in finitely many steps—large-range versions of gliders, in short. Moreover, the boundary between expansive and constrained dynamics appears to form a three-dimensional subset of the four-dimensional parameter space $\{\Sigma = (\tilde{\beta}_1, \tilde{\beta}_2, \tilde{\delta}_1, \tilde{\delta}_2)\} \subset [0,4]^4$. In other words, if a one-parameter subfamily $\Sigma_{\tilde{\alpha}}$ experiences a transition between expansive and constrained dynamics, then it is very likely (unless $\Sigma_{\tilde{\alpha}}$ happens to move on the critical surface) to experience a *sharp* transition: for some $\tilde{\alpha}_c$, the dynamics is expansive if $\tilde{\alpha} < \tilde{\alpha}_c$ and ρ is large enough, and constrained if $\tilde{\alpha} > \tilde{\alpha}_c$ and ρ is large enough.

The study of phase transition in deterministic CA is generally hampered by the fact that the rule space is inherently discrete. But in models such as LtL with intrinsic threshold-range scaling, there is a natural way to introduce continuously varying rules. Excitable media modeling provides other examples [11, 14].

An example in which sharp transition can be proved is monotone LtL, that is, for $\beta_2 = \delta_2 = (2\rho + 1)^2$ and $\delta_1 - 1 \leq \beta_1$. These CA are often referred to as monotone *Biased Voter Automata (BVA)*. In the threshold-range regime, expansive BVA dynamics are characterized by $\tilde{\beta}_1 < 2$, and constrained by $\tilde{\beta}_1 > 2$ [15].

Typically, one does not need to go all the way to the limit $\rho = \infty$ to experience interesting phenomena near critical points [11, 13]. Fairly small neighborhoods may already contain some ingredient of critical behavior. For example, a variety of interesting scaling phenomena occur in monotone BVA rules with $\delta_1 = 0$ near $\tilde{\beta}_1 = 2$ [19]. Two nonmonotone examples are given below. In the simulations of figures 5 and 6 we have chosen $n = 100$, on a 400×400 array.

The first example is range 1 Exactly θ solidification: $\rho = 1$ and $\pi(S) = 1_{|S|=\theta}$ if $0 \notin S$. Note that this rule is obviously constrained for $\theta = 4$; in fact, nothing

FIGURE 6 LtL with $\delta_2 = 22, 23$, and 24.

outside the initial box B_n gets occupied. On the other hand, overwhelming experimental evidence (e.g., see fig. 5) suggests that the $\theta = 2$ rule is expansive. The $\theta = 3$ case, called *Life without Death*, also seems supercritical, but close enough to critical for existence of *ladders*. These objects, which grow but are restricted to a strip, actually appear quite frequently within the chaotic spread. Papers by Griffeath and Moore [26] and Gravner and Griffeath [17] analyze this rule in some detail, proving its **P**-completeness [26], and showing that its growth is sensitive to small perturbations of the initial seed [17].

Our second example is from Evans [13]. Consider range 3 LtL with parameters $(14, 19, 14, \delta_2)$. This rule seems constrained for $\delta_2 \leq 22$ (the periodic state in figure 6 was achieved by $t = 40$) and expansive for $\delta_2 \geq 24$ (although the growth at $\delta_2 = 24$ is slow, the state in figure 6 was achieved at $t = 300$). The intermediate case $\delta_2 = 23$ is not so easy to decipher; in fact, it gives rise to a rich menagerie of bugs, and is otherwise remarkably similar to Life. (Note also that the proportions of neighborhood size, $[14, 19, 14, 23]/49$ and $[3, 3, 3, 4]/9$, are not too far apart, suggesting an interesting threshold-range critical point nearby.) The snapshot in figure 6 was taken at $t = 1000$, by which time the dynamics have neither settled into a periodic state, nor conquered much space.

In light of this last example, it is an interesting open question whether one might devise a general scheme to design gliders guns and other fundamental building blocks of universal computation for large-range LtL in some parameter regime.

6 SPATIAL PRISONER'S DILEMMA

Prisoner's Dilemma is a game in which either player chooses strategy 1 (cooperation) or 0 (defection), and the player with strategy i receives payoff a_{ij} when playing against a player who chooses strategy j.

	1 (coop.)	0 (def.)
1 (coop.)	a_{11}	a_{10}
0 (def.)	a_{01}	a_{00}

The basic assumption $a_{01} > a_{11} > a_{00} > a_{10}$ leads to a well-known paradox: the defection strategy is clearly the optimal choice for either player, but making this choice leaves them with lower payoffs than mutual cooperation. This has led to a large number of papers investigating strategies in tournaments with repeated rounds between players (see Grim [28] and other papers in the same volume of *BioSystems*); in this case, one usually makes the additional assumption that $a_{10} + a_{01} < 2a_{11}$ to make the cooperating strategy better than out-of-phase flip-flopping by two players.

As an alternative approach, Nowak and May [6, 36] investigated self-organizing properties of the *spatial* version of the game above. A later paper with a point of view somewhat similar to ours is Lindgren and Nordahl [33]. A version of the Nowak-May Spatial Prisoner's Dilemma (SPD) rule is as follows. Start with a configuration of 0s and 1s on $\mathbf{Z}^2$. The player at each site x plays against every player in its neighborhood $x + \mathcal{N}$ (excluding itself). After all payoffs are computed, the player at each site x changes its state to the state associated with the largest total payoff in $x+\mathcal{N}$. Moreover, with *mutation* probability p the player adopts a random state. Under indefinite iteration of this rule, let $\eta_t : \mathbf{Z}^2 \to \{0, 1\}$ be the state of the system at time t.

Without loss of generality, we will assume that $a_{10} = 0$ and $a_{00} = 1$. Therefore, the SPD parameters are $\mathcal{N}$, p, and the two remaining payoffs.

Clearly, the most interesting issue is how regions of substantial cooperation may emerge from a sea of defectors in this rule. Let us start with the observation that if $p = 0$, then all 0s is a fixed state, and then investigate what happens under a small p perturbation.

To reiterate, we will assume $\eta_0 \equiv 0$ from now on. Also, for simplicity, let us assume, unless specified otherwise, that $\mathcal{N}$ is the von Neumann neighborhood. This makes SPD a CA with range 2 Diamond neighborhood, although writing out the associated π by hand would take some time.

If $a_{01} + 3 > 3a_{11}$, then every 1 with a neighboring 0 changes into 0 with probability $1 - p$, so the set of 0s compares favorably to supercritical oriented percolation CA (see section 4). Thus,

$$\lim_{p \to 0} \limsup_{t} P(\eta_t(0) = 1) = 0\,, \tag{8}$$

and there is a very low level of cooperation.

In fact, simulation suggests that eq. (8) persists when $a_{01} + 3 > 2a_{11}$, but the situation changes when $a_{01} + 3 < 2a_{11}$. In the language of section 4, when $p = 0$ the SPD is constrained in the former case, and expansive in the latter case. According to the general (though unproved) principle that expansiveness is ro-

bust under small random permutations, we conjecture that cooperation emerges when $a_{01} + 3 < 2a_{11}$, to the extent that

$$\rho_e = \lim_{p \to 0} \liminf_t P(\eta_t(0) = 1) > 0 . \tag{9}$$

The right frame in figure 7 gives a snapshot of these dynamics at $p = 0.01$ at the time substantial cooperation (black) has started to emerge. In our simulations we have chosen $a_{01} = 4.2$ and $a_{11} = 4$.

As an interesting aside, figure 8 provides a plot of the equilibrium density $\lim_{t \to \infty} P(\eta_t(0) = 1)$ (assuming it exists) versus p in our case of SPD (diamonds) and the range 1 Box case with $a_{01} = 10$ and $a_{11} = 7.5$ (pluses). To estimate the densities, we made every initial state contain a 30×30 square of 1s (surrounded by 0s) to speed up emergence of cooperation. Simulations were run on squares of various sizes and up to various times, depending on the speed of convergence. As a result, we estimate $\rho_e \approx 0.62$. Moreover, as in many artificial life models (and also, presumably, in the real-life counterparts), a high level of mutation makes coherent self-organization impossible, so beyond a critical $p \approx 0.14$ the equilibrium density is driven purely by noise. Note also that the Box neighborhood example suggests a second-order phase transition exactly at the minimal density, which is rather mysterious and merits further study.

In view of eq. (9), it is natural to ask how long it takes for the cooperating region to reach a typical point if p is small. This is the statistic which measures *nucleation* of the SPD. To be more precise, call T the first time the box $B_{p^{-1/3}}$ (of $(2p^{-1/3} + 1)^2$ sites) around the origin contains $2\rho_e p^{-2/3}$ 1s. The choice of $p^{-1/3}$ reflects the trivial lower bound on the order of T obtained by assuming

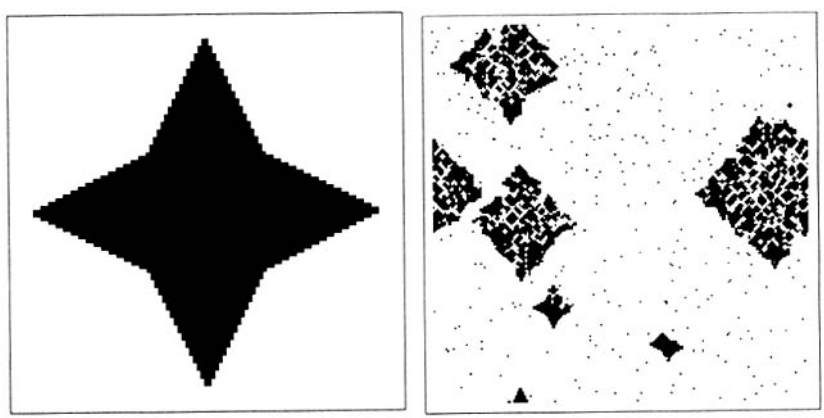

FIGURE 7 SPD with $p = 0$ and with $p = 0.01$.

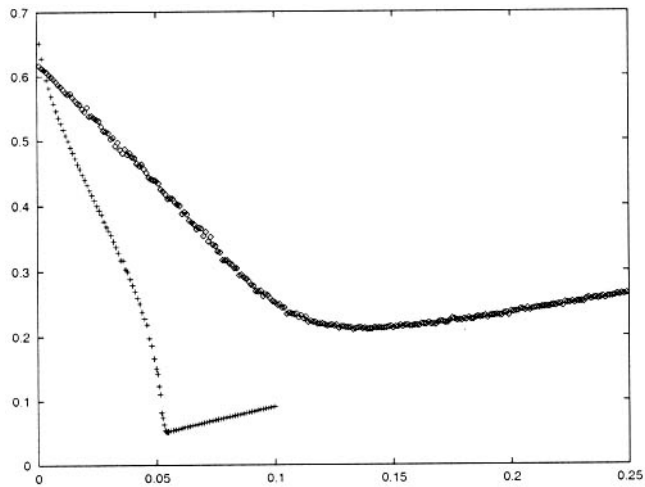

FIGURE 8 Equilibrium density vs. p in two SPD models.

that a single 1 generates growth which spreads with the speed of light (cf. the derivation of eq. (10) to see how this would yield $p^{-1/3}$).

Nucleation questions are connected to studying the smallest seeds which grow, as those are likely to be the first which affect the origin. Again, assume for a moment that $p = 0$. Then a single 1 dies, and so does any pair of 1s. Three 1s may form blinkers:

$$\begin{array}{ccc} & & \\ 1\ 1 & \longleftrightarrow & 1\ 1 \\ 1 & & \end{array} \quad \text{or} \quad \begin{array}{ccc} & 1 & \\ 1 & & \\ & 1 & \end{array} \longleftrightarrow 1\ 1\ 1 \ .$$

A 2×2 square of 1s expands linearly (the left frame in figure 7 is a snapshot of growth from this initial state), and this property apparently persists for small enough $p > 0$. (In fact, positive p seem to make the resulting shape convex.)

To estimate the order of T for small p, we now need to make a few estimates. In the discussion of the next paragraph, all times and probabilities are to be interpreted within the order of the quantity given.

Imagine space-time as embedded in $\mathbf{R}^3$, with the xy plane containing $\mathbf{Z}^2$ and time being the positive z-axis. Form a cone $\mathcal{C}_t$ with apex at $(0, 0, t)$, height t, and circular base of radius t. Then cooperation is likely to reach the origin at time t if $\mathcal{C}_t$ contains at least one 2×2 square of 1s. At any fixed time, such a square appears by itself with probability p^4. However, the first type of blinker above appears with a much higher probability p^3 and creates a square by time $1/p$ with a probability which is bounded below (by 0.2, say). Since there are t^3

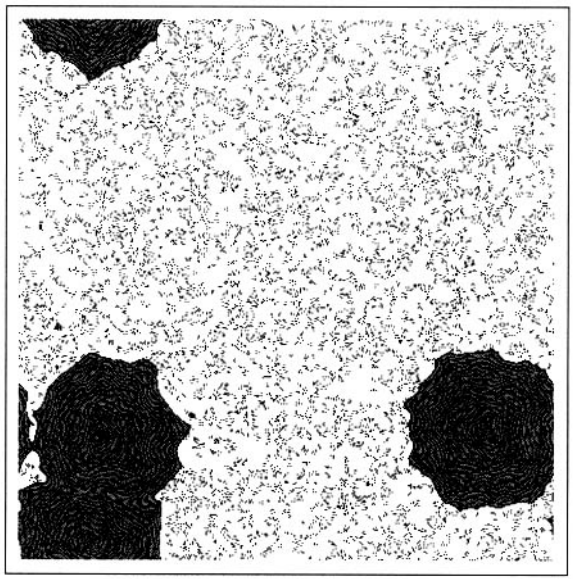

FIGURE 9 Nucleation in BVA.

sites in $\mathcal{C}_t$, T must satisfy

$$T^3 \cdot p^3 \approx 1 . \tag{10}$$

We can therefore, with some confidence, conjecture that T is on the order of p^{-1} as $p \to 0$.

Quite formidable obstacles would need to be overcome before our last conjecture could be proved. In fact, there are no rigorous results whatsoever on nucleation in nonmonotone dynamics. On the other hand, quite a lot is known about this aspect of monotone CA [1, 9, 15, 16, 19, 39]. As one illustration, consider the BVA (introduced in section 5) and assume for simplicity that randomness is confined to initial states with density p of 1s. In such cases, a rather general nucleation theory is possible, leading sometimes to power laws in p, and sometimes to exponential metastability. To explain the latter, we consider a specific example with range 2 Box neighborhood and $\beta_1 = 11$, $\delta_1 = 3$. In this case, T can be simply the first time the origin becomes occupied, although the definition given above also works. Then, it can be proved [22, 38] that every site eventually becomes permanently occupied. Thus the limiting equilibrium density ρ_e of eq. (7) equals 1. On the other hand, there exist constants C_1 and C_2 such that

$$P(C_1 p^{-3} \leq \log T \leq C_2 p^{-3}) \to 1$$

as $p \to 0$, so it takes a long while to be occupied when p is small. Simulations with very low p are therefore not feasible. Nevertheless, figure 9 depicts a 400×400

system with $p = 0.13$ at an intermediate time $t = 117$; occupied sites at different times are periodically shaded to give a basic impression of nucleation and growth in this CA.

ACKNOWLEDGMENTS

I wish to express my gratitude to David Griffeath and other organizers of the CA 1998 workshop, as well as the Santa Fe Institute, for the opportunity to participate in this wonderful event. In addition, I thank David Griffeath and Jeremy Quastel for help during the preparation of my talk and this chapter. The research presented here was partially supported by grant J1-8542-0101-97 from Slovenia's Ministry of Science and Technology and NSF grant DMS-9703923.

REFERENCES

[1] Aizenman, M., and J. Lebowitz. "Metastability Effects in Bootstrap Percolation." *J. Phys. A: Math. Gen.* **21** (1988): 3801–3813.

[2] Bezuidenhout, C., and G. Grimmett. "The Critical Contact Process Dies Out." *Ann. Prob.* **18** (1990): 1462–1482.

[3] Biggins, J. D. "The Asymptotic Shape of the Branching Random Walk." *Adv. Appl. Prob.* **10** (1978): 62–84.

[4] Bohman, T. "Discrete Threshold Growth Dynamics are Omnivorous for Box Neighborhoods." *Trans. Am. Math. Soc.* **351** (1999): 947–983.

[5] Bohman, T., and J. Gravner. "Random Threshold Growth Dynamics." *Random Structures and Algorithms* **15** (1999): 93–111.

[6] Bonhoeffer, S., M. A. Nowak, and R. M. May. "More Spatial Games." *Intl. J. Bif. & Chaos* **4** (1994): 33–56.

[7] Bramson, M., and C. Neuhauser. "Survival of One-Dimensional Cellular Automata under Random Perturbations." *Ann. Prob.* **22** (1994): 244–263.

[8] Cox, J. T., and R. Durrett. "Some Limit Theorems for Percolation Processes with Necessary and Sufficient Conditions." *Ann. Prob.* **9** (1981): 583–603.

[9] Dehghanpour, P., and R. H. Schonmann. "Metropolis Dynamics Relaxation via Nucleation and Growth." *Comm. Math. Phys.* **188** (1997): 89–119.

[10] Durrett, R. *Lecture Notes on Particle Systems and Percolation.* Pacific Grove, CA: Wadsworth & Brooks/Cole, 1988.

[11] Durrett, R., and D. Griffeath. "Asymptotic Behavior of Excitable Cellular Automata." *Exper. Math.* **2** (1993): 183–208.

[12] Durrett, R., and T. M. Liggett. "The Shape of the Limit Set in Richardson's Growth Model." *Ann. Prob.* **9** (1981): 186–193.

[13] Evans, K. "Larger than Life: It's so Nonlinear." Ph.D. Thesis, University of Wisconsin, Madison, WI, 1996.

[14] Fisch, R., J. Gravner, and D. Griffeath. "Threshold-Range Scaling for the Excitable Cellular Automata." *Stat. & Comp.* **1** (1991): 23–39.
[15] Gravner, J., and D. Griffeath. "First Passage Times for Discrete Threshold Growth Dynamics." *Ann. Prob.* **24** (1996): 1752–1778.
[16] Gravner, J., and D. Griffeath. "Nucleation Parameters in Discrete Threshold Growth Dynamics." *Exper. Math.* **6** (1997): 207–220.
[17] Gravner, J., and D. Griffeath. "Cellular Automaton Growth on $\mathbf{Z}^2$: Theorems, Examples, and Problems." *Adv. App. Math.* **21** (1998): 241–304.
[18] Gravner, J., and D. Griffeath. "Reverse Shapes in First-Passage Percolation and Related Growth Models." In *Perplexing Problems in Probability*, edited by M. Bramson and R. Durrett, 121–142. Boston, MA: Birkhäuser, 1999.
[19] Gravner, J., and D. Griffeath. "Scaling Laws for a Class of Critical Cellular Automaton Growth Rules." *Random Walks Workshop Proceedings*, Erdös Center, Budapest. (1999): to appear.
[20] Gravner, J., and D. Griffeath. "Asymptotics for von Koch Type Solidification." In preparation.
[21] Gravner, J., and D. Griffeath. "CA Growth in a Field of Obstacles." In preparation.
[22] Gravner, J., and D. Griffeath. "Two-Dimensional Monotone Cellular Automata: Growth, Regularity, and Scaling Laws." In preparation.
[23] Gravner, J., and E. McDonald. "Bootstrap Percolation in a Polluted Environment." *J. Stat. Phys.* **87** (1997): 915–927.
[24] Griffeath, D. "Self-Organization of Random Cellular Automata: Four Snapshots." In *Probability and Phase Transition*, edited by G. Grimmett, 49–67. Dordrecht, Netherlands; Boston, MA: Kluwer, 1994.
[25] Griffeath, D. "Primordial Soup Kitchen." ⟨`http://psoup.math.wisc.edu`⟩.
[26] Griffeath, D., and C. Moore. "Life without Death is **P**-Complete." *Complex Systems* **10** (1996): 437–447.
[27] Gravner, J., and J. Quastel. "Internal DLA and the Stefan Problem." *Ann. Prob.* (1999): submitted.
[28] Grim, P. "Spatialization and Greater Generosity in the Stochastic Prisoner's Dilemma." *BioSystems* **37** (1996): 3–17.
[29] Kesten, H. "On the Speed of Convergence in First-Passage Percolation." *Ann. Appl. Prob.* **3** (1993): 296–338.
[30] Krug, J., and H. Spohn. "Kinetic Roughening of Growing Surfaces." In *Solids Far From Equilibrium*, edited by C. Godrèche, 479–582. Cambridge, MA: Cambridge University Press, 1992.
[31] Langton, C. G. "Life at the Edge of Chaos." In *Artificial Life II*, edited by C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, 41–91. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. VIII. Reading, MA: Addison-Wesley, 1992.
[32] Lawler, G. F., M. Bramson, and D. Griffeath. "Internal Diffusion Limited Aggregation." *Ann. Prob.* **20** (1992): 2117–2140.

[33] Lindgren, K., and M. G. Nordahl. "Evolutionary Dynamics of Spatial Games." *Physica D* **75** (1994): 292–309.
[34] Meakin, P. *Fractals, Scaling and Growth Far From Equilibrium.* Cambridge, MA: Cambridge University Press, 1998.
[35] Moore, C., and J. Machta. "Internal Diffusion-Limited Aggregation: Parallel Algorithms and Complexity." *J. Stat. Phys.* (1999): submitted.
[36] Nowak, M. A., and R. M. May. "The Spatial Dilemmas of Evolution." *Intl. J. Bifur. & Chaos* **3** (1993): 35–78.
[37] Rucker, R. "Continuous-Valued Cellular Automata in Two Dimensions." This volume.
[38] Schonmann, R. H. "Finite-Size Scaling Behavior of a Biased Majority-Rule Cellular Automaton." *Physica A* **167** (1990): 619–627.
[39] Schonmann, R. H. "On the Behavior of Some Cellular Automata Related to Bootstrap Percolation." *Ann. Prob.* **20** (1992): 174–193.
[40] Toffoli, T., and N. Margolus. *Cellular Automata Machines.* Boston, MA: MIT Press, 1987.
[41] Uwaha, M., and Y. Saito. "Aggregation Growth in a Gas of Finite Density: Velocity Selection via Fractal Dimension of Diffusion-Limited Aggregation." *Phys. Rev. A* **40** (1989): 4716–4723.
[42] Vichniac, G. Y. "Simulating Physics with Cellular Automata." *Physica D* **10** (1984): 96–116.
[43] Voss, R. P. "Multiparticle Fractal Aggregation." *J. Stat. Phys.* **36** (1984): 861–872.

This page intentionally left blank

Constructive Molecular Dynamics Lattice Gases: Three-Dimensional Molecular Self-Assembly

Martin Nilsson
Steen Rasmussen
Bernd Mayer
David Whitten

Realistic molecular dynamics and self-assembly is represented in a lattice simulation where water, water-hydrocarbons, and water-amphiphilic systems are investigated. The details of the phase separation dynamics and the constructive self-assembly dynamics are discussed and compared to the corresponding experimental systems. The method used to represent the different molecular types can easily be expended to include additional molecules and thus allow the assembly of more complex structures. This molecular dynamics (MD) lattice gas fills a modeling gap between traditional MD and lattice gas methods. Both molecular objects and force fields are represented by propagating information particles and all microscopic interactions are reversible.

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

1 INTRODUCTION

1.1 CONSTRUCTIVE DYNAMICAL SYSTEMS

Living systems, perhaps the ultimate constructive dynamical systems, is the motivation for this work and our focus is a study of the dynamics of molecular self-assembly and self-organization.

In living systems, matter is organized such that it spontaneously constructs intricate functionalities at all levels from the molecules up to the organism and beyond. At the lower levels of description, chemical reactions, molecular self-assembly and self-organization are the drivers of this complexity.

We shall, in this chapter, demonstrate how molecular self-assembly and self-organization processes can be represented in formal systems. The formal systems are to be defined as a special kind of lattice gas and they are in a form where an obvious correspondence exists between the observables in the lattice gases and the experimentally observed properties in the molecular self-assembly systems. This has the clear advantage that by using these formal systems, theory, simulation, and experiment can be conducted in concert and can mutually support each other. However, a disadvantage also exists because analytical results are difficult to obtain for these formal systems due to their inherent complexity dictated by their necessary realism.

The key to novelty in molecular systems is their ability to generate aggregates that carry functionalities that cannot be observed at the level of the objects that make up these aggregates. As aggregates aggregate to yet higher-order aggregates, perhaps including simpler molecules (from lower levels), dynamical hierarchies are formed [2, 3]. Dynamical hierarchies are characterized by distinct observable functionalities at multiple levels of description. Since these higher-order structures are generated spontaneously due to the physico-chemical properties of their building blocks, complexity can come for free in molecular self-assembly systems. Through such processes, matter apparently can program itself into structures that constitute living systems [11, 27, 30]. Once a self-sustaining, self-reproducing molecular aggregate has been assembled, evolution can engage, which is the other powerful, natural complexity-creating mechanism. Note the order: first self-assembly, then evolution.

1.2 MOLECULAR SELF-ASSEMBLY AND DYNAMICAL HIERARCHIES

Molecular self-assembly is characterized by organizing a large amount of (heterogeneous) chemical entities and the driving forces of this process are solely defined by minimizing Gibbs free energy ΔG. However, the macroscopic structure is defined by the molecular interaction or recognition between single molecules (objects) on the microscopic scale. Local interactions generate mesoscale structures with unique functionalities.

Among the most prominent examples of molecular self-assembly is the formation of biological membranes composed of lipids characterized by their dual

(amphiphilic) nature [32], but a variety of recently described, manmade self-assembly products have opened new directions in the field of biomolecular materials design [15, 37]. Examples include the formation of cyclic peptide tubes [13] as well as of artificial membranes composed of synthetic polymers [1]. The emergent functionalities carried by the structure are as important as structural aspects of such assemblies. A beautiful example of such emergent functionalities is described by Luisi et al. [5, 6, 23]: Membraneous surfaces define a unique interface to the environment (water) which may lead to novel chemical reactivity. Luisi's group performed a sequence of studies on the increased hydrolysis rate of oleic acid anhydride at the interface of oleic acid membranes. In this particular example the emergent function (hydrolysis) of the higher-order aggregate (membrane) leads to the formation of the constituents of the membrane itself, which ultimately results in the autocatalytic self-reproduction of the catalytic interface.

This example clearly illustrates constructive molecular dynamics spanning various levels of complexity: molecular objects (first level); water, and hydrophilic and hydrophobic monomers, which may partly polymerize, e.g., into an amphiphilic string (second level); then self-assembly into aggregates, e.g., micelles or vesicles (third level) which are able to self-reproduce. At each of these levels we can observe emergent properties as pair distribution within water, between water and monomers (level 1), elasticity of the polymer (level 2), and inside/outside, permeability, or reactive surface at the aggregate level (level 3). The aggregate may even, as described above, self-reproduce which ultimately gives a third level, self-sustaining dynamical hierarchy in a chemical system [19].

It should be stressed that the functional properties of the basic, first-order objects, both the real water molecules and monomers as well as the formal models of these—which we shall define a little later—do not change during the process. It is only the *context* within which these objects are arranged that changes. Thus, the operational semantics of the information, the forces each object receives from its environment, is *context sensitive.* For example, the accessible states for a hydrophobic monomer in bulk polar phase (water) are distinctively different from the respective states in a nonpolar phase and again different from the states of a hydrophobic monomer in an amphiphilic polymer [17, 29]. This fact defines a downward causality as the higher-order structures modulate or restrict the dynamics of the lower-order structures which comprise them. This phenomenon of observed downward causality in dynamical hierarchies is related to the "slaving principle" as suggested by Haken [14].

There are two significant reasons why it is not trivial to generate such a dynamical hierarchy in a model of physico-chemical systems (or any other system for that matter): (i) It involves multilevel dynamics—that is simultaneous dynamics on many times—and length scales, which requires large computational resources. (ii) Also, the natural, conceptual framework for such a system seems to be a set of interacting objects and not a closed form model, such as a differential equation system. However, to form the higher-order structures from the bottom

up becomes very simple once the systems are composed of interacting objects. For example, starting with objects that are models of monomers it is trivial that the monomer objects can form polymers as they are combined into a string. Now the polymers can form membranes as they are aggregated in a particular fashion, and so forth. It turns out that a formulation of a dynamical hierarchy can be made conceptually as well as computationally simple if the interacting objects are defined on a lattice. Furthermore, other groups have obtained promising results by modeling macroscopic effects in chemical systems based on a fine-grained system representation using lattice-type simulation methods [8, 9, 24] and hybrid methods [16].

1.3 CELLULAR AUTOMATA AND LATTICE GASES

Lattice gases are a particular kind of cellular automata which allow particles to propagate on the lattice in a natural manner. The original lattice gas automata (LGA) [12, 38] are nice examples of how it is possible to simulate macroscopic effects (fluid flow) based on microscopic rules of interaction between the lattice particles. For a more general introduction to lattice gases, see for example Boghosian [7] and Doolen [10]. Our molecular dynamics (MD) lattice gases [17, 18, 28] are a natural extension of the classical lattice gases, and the basic idea behind this discrete field automaton is (i) to model both matter and fields as mediating information particles, as well as (ii) to allow particles to polymerize and the polymers (and monomers) to aggregate. The description of the physics within these formal systems is the topic of the next section.

2 PHYSICS

2.1 REPRESENTATION

The fundamental concept in our representation is that information particles represent both matter and interaction (force) fields. The presence of a particular information particle at a lattice point defines a particular kind of molecule currently present at this site. Propagated to neighboring lattice sites, information particles also represent a specific electromagnetic field from this particular molecule. Every type of interaction is associated with a separate field. Each molecular potential is thus decomposed into a set of repelling and attracting particles of varying value depending on molecular interaction type and position. At any lattice point the resulting field influences the residing molecule and determines where it moves in the future. If no molecule is present at a lattice site, obviously no force fields will be propagated from that site. However, fields will reach nonoccupied sites as well as being influenced (partially shielded) by the presence of molecules.

Molecules have excluded volumes. Only one molecule can reside at each site at any given time. The molecules' orientation determines the structure of the associated field. Thus rotations resulting in lower potential energy may occur as

neighboring molecules interact. Bonds between monomers in polymers may also be viewed in this picture: Bond information (particles) needs to be propagated between the monomers to insure that a polymer does maintain its configuration as it moves around on the lattice. Only local rules for the molecules are used to move extended objects such as polymers and aggregates around [18]. Finally, each molecule has an associated kinetic energy and a certain direction which is modified, e.g., in collisions.

Three main steps determine the molecular dynamics: (i) rules that propagate field-information particles, (ii) rules that evaluate the received information together with the local state, and (iii) rules that move molecules on the lattice and transform the system into the next time step. Since our system contains several different information particles the update cycle is a bit more complicated than the well-known cycles of the cellular automata and lattice gases. The clock for the update cycle for a traditional cellular automata goes tac, tac, as each cell is updated in parallel. For a traditional lattice gas the clock goes tic-tac, tic-tac, as the fluid particles move and scatter. For the MD lattice gas the update clock goes tic-tic-...tic-tac, tic-tic-...tic-tac, as the field particles are propagated over the lattice (this happens at light speed in the physical system), which may result in rotation of a molecule, a molecular collision, and/or a molecular move (at a different physical time scales). The reason why we use this formulation is because of its simplicity (modulatity and parallizability) and because each part of the dynamics has a clear physical interpretation.

It should be noticed that all the interaction rules are based on first principles. These microscopic interactions are deterministic and reversible, and mass, momentum and total energy all (= kinetic + potential + internal) are conserved. The overall setup corresponds to a microcanonical ensemble. In the next section we shall make all of the above precise.

2.2 LATTICE, DATA STRUCTURES, AND UPDATE FUNCTIONAL

The formal system is defined on a three-dimensional (cubic) lattice, or more precisely on two connected three-dimensional lattices. One of the lattices, $\mathcal{L}_m$ (N^3), has a one-dimensional data structure associated with each lattice point. It represents the possible location of a molecule and it carries part of the molecular object properties: x_1 the molecular type (including no molecule present = vacuum; for vacuum the following variables are also zero), x_2 the molecular orientation, x_3 the kinetic energy, as well as variable x_4 which is used to collect the local potential energy information from the field lattice $\mathcal{L}_f$. This other lattice, the field lattice $\mathcal{L}_f$ ($(2N)^3$), has as many variables $y_1, \ldots, y_d$ associated with each lattice point as there are field interaction types ($d = 9$), which we shall discuss in the next section (section 2.3). These are all propagated from the position lattice $\mathcal{L}_m$, and define the repulsive/attractive field information. Note that not all sites on this lattice are accessable. The fields are only defined at the sites that intersect the "midpoints" of the lattice edges that connect the molecular position

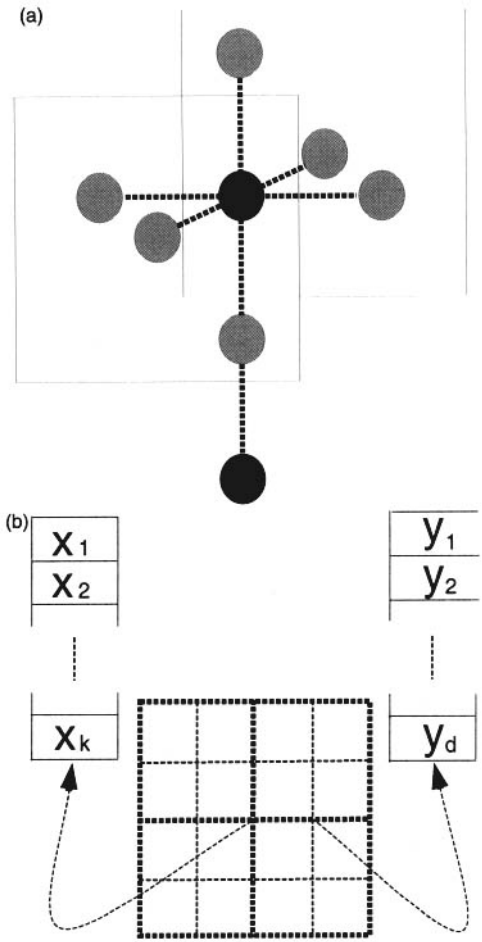

FIGURE 1 The two three-dimensional (cubic) lattices, the molecular lattice $\mathcal{L}_m$ (N^3) and the field lattice $\mathcal{L}_f$ $((2N)^3)$. (a) The field lattice (grey) is face centered with the molecular lattice (black) at the center of the cube. (b) A two-dimensional projection of the two lattices showing their data structures, $(x_1, \ldots, x_k)$ and $(y_1, \ldots, y_d)$ for $\mathcal{L}_m$ and $\mathcal{L}_f$, respectively.

lattice. Please see figure 1. As an example of how a water molecule is represented on the lattice using these structures please see figure 2(a), (b), and (c). Lattice definitions of monomers, polymers, and aggregates are given in figure 3.

The dynamical system that defines our lattice gas is of the form

$$\{S_r(t+1)\} = U\{S_r(t)\}, \quad r = 1, \ldots, n, \tag{1}$$

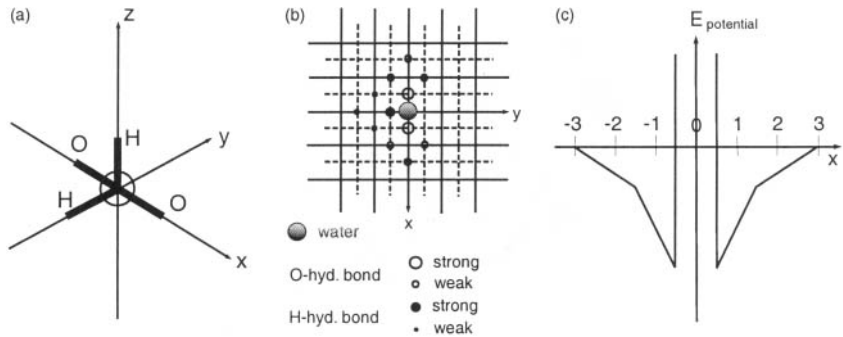

FIGURE 2 Representation of water on the lattice. (a) Note the four hydrogen bond sites. (b) The xy-projection of (force) field from water. (c) Representation of the potential energy of the water molecule along the x-axis.

where

$$S_r = S_r(f_{rs}, x_r, y_s, \tau_r), \quad s = 1, \ldots, 2q\,, \tag{2}$$

denotes the interacting molecular objects (and vacuum) defined on the three-dimensional, cubic lattices $\mathcal{L}_m$ and $\mathcal{L}_f$. Each object has an internal state x_r, neighboring molecular and field states y_s, an object-object interaction function f_{rs} (which has its own state x_r as an argument together with the field state(s) from the object(s) that it is interacting with $y_s, s = 1, \ldots 2q$), and a local time τ_r. To generate the dynamics the object-object interactions have to be scheduled by an update functional U, which is random sequential for this version of our MD lattice gas. And q denotes the nearest neighborhood of each of the lattices, which is 12 (2 times 6) when both lattices are taken into account.

A data structure $\mathcal{M}_t^{(i,j,h)}$ at the $\mathcal{L}_m$ lattice location (i, j, h), at time t, denoting an object S_r—e.g., vacuum if it is empty and a molecule if it is occupied—is given by

$$\mathcal{M}_t^{(i,j,h)} = ({x_1}^{(i,j,h)}(t), \ldots, {x_k}^{(i,j,h)}(t))\,. \tag{3}$$

The field lattice at time t is similarly given by

$$\mathcal{F}_t^{(i',j',h')} = ({y_1}^{(i',j',h')}(t), \ldots, {y_d}^{(i',j',h')}(t))\,. \tag{4}$$

The system update consists of three principal parts: (i) field propagation, (ii) evaluation of local fields, molecular reorientation, and (iii) molecular movement. For the field propagation steps (i) we have

$$\mathcal{F}_{t*}^{(i',j',h')} = F_1(\mathcal{M}_t^{(i,j,h)}), (i, j, k) \in I\,, \tag{5}$$

where I defines the interaction neighborhood where to the molecule irradiates (propagates) its field and where F_1 is an only implicitly given function. The

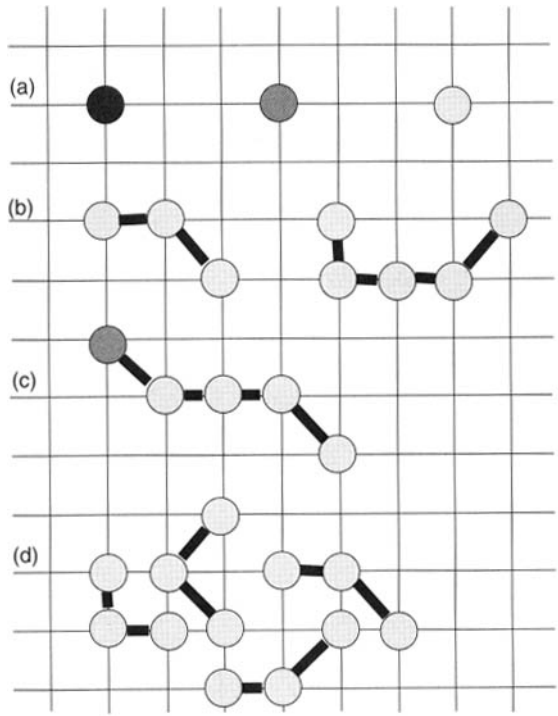

FIGURE 3 Representation of water, monomers, polymers, and aggregates on the lattice. (a) Water (black), hydrophilic monomer (dark grey), and hydrophobic monomer (light grey). (b) Hydrophobic polymers: a trimer and a pentamer. (c) Amphiphilic pentamer. (d) Small aggregate of hydrophobic trimers. It should be noted that each hydrophobic monomer object in a polymer can represent *two* CH_2 groups. See discussion in section 2.4.

evaluation of local fields, molecular reorientation (ii), and molecular movement (iii) on the lattice is given by

$$\mathcal{M}_{t+1}^{(i,j,h)} = F_2(\mathcal{M}_t^{(i,j,h)}, \mathcal{M}_t^{1(i,j,h)}, \mathcal{F}_t{*}^{1(i',j',h')}, \ldots, \mathcal{M}_t^{q(i,j,h)}, \mathcal{F}_t{*}^{q(i',j',h')}), \quad (6)$$

where the other data structures are located at the $q = 6$ neighboring lattice $\mathcal{L}_m$ and $\mathcal{L}_f$ positions (please recall fig. 1). Again F_2 is only an implicitly given function.

However, the detailed structure of each step in the update cycle may in fact be a composition of eqs. (5) and (6). Thus, the general form of the update of an individual data structure element x_p at (i, j, h) at the molecular lattice can be expresses as

$$x_p^{(i,j,h)}(t+1) = f_0(x_1^{(i,j,h)}, \ldots, x_k^{(i,j,k)})(t') \circ$$

$$
\begin{aligned}
&f_1[(x_1^{1,(i,j,h)}, \ldots, x_k^{1,(i,j,h)})(t''), (y_1^{1,(i,j,h)}, \ldots, y_d^{1,(i,j,h)})(t''')] \\
&\quad \circ \ldots \circ \\
&f_6[(x_1^{6,(i,j,h)}, \ldots, x_k^{6,(i,j,h)})(t''), (y_1^{6,(i,j,h)} \ldots, y_d^{6,(i,j,h)})(t''')],
\end{aligned} \tag{7}
$$

where "$\circ$" denotes function composition. This means that the new value of a given variable at a given molecular lattice site is a composed function of the variables at the site (i, j, h) itself and of the variables at the neighboring molecular *and* field sites (each in the six principal directions). An example of such a step is a molecular rotation due to the local force field.

For a more detailed discussion of the formal properties of such a dynamical system we refer to Baas et al. [3], and technical details of the formulation of the individual steps in the update are given in Mayer and Rasmussen [18].

2.3 INTERACTIONS

A Boltzmann distribution of kinetic energies together with potential energies (based on discrete force fields) are implemented to drive the molecular dynamics. Kinetic energies are distributed between colliding molecules following a hard sphere model conserving the momentum. The total potential energy V_{total} of our lattice system with n molecules on a lattice each with $q = 6$ neighbors is described by:

$$
\begin{aligned}
V_{\text{total}} = &\sum_{i=1}^{n} \sum_{j=1}^{q} V_{\text{dip.-dip., charge-charge, H-bond}}^{i,j} \\
&+ \sum_{i=1}^{n} \sum_{j=1}^{q} V_{\text{dip.-ind.dip.}}^{i,j} + \sum_{i=1}^{n} \sum_{j=1}^{q} V_{\text{ind.dip.-ind.dip.}}^{i,j} \\
&+ \sum_{i=1}^{n} \sum_{j=1}^{q} V_{\text{coop.}}^{i,j} .
\end{aligned} \tag{8}
$$

These potential energy terms are implemented to account for specific physicochemical properties of our molecular species as, e.g., dipoles, induced dipoles, hydrogen bond donor and acceptor sites, or polarizability volumes, all crucial parameters for the generation of molecular self-assembly in a polar environment [17, 25]. This set of weak intermolecular interactions given in the above equation is commonly summarized as Van der Waals forces.

The lattice gas interactions conserve mass, energy, and momentum in the interactions. Mass conservation is trivially fulfilled as no fundamental molecular objects are created or destroyed as a result of the interactions. The momentum conservation is guaranteed through an exchange of kinetic energy in the direction of the collision. An example of a collision between two neutral molecules (no attracting or repelling forces) along the x-direction is given in figure 4(a) and a collision between two water molecules with attracting/repelling hydrogen

bond interaction sites is given in figure 4(b). Note how the water-water collision temporally results in a hydrogen bond between the two water molecules and the kinetic energy of the molecules temporally may be interpreted as internal energy (e.g., vibrational energy) of these molecules. As other molecules at some later point interact with this water-water cluster it may break up and the internal energy is again transformed or released into kinetic energy. Note how in figure 4(b) part of the update dynamics also involves a molecular rotation to find a local, potential energy minimum.

The relative values of the different interaction terms in eq. (9) together with the mean of the used Boltzmann distributed kinetic energies are important for the dynamics. In table 1 the different interaction values are listed.

Intermolecular interactions naturally play an essential role in defining molecular self-assembly, considering first the intramolecular structure of solute molecules as, e.g., polymers (their conformation); second denoting solute-solute interactions; and third, solute-solvent interactions. This last type of interaction is considered as decisive for molecular self-assembly, especially denoting the hydrophobic effect in a polar solvent, as described in more detail in section 4.

To represent the key feature of a polar solvent like water with its local and distant order (see section 3), the capability of hydrogen bond formation has to be represented. Formally the hydrogen bond has to be treated as a combination of a dipole-dipole interaction and a charge transfer reaction, and this type of non-covalent donor-hydrogen-acceptor bond is commonly represented via a Lennard Jones potential in force fields.

One important aspect of hydrogen bonds is their strictly defined potential energy surface where the optimum geometry is given by a donor-hydrogen-acceptor-oxygen angle of 180 degrees and a donor-acceptor distance of around 3 Å (Ångstrøms). In this optimum configuration the hydrogen bond contributes with 1–2 kcal/mol to the total potential energy. However, variation of this optimum geometry is immediately followed by an increase in potential energy. Hydrogen bonds are among the strongest intermolecular interactions, but their ideal geometry is highly constrained. In the simulation water molecules and hydrophilic monomers shield for the hydrogen bond fields (see also the Coulomb interactions below).

Another aspect of hydrogen bonds in liquid water is their cooperativity. A single water molecule in bulk water is in a tetrahedral arrangement defined via four hydrogen bonds (recall figure 2). However, the formation of hydrogen bonds is cooperative, i.e., the single interaction strength of a formed hydrogen bond increases if an additional hydrogen bond is formed. The implementation of these important cooperative phenomena for hydrogen bond networks, and also for induced dipoles as described below, is straightforward in our lattice gas, whereas these features are commonly neglected in traditional force-field calculations as they compute potential energies only on the basis of pair potentials, which naturally do not take into account many-body properties as cooperativity. In the present model, hydrogen bonds are implemented to characterize water structure

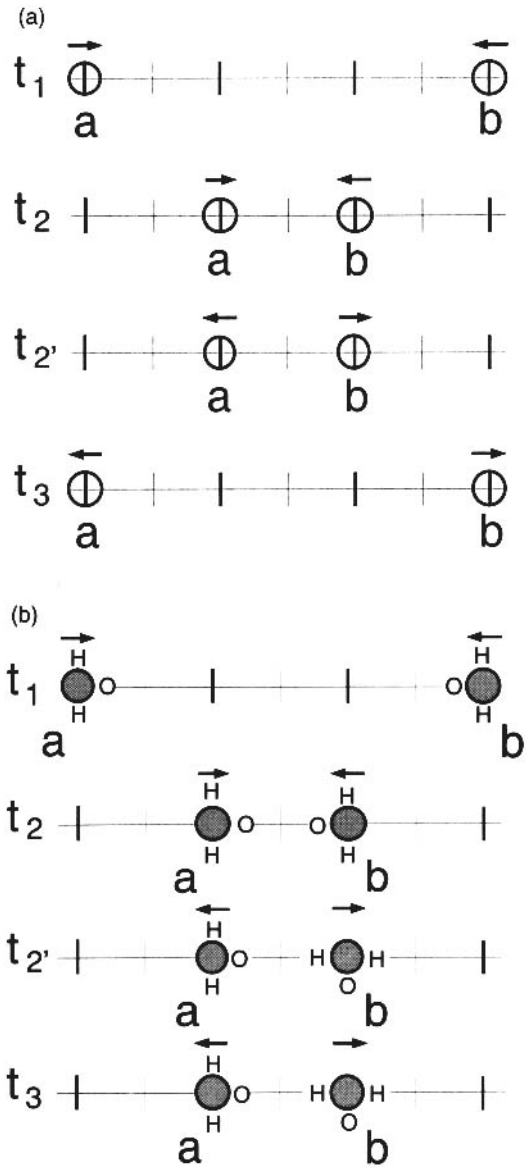

FIGURE 4 Examples of collisions. (a) Collision of two neutral molecules. (b) Collision of two water molecules. In this situation the potential energy of the formed hydrogen bond exceeds the kinetic energy of either of the two molecules so they maintain bonded after the collision. Assuming that b is updated first (random sequential update) note how b rotates (recall update dynamics) and finds local potential minimum.

(i.e., water-water interactions) as well as to define hydrophilic monomer units (e.g., a COOH group).

A Coulomb potential is used to model charge-charge interactions. The relative scale is comparable (here chosen identical) to hydrogen bonds. What differs is the lack of geometric constraint that is present in the hydrogen bond interactions. One important aspect for this type of interaction as well as for the hydrogen bond interaction is the relative dielectric constant ϵ_r, which reflects the shielding of charge fields by a polar environment like water. The relative scale of this shielding is implemented in the course of the charge propagation in our lattice gas: water strongly shields the propagation of the field (ϵ_r around 80), whereas field propagation is less constrained when the charge is in an apolar environment as, for example, given by hydrophobic monomers (ϵ_r around 10). This partial schielding of field propagation is also implemented for the propagation of force fields representing hydrogen bonds to again reflect the high relative dielectric constant in bulk water. Both water molecules and hydrophilic monomers shield the Coulomb in the simulation.

The following interactions represent potentials based on dipoles and induced dipoles, which are commonly about one order of magnitude weaker compared to the interaction strength of hydrogen bonds: Molecules or molecular groups with a permanent dipole moment stabilize respective assemblies via dipole-dipole interactions, which are again modeled via a Coulomb potential. Water and hydrophilic monomer pairs as well as mixed pairs of those two show this dipole interaction. Another interaction for these molecular types is based on dipole-induced dipoles. This effect is based on spatial distortion of the electron distribution on molecular surfaces, which is, for example, caused by a permanent dipole in close proximity to this molecular surface. The strength of this dipole-induced dipole interaction is based on the polarizability volume of the respective molecular surface, which is in general inversely proportional to the electronegativity of the respective atom. Carbon surfaces are therefore more highly polarizable than oxygen surfaces, which results in a stronger interaction of water-hydrophobic monomers and hydrophilic monomers-hydrophobic monomers compared to the interaction between groups holding more electronegative elements (see table 1).

The last type of intermolecular interaction is induced dipole-induced dipole, commonly called dispersion, or London forces. Molecules involved in this interaction are uncharged compounds with no permanent dipole and the interaction is, therefore, solely attributed to spatial fluctuations of the electron distribution on the respective molecular surfaces. We use this potential to model interactions between hydrophobic monomers. Again a cooperativity term is implemented to reflect the cooperativity of induced dipole-induced dipole interactions in bulk apolar phase.

Intramolecular potentials for angles and dihedral angles are not explicitly encoded but are expressed via the intermolecular terms. This approach is also applied in common force fields used for the calculation of potential energies of macromolecules as, for example, the empirical conformational energy program

TABLE 1 Molecular interactions encoded in the MD lattice gas. The description of molecular units on a monomer level depicts the polar solvent water, w; hydrophilic, hphil; and hydrophobic, hphob, and monomers. NH1 and NH2 denotes the neighborhood 1 and 2 on the lattice, the parametrization gives site interactions. The bond energy values in the table can, e.g., be translated into eV by multiplying by -2. For example, the hydrogen bond energy between two water molecules at lattice distance one is ~ -18 eV.

Intermolecular Interaction	Molecular Pairs	NH1 Pair	NH2r Pair
bond stretching, in-diagonal	monomer-monomer bonds	5	∞
hydrogen bond	w-w, phil-w, phil-phil	9	3
cooperativity increment, hydrogen bonds	w-w, phil-w, phil-phil	2	0
charge-charge	phil-phil	9	3
dipole-dipole	w-w, phil-w, phil-phil	2	0
dipole-ind. dipole	w-w, phil-w, phil-phil	1	0
dipole-ind. dipole	w-hphob, hphil-hphob	3	0
ind. dipole-ind. dipole	phob-phob	4	2
cooperativity increment, ind. dipole-ind. dipole	phob-phob	1	0

for peptides, ECEPP/3 [22]. The only intramolecular energy term implemented in the MD lattice gas is bond stretching, which allows monomers within polymers under certain energetic conditions to access in-diagonal lattice points on the cubic lattice.

2.4 PHYSICAL SCALES AND CONSTANTS

To establish a tighter correspondence between the simulated physics in the MD lattice gas and the physics of the experimental systems, we need to define how the physical scales and constants in the two systems correspond to each other. This involves identification of the lattice size, the modeled molecular bond lengths, the force fields, the update steps, the molecular weights, the temperature, and the pressure.

Since H_2O has a molecular weight of about (2 × 1 g/mol+16 g/mol) = 18 g/mol, it has a density of about 1kg/l, each l contains about 55.6 mols of water molecules which corresponds to about 3.3×10^{28} molecules per m^3. In a 20^3 lattice simulation of water we typically have $0.75 \times 20^3 = 6,000$ molecules which correspond to a volume of 182×10^{-27} m^3 which yields a cube side of 5.7 nm. Thus our 20^3 lattice is about 6 nm on each side. This is in good agreement

with the 3 Å oxygen-oxygen distance between two water molecules (recall section 2.3 about the hydorgen bonds) which also yields a cube size of 6 nm.

However, there is an inconsistency between the water-water distance of 3 Å and the CH_2–CH_2 distance in the hydrocarbons as the C–C bond distance in the polymers is about 1.5 Å if we assume that each monomer contains only one CH_2 group. Assuming instead that each monomer object contains *two* CH_2 groups resolves this problem, but it raises a question about the monomer-monomer and the monomer-water field interaction values. The simulation does not easily produce the same dynamics as the experimental systems without a slight modification. The simulations indicate that the hydrophobic monomer-object–monomer-object field interaction needs to be increased a little compared to observed values for the induced dipole-induced dipole interaction between two *single* CH_2 monomers located in two different polymers.

The individual force field interactions in the simulation (see table 1) can be calculated from the hydrogen bond strength of about 1.5 kcal/mol which corresponds to a potential energy of about -10^{-20} J/bond ~ -18 eV/bond. Thus, for example, the dipole-dipole interaction energy at distance 1 corresponds to about $-1/9 \times 18$ eV/bond $= -2$ eV/bond. For simplicity, we have assumed that all fields are local only to lattice distance two. To propagate the fields to any other lattice neighborhood can be implemented without any change of concepts, only with the expense of (a linear) run-time complexity for the field propagation steps.

The molecular mass of all interacting molecular objects are assumed to be identical. This is, of course, a simplification and the masses could easily be disaggregated to fit each of the different molecular types, but given the level of aggregation in the current PLG we have not experimented with this possibility.

A discussion of the molecular lattice occupation (the pressure) and average value for the used Boltzmann distribution (the temperature) is given in section 3.1 where the fundamental properties of the simulated water dynamics is presented. The update cycle (the physical time) is discussed in section 3.2 and Mayer and Rasmussen [17].

2.5 OBSERVABLES

The main observables used to characterize the lattice gas dynamics over time are the average potential energy per molecule, the average number of hydrogen bonds per water molecule, the average moving rate per update for water and monomers, as well as the pair correlations for water-water, water-hydrophilic monomers, water-hydrophobic monomers, water-hydrophobic monomers at ends of polymers, and monomers-monomers. That these observables all are relevant for a detailed understanding of the dynamics was not clear to us from the beginning. However, each of these observables tells a particular part of the story and we shall refer to them in the sections to follow. It should be noted that the signature of the

dynamics is quite different moving from two-dimensional to three-dimensional space.

2.6 IMPLEMENTATION AND GRAPHICS

The MD lattice gas is implemented in plain C and has been developed in a UNIX environment.[1] The simulations are performed on a SUN Ultra 5 workstation where a typical run on a 20^3 lattice for 20,000 updates takes about three hours. For the three-dimensional graphics we have used Mathematica. A listing of the main structure of the code is given in the appendix.

3 MOLECULAR DYNAMICS AND SELF-ASSEMBLY

3.1 WATER (H_2O)

The two most fundamental parts of the dynamics for this lattice gas is (i) the balance between the kinetic—and the potential energy for the molecules—and (ii) an appropriate molecular density on the lattice. Obviously, if the kinetic energy is too low relative to the potential energy, the system freezes up and no self-assembly is possible. If the kinetic energy is too high compared to the potential energy, no bonds can form and no self-assembly is possible either. It is also clear that, as the molecular occupation number on the lattice approaches 1.0, the description breaks down as the dynamics is inhibited. As the molecular occupation becomes too sparse, the description also breaks down because the dynamics no longer describes a liquid. The details of these phase transitions in this system are not yet clear to us and will be discussed in a forthcoming publication. In the following simulations we use a mean value (T) for the Boltzmann distribution of the kinetic energies between 10 and 15 which balances well with the potential energies given in table 1. These T values may be interpreted to correspond to "room" temperature. The molecular lattice occupation is between 0.6 and 0.8 in the reported simulations which may correspond to the usual 1 atm pressure in the corresponding experimental systems. These relations yield an average moving rate of about 0.1 for each water molecule per update and a little higher moving rate for hydrophobic monomers. A further discussion of these basic issues can also be found in Mayer et al. [17].

In addition to these fundamental properties, the polar solvent (water) is characterized by an extended hydrogen bond network resulting in local and distant order which is, for example, reflected in the radial distribution function of water and which is the basis for the hydrophobic effect [17]. Different examples of radial distribution functions (correlation functions) are discussed in figure 5.

The hydrophobic effect, describing phase separation of apolar compounds in a polar environment, is generated as a result of the relative balance between

[1]The source code may be obtained from the corresponding author, Steen Rasmussen.

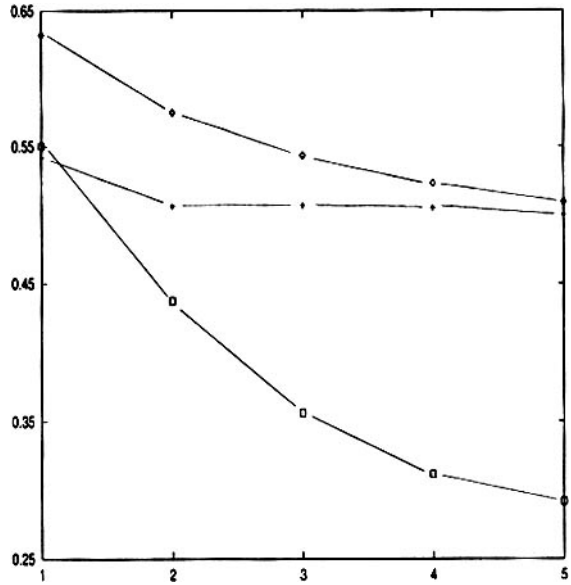

FIGURE 5 Pair correlation functions starting at lattice distance 1 (all identical to 1 at distance 0) for water-water (cross) in pure water at lattice occupation 0.5; water-water (diamonds) in the presence of bicontinous amphiphilic phases in water at lattice occupation 0.75 (0.5 water and 0.25 amphiphilic pentamers); monomer-monomer (boxes) in the same situation. Note the increased local ordering of water in the presence of cosolvents.

the water-water interactions, the water-monomer interactions, and the monomer-monomer interactions. If water likes water much more than water likes monomers and monomers like each other and water about equally well, there is an energetic gain from isolating the hydrophobic monomers from the water so that the water molecules can be in proximity to each other and participate in the hydrogen bond network.

3.2 HYDROCARBONS ($CH_3[CH_2]_NCH_3$)

The biological solvent, water, is characterized by an extended hydrogen bond network which results in ordered clusters that constantly form and break up, as discussed in section 3.1. The enthalpic and entropic balance, which results in an optimum free energy for liquid water, is crucially affected by solvating *hydrophobic* (apolar) compounds which ultimately may result in positive changes in the free energy of solvation processes in the course of solvating apolar surfaces. Surprisingly, this effect is not based on a net loss of enthalpic contributions,

as experimental data on transfer experiments of apolar compounds from vapor phase to water give ΔH values around zero [26].

Mainly two contributions compensate the energetic loss of hydrogen donor—and acceptor sites in mixtures of hydrophobic compounds in water compared to pure bulk water. First, cooperative intermolecular interactions between hydrophobic objects of the induced dipole type result in low energetic minima for such separated phases, although the single pair interactions are comparably weak. Another, less obvious cause for the isoenthalpic situation is the additional stabilization of the remaining hydrogen bond forming water molecules in the mixture [17]. This additional "freezing" of water molecules, especially in the innermost solvation shells around a hydrophobic cluster points toward the entropy as a central element in controlling solvation. The changes of $T\Delta S$ are also strongly negative for solvating hydrophobic compounds in water, which ultimately results in a positive change of the free Gibbs energy ΔG for the overall solvation process of apolar compounds in water [26].

Earlier, we performed a variety of lattice gas simulations in two-dimensions [17, 18] and implemented these experimental findings by realizing a net isoenthalpic situation comparing pure water and water-hydrophobic monomer mixtures, which results in $\Delta H \simeq 0$ for the solvation process in the simulation. However, phase separation of hydrophobic compounds in aqueous solution is still realized in our simulations via the entropy-driven hydrophobic effect. Phase separation is observed both for water-hydrophobic monomer as well as for water-hydrophobic polymer situations and both processes are characterized by $\Delta H \simeq 0$ although the pairwise interaction between two hydrophobic monomers and between water and a hydrophobic monomer is roughly kept isoenthalpic. However, organizing the hydrophobic monomers in polymers enhances their capability to phase-separate and the longer the polymers, the easier they will separate in water.

In the following two examples, both the hydrophobic effect and a slight monomer-monomer preference over a monomer-water interaction drive the phase separation processes (see table 1). Typical chemical analogs to our model monomers are acetonitrile and propionitrile, which show weak interactions with water, but an increased tendency to self-associate.

In figure 6, hydrophobic trimers (3-mers) are initially randomly placed on a 20^3 lattice with the parameters as discussed in section 2.4. The approximate side length of the cube is about 6 nm. The molecular (object) density on the lattice is 0.75 with 66% water and 33% hydrophobic monomers ($\sim$ 11% trimers). The simulation time between start and finish is 20,000 updates which corresponds to about 20×10^{-6} sec = 20 microseconds, physical time.

In figure 7 32,000 water molecules and 3,200 hydrophobic pentamers (5-mers) are randomly placed on the lattice and simulated for 50,000 time steps ($\sim$ 12 μs) on a 40^3 lattice ($\sim$ 12 nm). To better show the structure of the separation only the hydrophobic polymers are plotted. A rough estimate of the "cluster" size in this system yields less than a thousand pentamers per cluster.

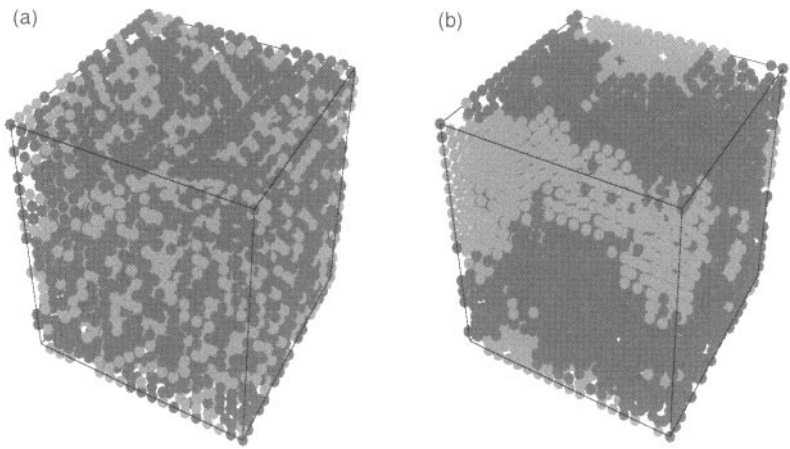

FIGURE 6 (a) Random, initial distribution of hydrophobic trimers (3-mers) and water and (b) phase separation after 20,000 updates (about 20×10^{-6} sec = 20 microseconds, physical time) on a 20^3 lattice (about 6-nm cube side = 6×10^{-9} m). Water shown in blue and hydrophobic monomers shown in yellow. Parameters: $T = 13$, $\rho = 0.74$, and monomer concentration 33% (4,000 water molecules and 666 hydrophobic trimers).

This is about the same cluster size as for the trimers in figure 6 where only a single cluster is formed due to the smaller lattice size. Thus, we do not observe much difference in the phase separation dynamics as we vary the hydrocarbon chain length between three and five.

The potential energy as a function of time for the phase separation is shown in figure 8 where the transition dynamics is clearly reflected.

3.3 AMPHIPHILES ($COOH[CH_2]_NCH_3$)

The simplest form for ordered aggregates that assemble in water are built by *amphiphilic* polymers which are polymers with a *hydrophilic* head and a *hydrophobic* tail. Hydrotopes [34] and lipids [32] are typical representatives for this type of polymer and their interaction dynamics in polar solvents results in self-assembly products with characteristic morphology as, for example, hydrotope aggregates, micelles, and membranes. The mechanism for organized self-assembly is attributed to the hydrophobic effects in which an entropically favored release of interfacial water provides the driving force for such a process [4, 21, 35]. The additional (object) complexity given by the dual functionality within the poly-

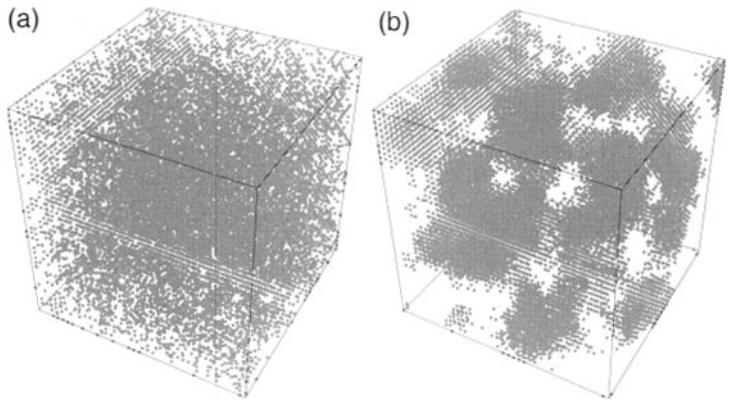

FIGURE 7 Water and hydrophobic pentamers on a 40^3 lattice (12-nm cube side). (a) Random, initial distribution of hydrophobic pentamers (5-mers) and water (water molecules not shown) and (b) phase separation after 50,000 updates (about 50×10^{-6} sec physical time). Parameters: $T = 13$, $\rho = 0.75$, and monomer concentration 33% (32,000 water molecules and 3,200 hydrophobic pentamers). Typical "cluster" size is 500–1,000 pentamers.

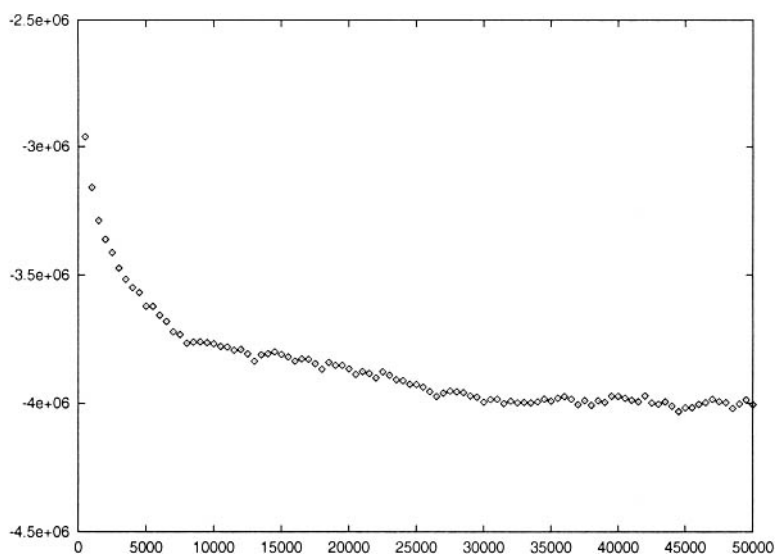

FIGURE 8 Total potential energy as a function of time for the phase separation dynamics of hydrocarbons in water also shown in figure 7. Compare with amphiphilic dynamics in figure 10.

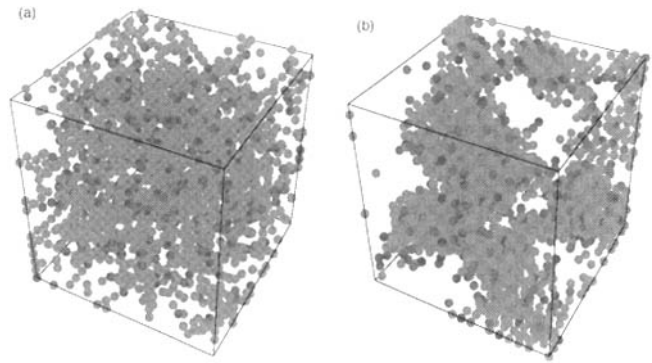

FIGURE 9 Water and high concentration of amphiphilic pentamers on a 20^3 lattice (6-nm cube size). (a) Random, initial distribution of amphiphilic pentamers and water, and (b) bicontinous (wormlike) phases after 20,000 updates ($\sim$ 20 microseconds physical time). Hydrophobic monomers shown in yellow and hydrophilic monomers shown in green. Water not shown. Parameters: $T = 15$, $\rho = 0.75$, and monomer concentration 33% (4,000 water molecules and 400 amphiphilic pentamers).

mer affords the formation of more order structures compared to the unstructured polymer clusters composed solely of hydrophobic polymers. In the one hand, such ordered structures can be demonstrated by the experimental data. For example, amphiphiles containing stilbene or azobenzene chromophore can form highly ordered structures such as large ($>$ 100 nm in length) plates, long tubules or spherical vesicles in water which can be directly detected by cryo-transmission electron microscopy (cryo-TEM) [33]. On the other hand, formation of such ordered structures resulting from the self-assembly of amphiphiles can also be simulated. Figures 9 and 11 show initial and end snapshots of the simulated amphiphilic polymer self-assembly process in an aqueous environment which results in the formation of bicontinous (wormlike) and micellar structures respectively. Water not shown, hydrophilic head groups are green and the hydrophobic tail monomers are yellow. Both structures are characterized by a hydrophobic core and a well-solvated hydrophilic surface. The only difference between these two simulations is the concentration of amphiphiles which is higher in figure 9 (10% amphiphiles) than in figure 11 (2% amphiphiles). It should be noted that these amphiphilc polymer concentrations are about five times higher than the corresponding experimental concentrations that generate the same mesoscopic structures.

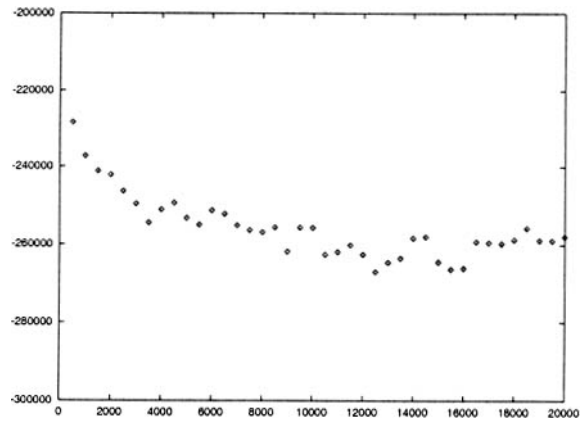

FIGURE 10 Total potential energy as a function of time for the dynamics resulting in bicontinous phases also shown in figure 9. Compare with hydrocarbon dynamics in figure 8 and see discussion in text. Potential energy shows a "compromise" between minimizing hydrocarbon-water contact and solvating hydrophilic head groups in water.

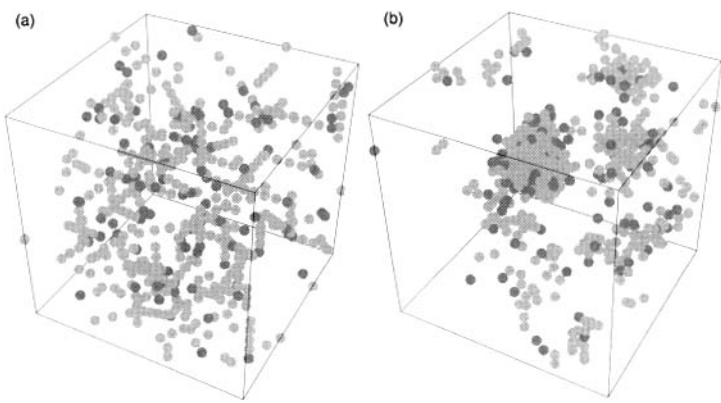

FIGURE 11 Water and low concentration of amphiphilic pentamers on a 20^3 lattice (6-nm cube size). (a) Random, initial distribution of amphiphilic pentamers (hydrophilic head groups green and hydrophobic monomers yellow) and water (water not shown) and (b) micelle formation after about 7,000 updates. Parameters: $T = 13$, $\rho = 0.75$, and monomer concentration 10% $\sim$ 2% polymer (volume) concentration (5,500 water molecules and 100 amphipilic pentamers). These small micelles (15–40 amphiphiles each) are very dynamics and "fluid" structures as they form and disassociate continuously.

For both figure 9 and 11 it is clear that virtually all head groups are exposed to the water phase (none or only very few headgroups are "buried" in the hydrocarbon environment). As a consequence the cluster size or worm thickness is correspondingly small compared to the hydrocarbon clusters in figures 6 and 7. Thus, the global potential bond energy gain from this process is much less pronounced than in the corresponding phase separation process involving hydrocarbons, as seen in figure 10. The resulting structures are an energy "compromise" in this situation as there is still a lot of hydrocarbon-water contact in figures 9 and 11 compared to figures 6 and 7.

We again note that this assembly process is not driven by the enthalpic contributions, for example, coming from specific interactions between hydrophobic monomers. Mainly entropic contributions based on solvating the accessible hydrophobic surfaces guide the formation of the self-assembly products.

Also note that this mesoscopic self-assembly product represents a higher-(third) order structure generated by the dynamics at the microscopic level [3, 18, 29]. This is true as the micelle obviously has, for example, a surface, an inside and an outside as well as a lifetime (all polymers exchanged), which can neither be observed at the level of the individual polymer (second-order structure) that make up the micelle, nor at the level of the individual water molecules or monomers (first-order structures) which make up the polymers.

4 DISCUSSION

The ansatz we have developed for generated higher-level structures and novel functionalities simply by increasing the object complexity of the interacting lattice gas objects does not seem to have any principal limitations. Our ansatz is in the same spirit as when the experimental chemist introduces more complexity among the basic molecular building blocks (e.g., more variety) used in a single experiment and thereby enables the experimental system to assemble into more intricate structures with multilevel, novel properties. In this chapter we represent and discuss three different levels of description within the MD lattice gas: the water and monomer level, the hydrocarbon and amphipilic polymer level, and the aggregate level consisting of polymer self-assembled structures. Earlier we have demonstrated how micellar self-reproduction is possible in simulation [18, 20] and how fifth-order emergent properties can be generated in simulation [30] although the latter is not yet implemented.

We have set out to construct a lattice gas simulation that is able to address both microscopic and mesoscopic questions that confront the experimental chemist when studying molecular self-assembly. This has unfortunately resulted in a formal system where the general level of complexity is high. It is not a simple "toy" lattice gas from which it is easy to obtain analytical results. A very interesting theoretical question is how a toy model can be constructed which is

able to generate higher-order structures and functionalities as found in both in real molecular self-assembly and in the MD lattice gas.

As we use a cellular automata approach—discrete space and discrete time—we gain a number of conceptual and computational advantages, but at the same time we create a number of other problems for our description. The advantages are mainly given by conceptual and computational ease by which new higher-order structures can be autonomously constructed. The main disadvantages follow from the nature of the lattice which restricts the degrees of freedom so much that the representation in some situations looses its beauty. Molecules can only move in six different directions, as can force fields, and diverse molecular bond sizes are difficult to implement. These lattice imposed restrictions are perhaps clearest for the polymer dynamics.

As an explicit representation of the polymers is made to obtain realism a nontrivial problem arises concerning how to move an extended object (a polymer) only using local rules. This problem can be solved in a variety of ways and our use of a random sequential update simplifies the problem. This problem can be completely bypassed by representing the polymers as first-order objects as, for example, in Conveney et al. [9]. Although such a representation allows a generation of mesoscopic structures, most microscopic issues are lost.

Another well-known problem stems from the (realistic) assumption that at most one molecule can reside on a single lattice site. In two dimensions this turns out to be a major problem which easily causes "traffic jams" (lattice dynamics freeze up locally) when assemblies are formed. Thus the description breaks down unless, for example, a boundary layer or some other action is taken. However, this problem is virtually gone in three dimensions. We are currently studying (bilayer) membrane stability and this lattice gas even works for that purpose [36]. However, an obvious future extention of the MD lattice gas could, in a natural manner, consist of a higher (say, double) lattice resolution which should remove this traffic jam problem completely as well as give us a more precise field description.

A higher velocity (kinetic energy) resolution is also desirable. For example, two (or more) classes of kinetic energy which result in an update where molecules with different kinetic energy can be propagated zero, one, two, or more lattice sites in a single update as opposed to only zero or one in the current formulation. Besides a more realistic detailed dynamics this should also give more realistic phase transitions for the ice-liquid-gas transitions than we currently have.

Also a longer range field propagation is desirable, such as, in situations where charged surfaces are formed. Without shielding, a charged, planar surface will have a very long range mesoscopic field which we cannot model with only a field propagation to neighborhood distance two. Note, however, that it is a trivial extention to include this feature in the dynamics.

The observed differences between the simulated and the experimental micellation concentrations of amphiphiles can presumably be corrected by modifying the parameters in table 1.

Finally we must remember that as we increase the realism the complexity goes up as well as does the simulation time and for the very detailed questions at the ps-ns and Å-nm scales the long tradition of molecular dynamics simulations should be consulted. Of most interest is perhaps to formalize an up-and-down scaling that allows a "linking" of simulations at these two different levels of description.

5 CONCLUSION

The ultimate constructive dynamical systems are found in natural molecular self-assembly and self-organization processes. These systems can obviously generate successive levels of higher-order emergent properties (dynamical hierarchies) without any apparent limit.

A MD lattice gas simulation is defined to produce realistic water dynamics, polymer water phase separation, as well as molecular self-assembly and self-organization dynamics in three dimensions. This system demonstrates a third-order dynamical hierarchy, but the ansatz used is applicable for the generation of any order of emergence.

The presented MD lattice gas fills a modeling gap between traditional MD and lattice gas methods. It thus offers unique computational capabilities for studying complex molecular interactions on nanometer to micrometer length scales and over time scales of nanoseconds to milliseconds on modern workstations.

The gist of the formal representation is that matter and (force) fields can be represented as information particles residing and propagating on a lattice. The local cellular automaton rules define the polymer lattice automaton. All interactions are derived from Newton's laws and the corresponding microscopic and mesoscopic experimental observables can be compared in a direct manner. The details of the phase separation dynamics for hydrocarbons in water is demonstrated, and bicontinuous (wormlike) phase generation and micellation is demonstrated for amphiphiles in water. The physical observables are identified and discussed.

ACKNOWLEDGMENTS

The authors would like to thank Claes Andersson, Jesse Taylor, Gottfried Köhler, Liaohai Chen, and Arno Lukas for valuable discussions and suggestions. This work was financially supported by the Los Alamos Astrobiology section of the LDRD-CD Space Science grant. Martin Nilsson was in addition financially supported by the Santa Fe Institute and the Center for Nonlinear Studies at Los Alamos. Bernd Mayer thanks the Austrian Academy of Sciences for generous support within APART.

APPENDIX

A listing of the main structure of the code is given below. The arrays `random_coord[i][k]` define the next lattice point to update in the randomized sequence generated by `randomize_update()`. The structure of main should be self-explanatory. For more details please contact corresponding author.

```
int main(void)

  initialize_rotation_matrices();
  make_molecules();
  number_of_molecules();
  initialize_arrays();
  initialize_lattice();
  create_readme_file();
  initialize_files();
  rotate_fields();

  for(time=0; time<time_steps; time++)
  {
    print_to_files();
    randomize_update();
    set_fields_to_zero();

    for(x=0; x<lattice_size; x++)
     for(y=0; y<lattice_size; y++)
       for(z=0; z<lattice_size; z++)
         propagate_field(x,y,z);

    for(x=0; x<lattice_size; x++)
     for(y=0; y<lattice_size; y++)
       for(z=0; z<lattice_size; z++)
         rotate_molecule(random_coord[0][x],

    random_coord[1][y],random_coord[2][z]);

              for(x=0; x<lattice_size; x++)
      for(y=0; y<lattice_size; y++)
        for(z=0; z<lattice_size; z++)
          elastic_collision(random_coord[0][x],

    random_coord[1][y],random_coord[2][z]);

    for(x=0; x<lattice_size; x++)
     for(y=0; y<lattice_size; y++)
       for(z=0; z<lattice_size; z++)

move_molecule(random_coord[0][x],random_coord[1][y],

    random_coord[2][z]);
  }
close_files();
return 0;
};
```

REFERENCES

[1] Aranda-Espinoza, H., Y. Chen, N. Dan, T. C. Lubensky, P. Nelson, L. Ramos, and D. A. Weitz. "Electrostatic Repulsion of Positively Charged Vesicles and Negatively Charged Objects." *Science* **285** (1999): 394.

[2] Baas, N. A. "Emergence, Hierarchies and Hyperstructures." In *Artificial Life III*, edited by C. G. Langton, 515–537. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVII. Reading, MA: Addison-Wesley, 1994.

[3] Baas, N., M. Olesen, and S. Rasmussen. "Generation of Higher Order Emergent Structures." Technical report, LA-UR 96-2921, Los Alamos, NM, 1996.

[4] Bonilha, J. B. S., R. M. Z. Georgetto, A. C. Tedesco, L. Miola, and D. Whitten. "Counterion Effects on the Structure and Solubilization Properties of Anionic Detergents: Role of Hydrophobic Cations in Stabilizing Micelles and Attenuating Their Solvent Properties." *J. Phys. Chem.* **93** (1989): 367–372.

[5] Bachmann, P. A., P. Walde, P. L. Luisi, and J. Lang. "Self-Replicating Reverse Micelles and Chemical Autopoiesis." *J. Am. Chem. Soc.* **112** (1990): 8200.

[6] Bachmann, P. A., P. L. Luisi, and J. Lang. "Autocatalytic Self-Replicating Micelles as Models for Prebiotic Structures." *Nature* **357** (1992): 57.

[7] Boghosian, B. M. "Lattice Gases and Cellular Automata." ⟨http://xxx.lanl.gov, comp-gas/9905001⟩, 5 May, 1999.

[8] Chen, S. H., J. S. Huang, and P. Tartaglia, eds. *Structure and Dynamics of Strongly Interacting Colloids and Supramolecular Aggregates in Solution.* Dordrecht: Kluwer, 1992.

[9] Coveney, P., A. Emerton, and B. Bogoshian. "Simulation of Self-Reproducing Micelles using a Lattice Gas Automaton." Oxford University Technical Paper 13S, 96 (1996).

[10] Doolen, G., ed. "Lattice Gas Methods for PDE's: Theory, Applications, and Hardware." *Physica D* **47** (1991).

[11] Fontana, W., and L. Buss. "The Arrival of the Fittest: Toward a Theory of Biological Organization." *Bull. Math. Biol.* **56** (1994): 1–64.

[12] Frisch, U., B. Hasslacher, and Y. Pomeau. "Lattice Gas Automata for the Navier-Stoke Equation." *Phys. Rev. Lett.* **56** (1986): 1722.

[13] Ghadiri, M. R., J. R. Granja, and L. K. Bühler. "Artificial Transmembrane Ion Channels from Self-Assembling Ppeptide Nanotubes." *Nature* **369** (1994): 301.

[14] Haken, H. "Information Compression in Biological Systems." *Bio. Cybern.* **56** (1987): 11.

[15] Hoch, H. C., L. W. Jelinski, and H. G. Craighead, eds. *Nanofabrication and Biosystems: Integrating Materials Science, Engineering and Biology.* New York: Cambride University Press, 1996.

[16] Kawakatsu, T., K. Kawasaki, M. Furusaka, H. Okabayashi, and T. Kanaya. "Late Strange Dynamics of Phase Separation Processes of Binary Mixtures Containing Surfactants." *J. Chem. Phys.* **99(10)** (1993): 8200–8217.

[17] Mayer, B., G. Koehler, and S. Rasmussen. "Simulation and Dynamics of Entropy Driven, Molecular Self-Assembly Processes." *Phys. Rev. E.* **55(4)** (1997): 4489–4500.

[18] Mayer, B., and S. Rasmussen. "Lattice Molecular Automata (LMA): A Simulation System for Constructive Molecular Dynamics." *Intl. J. Mod. Physics C* **9(1)** (1998): 157–178.

[19] Mayer, B., and S. Rasmussen. "Self-Reproducing Dynamical Hierarchies in Chemical Systems." In *Artificial Life 6*, edited by C. Adami, R. K. Belew, H. Kitano, and C. Taylor, 123. Cambridge, MA: MIT Press, 1998.

[20] Mayer, B., and S. Rasmussen. "Dynamics and Simulation of Micellar Self-Reproduction." *Phys. Rev. E* (1999): submitted.

[21] Menger, F. M. *Micelles, Microemulsions, and Monolayers*, edited by D. O. Shah, 53–72. Marcel Dekker, 1998.

[22] Nemethy, G., K. D. Gibson, K. A.Palmer, C. N. Yoon, G. Paterlini, A. Zagari, S. Rumsey, and H. A. Scheraga. "Energy Parameters in Polypeptides. 10. Improved Geometrical Parameters and Nonbonded Interactions for Use in the ECEPP/3 Algorithm, with Application to Proline—Containing Peptidesi." *J. Phys. Chem.* **96** (1992): 6472.

[23] Oberholzer, T., R. Wick, P. L. Luisi, and Ch. K. Biebricher. "Enzymatic RNA Replication in Self-Reproducing Vesicles: An Approach to a Minimal Cell." *Biochem. Biophys. Res. Commun.* **207** (1995): 250.

[24] Ostrovsky, O., M. A. Smith, and Y. Bar-Yam. "Applications of Parallel Computing to Biological Problems." *Ann. Rev. Biophys. Biomol. Struct.* **24** (1995): 239–267.

[25] Privalov, P. L., and S. J. Gill. "Stability of Protein Structure and Hydrophobic Interaction." *Adv. Protein Chem.* **39** (1988): 191.

[26] Privalov, P. L. "Thermodynamic Problems of Protein Structure." *Ann. Rev. Biophys. Chem.* **18** (1989): 47.

[27] Rasmussen, S., C. Knudsen, and R. Feldberg. "Dynamics of Programmable Matter." In *Artificial Life II*, edited by C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, 211–254. Santa Fe Institute Studies in the Sciences of Complexity, Proc. Vol. XVII. Reading, MA: Addision-Wesley, 1991.

[28] Rasmussen, S., and J. Smith. "Lattice Polymer Automata." *Ber. Bunsenges. Phys. Chem.* **98(9)** (1994): 1185–1193.

[29] Rasmussen, S., N. Baas, C. Barrett, and M. Olesen. "A Note on Dynamical Hierarchies." In *Self-Organization in Complex Structures—From Individual to Collective Dynamics*, edited by Frank Schweitzer, ch. 7, 83–89. Gordan & Breach, 1997.

[30] Rasmussen, S., S. Colgate, K. Lackner, G. Koehler, H. Li, B. Mayer, M. Nilsson, J. Solem, and D. Whitten. "A Possible Universal Origin of Life." In preparation, 1999.

[31] Rothman, D. H., and S. Zaleski. *Lattice Gas Automata: Simple Models of Complex Hydrodynamics.* Cambridge, MA: Cambridge University Press, 1997.

[32] Schnur, J. "Lipid Tubules: A Paradigm for Molecular Engineered Structures." *Science* **262** (1993): 1669.

[33] Song, X., J. Perlstein, and D. G. J. Whitten. "Supramolecular Aggregates of Azobenzene Phospholipids and Related Compounds in Bilayer Assemblies and Other Micro-Heterogeneous Media: Structure, Properties, and Photoreactivity." *J. Am. Chem. Soc.* **119** (1997): 9144–9159.

[34] Srinivas, V., G. A. Rodley, K. Ravikumar, W. T. Robinson, M. N. Turnbull, and D. Balasubramanian. "Molecular Organization in Hydrotrope Assemblies." *Langmuir* **13** (1997): 3235.

[35] Tanford, C. *The Hydrophobic Effect: Formation of Micelles and Biological Membranes*, 2d ed. New York: Wiley, 1980.

[36] Taylor, J., A. Ni, and S. Rasmussen. "Amphiphilic Bilayer Stability as a Function of Tail Chain Length and Head Charge." In preparation, 1999.

[37] Whitten, D., L. Chen, C. Geiger, J. Perlstein, and X. Song. "Self-Assembly of Aromatic-Functionalized Amphiphiles: The Role and Consequences of Aromatic-Aromatic Noncovalent Interactions in Building Supramolecular Aggregates and Novel Assemblies." *J. Phys. Chem.* **102** (1998): 10098.

[38] Wolfram, S. "Cellular Automaton Fluids." *J. Stat. Phys.* **45** (1986): 471.

Simulating Digital Logic with the Reversible Aggregation Model of Crystal Growth

Raissa M. D'Souza
George E. Homsy
Norman H. Margolus

We are concerned with understanding the implicit computation occurring in a physical model of crystal growth, the Reversible Aggregation (RA) model. The RA model is a lattice gas model of reversible cluster growth in a closed two-dimensional system, which captures basic properties of physics such as determinism, locality, energy conservation, and exact microscopic reversibility. There are three species of particles in the RA model: gas, heat, and crystal. A diffusing gas particle may aggregate when contacting the boundary of a crystal cluster. Latent heat is released during each aggregation event and is explicitly modeled by introducing a heat particle into a diffusing heat bath. Conversely a cluster member at the boundary of the crystal may absorb a heat particle and evaporate, becoming a diffusing gas particle.

Allowing ourselves complete control over all the initial conditions of the model, we show that the RA model can simulate any logic circuit, and, hence, perform any computation. The mobile gas and heat particles are used as logic signals. The paths these particles take are the wires.

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

Sequences of conditional crystallization events form the basis of the logic gates. We show how to embed a universal single use gate into the dynamics of the model, then show how to construct a reusable universal gate, showing the system is capable of space-efficient computation. We show how to build arbitrary logic circuits by interconnecting gates. This requires steering and routing the signals, delaying them, and letting them cross. Finally, we briefly discuss the relationship of computation in the RA model to computation in real physical systems.

1 OVERVIEW

We examine the computational capabilities of a physical model of crystal growth, the Reversible Aggregation (RA) model [3], which captures basic properties of physics such as determinism, locality, energy conservation, and exact microscopic reversibility. The RA model is a lattice gas model of reversible cluster growth in a closed two-dimensional system. It was introduced as a microscopically reversible physical model for studying the thermodynamics of crystal growth and pattern formation. By microscopically reversible we mean that from any state in the system we can recover the previous state exactly. There are three species of particles in the RA model: gas, heat, and crystal. A diffusing gas particle may aggregate at the boundary of a crystal cluster; if it lands next to a single nearest-neighbor crystal particle, it may become a crystal particle, enlarging the cluster. Latent heat is released during each aggregation event and is explicitly modeled by introducing a heat particle into a diffusing heat bath. Conversely if a heat particle contacts a singly connected cluster member, it may be absorbed and that crystal particle will "evaporate" from the cluster, becoming a gas particle. When started with a dilute gas and a single crystal seed particle the model exhibits an initial regime of rapid nonequilibrium growth followed by a slow quasistatic regime with a well-defined temperature. In the first regime the crystal rapidly grows, in the second the crystal slowly anneals.

The present study showing a construction for computational universality in the RA cellular automaton seems fitting for these proceedings of a workshop on "Constructive Cellular Automata." The mobile gas and heat particles are the logic signals used in the computation. The paths these particles take are the wires. We show how to steer and route the signals, how to delay them, and how to allow signals to cross each other. Crystallization events occur only at sites with one nearest neighbor which is already a crystal. By routing a control signal through a potential crystallization site we can conditionally create new potential crystallization sites. Such sequences of conditional crystallization events are the basis of our logic gates. We show how to embed into the dynamics of the RA model a universal single use gate, and then how to embed a reusable universal gate. We show how to interconnect gates and thus how to build arbitrary

digital logic circuits, proving that the RA model is capable of space-efficient computation.

In this chapter we take a microscopic perspective; that is we give ourselves control over all of the microscopic degrees of freedom, namely the initial positions of all the gas, crystal, and heat particles, and detailed control over the microscopic parameters controlling the pseudorandom motion of the gas and heat particles. With microscopic control and synchronous time evolution we can compute with this discrete lattice system. A more general issue is understanding computation in real physical systems where we have control only over macroscopic degrees of freedom and where we cannot depend on perfect synchronization. In the final section we will address issues of whether we have abstracted concepts from our microscopic dynamics which apply to computation in real physical systems.

2 THE REVERSIBLE AGGREGATION MODEL

The RA model is a reversible, deterministic model of cluster growth in a closed two-dimensional lattice system. It extends the canonical Diffusion Limited Aggregation (DLA) model of cluster growth on a lattice [13]. The DLA model is an irreversible, deterministic model with two particle species: gas and crystal. The gas particles follow a pseudorandom walk along the lattice sites, resulting in diffusive behavior at the scale of several lattice sites. If a diffusing gas particle contacts a stationary cluster member, it aggregates, itself becoming a stationary cluster member. DLA is a serial model: Only one gas particle diffuses at a time. A few parallel versions of DLA, with multiple particles diffusing at once, have been considered [5, 10, 11, 12]. The parallel model we will extend begins with a uniform dilute configuration of gas particles. They aggregate, but no more are ever added to the system. Before a substantial fraction of the particles aggregate, this parallel version of DLA is equivalent to the serial version [11, 12]. When a large cluster has formed, the simulated structure can be compared to structures found in nature, typically by comparing the fractal dimensions. DLA is not a thermodynamic model: Particles stick irreversibly, so there is no notion of detailed or semi-detailed balance.

The RA model extends the DLA model by placing a parallel DLA model in contact with a heat bath, implemented as a field of diffusing heat particles, or "tokens," on a superimposed lattice. Both gas and heat particles diffuse throughout the system independently, each on their own lattice, under pseudorandom dynamics. They diffuse freely over the crystal and through empty space. The gas and heat particles interact through crystallization and evaporation events along the boundary of the crystal.

A potential crystallization site for a gas particle is a site unoccupied by crystal, with exactly one nearest neighbor occupied by crystal. Upon reaching such a site a gas particle will aggregate, becoming a crystal particle and releasing a heat particle, which represents the latent heat of crystallization. This interaction

is contingent on there being room locally in the heat bath to accept the heat particle. Explicitly modeling the latent heat released upon aggregation provides a mechanism for modeling the inverse process of evaporation. Similarly, a potential evaporation site is a site occupied only by a singly connected crystal particle. A heat particle arriving at such a site is absorbed by the crystal particle which evaporates, becoming a gas particle. This is contingent on there being no gas particle already at that site.

The model is implemented with a two phase rule. Diffusion steps, in which the particles move, alternate with interaction steps in which the gas, heat, and crystal interact, allowing aggregation and evaporation. The interaction rule is as given above. The details of the diffusion implementation are given in section 4.1. In order to facilitate a parallel updating of the space, we divide the system into checkerboard sublattices. Since our interaction range is nearest neighbor, we can update all the even parity sites while holding the odd parity sites fixed and vice versa.

In our cluster growth simulations, we begin with an empty heat bath, thus only crystallization can occur. We observe rapid, nonequilibrium growth of the cluster and concomitant increase in the population of the heat bath. The occupancy of the heat bath (hence, also the mass of the cluster) quickly reaches steady state, meaning evaporation and crystallization events are equally likely. At this point the system has attained a single well-defined temperature, despite the fact it has not yet reached thermodynamic equilibrium. During the subsequent slow approach to thermodynamic equilibrium we observe a quasistatic annealing of the cluster. The cluster morphology, as quantified by the fractal dimension, undergoes a transition from the typical ramified pattern observed for irreversible models of diffusive cluster growth (resembling frost on a window pane), to the highest entropy macrostate allowed for a connected cluster in a finite volume: a branched polymer. Figure 1 shows the typical cluster morphology in each of the two regimes. The small grey dots also shown are the gas particles.

The dynamics of the RA model captures a number of properties of realistic microscopic physical dynamics such as locality, conservation of energy, determinism, and microscopic reversibility. Since crystallization and evaporation are both allowed and heat is explicitly modeled, any transition between two states may occur in either direction. This gives us a realistic thermodynamics: When started from a low entropy state (e.g., with an empty heat bath), entropy increases and the system approaches a state of detailed balance, or thermodynamic equilibrium. Since we realistically model thermodynamic variables—local heat flow and the creation of entropy—we can do more than study simulated structures: We have a laboratory for studying nonequilibrium thermodynamic behavior of growing crystals. For a detailed discussion of the thermodynamics, the temperature, and the evolution of the growth morphology see D'Souza and Margolus [3].

Until now we have been discussing a closed two-dimensional system with periodic boundary conditions. If we modify the boundary conditions of the heat bath lattice from being periodic to open, we essentially place the heat bath in

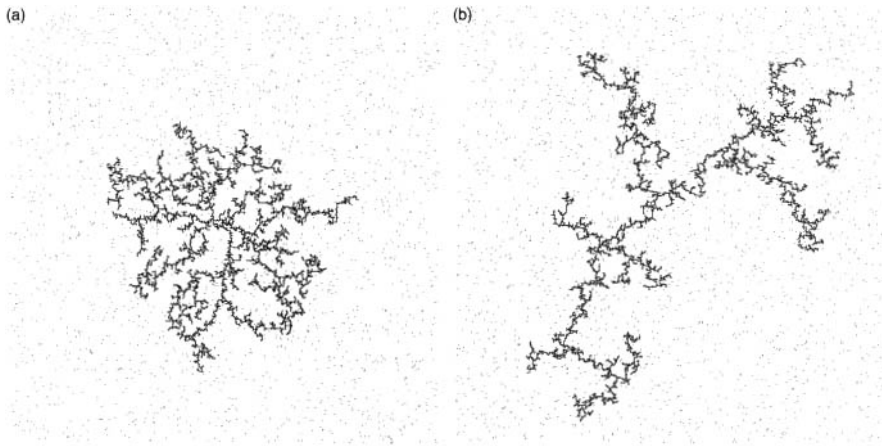

FIGURE 1 (a) An RA cluster of approximately 8270 particles, pictured at the time $t = \tau_T$, which is when the heat bath and the gas-aggregate system first reached the same temperature. Note the grey dots represent the diffusing gas particles. (b) An RA cluster with the same number of particles pictured at time $t = 80\tau_T$. The fractal dimension for this cluster ($d_f = 1.63 \pm 0.02$) has apparently reached the asymptotic value, and is equivalent to the fractal dimension for a quenched branched polymer.

contact with a reservoir at zero temperature and release the heat particles into this reservoir once they reach the edges of the heat bath. We can control the relative speed with which this happens through independent control of the heat and gas diffusion lengths and thus control the effective temperature in which the aggregate grows. The relative tunability of the two diffusion fields allows us to observe a rich variety of growth structures ranging from uniform growth, through invasion percolation, to classical DLA growth.

3 COMPUTATION IN REVERSIBLE SYSTEMS

Computation was long considered to be an inherently dissipative process, requiring the "decision of a two-way alternative and elementary transmittal of one unit of information" (von Neumann as quoted in Bennett [2]). A quantitative understanding of the mechanism of dissipation came with Landauer's work on erasing a bit of information [7]. Erasure requires the transfer of information from computational to other degrees of freedom and normally ultimately to thermal degrees of freedom. The lower bound on the heat produced by erasing one bit is $kT \ln 2$. More than a decade later Bennett showed that erasure is not necessary: Computation can in principle be performed with no dissipation (i.e., no loss of information) [2]. Bennett's proof was based on an abstract Turing machine. He

suggested RNA transcription (in the limit where the rate of transcription approaches zero), as a possible example of a dissipationless digital process. Interest in physical models of computation began around this same time, focusing on prototype Brownian motion computers such as Bennett's RNA model [2, 6].

After yet another decade, Fredkin introduced the concept of conservative logic and introduced a universal conservative logic gate [4]. His goal was to formulate laws of computation more like the laws of microscopic physics, with particular emphasis on microscopic reversibility and exact conservation laws. Conservative logic gates are reversible, and the total number of one's on the wires of a closed system is conserved. The output of each gate is a permutation of its inputs. The Fredkin gate is a three-input, three-output conservative logic gate which implements a conditional swap. A schematic is shown in figure 2. The standard logical primitives ("and," "not," and "fanout") can easily be built out of a Fredkin gate by supplying some constant inputs. The existence of a universal, reversible logic gate means it is possible to implement any digital computation out of these gates without ever erasing information. Moreover it makes reversible computation and circuit design look very similar to conventional computation and circuit design (but using different primitive logic elements).

Fredkin was also interested in embedding his conservative logic computations into realistic physical systems. In Fredkin's Billiard Ball Model (BBM) of computation [4], finite-diameter moving balls are the signals. Collisions between balls implement logic gates. The trajectories of the balls are the wires. Strategically placed mirrors reflect the balls, implementing bends in the wires. The initial positions of the mirrors and the balls must be carefully chosen to allow for precise control and synchronization of signals. The BBM Cellular Automaton (BBMCA) is a universal, reversible CA modeled after the BBM [8].

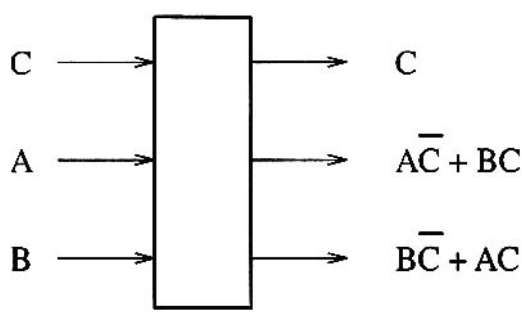

FIGURE 2 The Fredkin gate: a conservative logic gate which performs a conditional swap. If the value of the signal C is true, signals A and B are interchanged; otherwise A and B go straight through. This is a reversible operation which conserves ones: the number of ones entering and leaving the gate is the same.

It uses pairs of particles as the "balls," with the spacing between particles corresponding to the finite diameter of the balls.

4 COMPUTATION IN THE RA MODEL

The scheme proposed for computation in the RA model is similar to that for the Billiard Ball Model. They both use conserved particles as signals and timing and synchronization are crucial. The conditional crystallization events in the RA model are sufficiently complicated that we expect *a priori* that the model is capable of computation. We can show this capability if we allow ourselves to explicitly specify the initial state of all the microscopic degrees of freedom of the system. We must specify the initial positions of all of the gas, crystal, and heat particles, the positions of the "mirrors" which control the diffusion (as discussed below), and the parameters controlling the motion of these mirrors (as discussed in section 4.3.6).

We will first show how to implement wires and delays, and how to route signals. Using these elements and the RA model interaction we show the structure of a simple one-time-use logic gate, and thus prove the system is capable of computing combinatorial logic functions. However, we are interested in more general circuits: those with reusable gates, feedback, and memory elements. To this end we introduce a more complicated dyadic signaling implementation and describe a reusable universal logic gate. We then discuss issues of interconnection and signal crossing. With these in hand, we can build a simulator for any digital logic computer. We demonstrate the construction technique for a simple example circuit.

4.1 SIGNAL ROUTING AND DELAY

In the RA model the gas and heat particles undergo pseudorandom walks, implemented using a lattice gas transport algorithm. For each particle species (gas and heat), there are two transport channels moving in opposite directions along one of the principle lattice directions. At consecutive updates of the system, we alternate between lattice directions (i.e., transport is along the $+\hat{x}$ and $-\hat{x}$ directions on odd time steps and along the $+\hat{y}$ and $-\hat{y}$ directions on even time steps). This scheme can be extended to arbitrary dimensions by introducing additional substeps along each additional lattice direction. A particle remaining in one channel exclusively will follow a diagonal path through the lattice as shown in figure 3(a).

To simulate diffusion we cause the particle to switch between the two channels at random. We include a pseudorandom number field: a "random" binary variable at each site at each time $\eta(\vec{x}, t)$. If $\eta(\vec{x}, t) = 1$, the particle switches channels. At the end of each update we spatially permute the η values in a deterministic and invertible manner so as to have fresh random bits at each site,

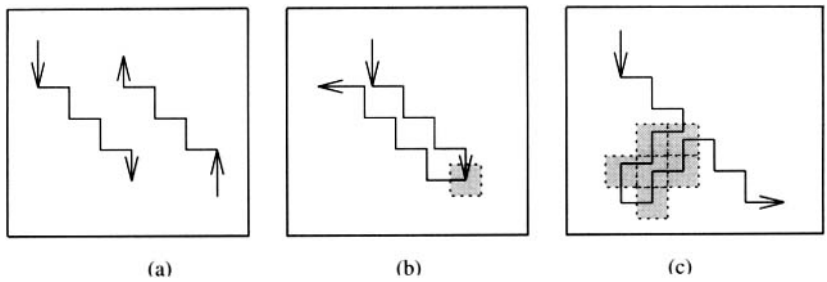

FIGURE 3 (a) Streaming in channel 1 and channel 2. (b) A mirror. (c) A delay loop of eight time steps.

while maintaining constant the probability that $\eta = 1$. The permutation must be deterministic, so we can invert the dynamics when running the model in reverse. This allows us to recover the data used to make switching decisions so we can unswitch the particles and invert the "random" walks. If we use the identity permutation, the η values remain fixed and so does the particle motion (which depends on these values).

If the pseudorandom bit fields are initially filled randomly or pseudorandomly with ones and zeros, the gas and heat particles switch between channels in an unbiased manner and simulate large scale diffusion (c.f. D'Souza and Margolus [3] for a quantitative discussion of diffusion coefficients and a comparison of theoretical and empirical results). If, instead, the $\eta(\vec{x}, t)$ bits and their motion are precisely controlled, the one bits act as deliberately placed mirrors, switching the gas and heat particles between transport channels at determined locations. An example with fixed mirrors is shown in figure 3(b). Between encounters with mirrors, the particles "stream" along a given channel uninterrupted. The gas and heat particles are controlled by separate η bits, so the gas and heat particles are reflected by separate mirrors.

Timing and synchronization are crucial to our logic scheme. To adjust timing, we can use delay loops. A delay loop can be constructed from a collection of mirrors, placed to implement a sequence of reflections. Using the transport algorithm described above, each particle takes a step in the horizontal and then the vertical directions. Since it takes at least four steps for a particle to return to its original location, delays must be a multiple of four. Figure 3(c) shows a delay loop of eight time steps.

4.2 UNIVERSALITY: A SIMPLE GATE

Consider a simple gate, as shown in figure 4. If we count only gas and heat particles entering and exiting at A, B, C, and D (i.e., not counting crystal particles), this gate conserves particle number. The shaded squares represent

crystal, and positions of the mirrors are implied by the particle paths. Signal A must precede B in time. This gate has the following truth table:

A	B	C	D
gas	gas	heat	heat
gas	heat	heat	gas
heat	gas	gas	heat
heat	heat	gas	heat

If we take the signal convention that a gas particle represents one logical truth value, and a heat particle the other, then clearly this gate is universal. Say, for instance, gas represents true and heat represents false. Then $C = \bar{A}$, and $D = A\bar{B}$. So, if we can build arbitrary networks of gates such as this, then we can build arbitrary logic circuits.

There is a subtlety here, however. This gate is universal, yet it is not reusable. So, in fact, we may only build combinatorial logic circuits, and not those with feedback. If we wish to simulate the operation of a universal Turing machine, we must "unroll" the operation of the TM. That is, we simulate the time evolution of the machine's state by computing each state combinatorially from the previous state. This spreads the time progress of the computation out spatially, requiring more logic levels for each step we wish the machine to execute. So with a polynomial amount of space, we may simulate the machine's action for a polynomial number of steps.

We can go further, though. We wish to simulate normal reusable digital logic, so that we can build arbitrary logic circuits, with feedback, memory, etc. To this end, we propose a scheme for reusing gates which utilizes matched pairs of crystallization and evaporation events. It is a dyadic signaling scheme in which pairs of particles, appropriately delayed, are routed through the same gate so as to clean up after the computation (i.e., remove the state from the gate) leaving the gate ready for reuse.

4.3 A REUSABLE GATE, GATE INTERCONNECTION, AND CIRCUITS

The gate we choose to implement is the "switch gate": a two-input, three-output conservative logic gate which is universal [4]. It has a control input, which we will call B, and a switched input, which we will call A. The B input passes through the gate unchanged. The A input is routed to one of two outputs, conditional on the state of B. A schematic is shown in figure 5.

4.3.1 Summary of the Gate Implementation. Our first consideration in implementing such a gate is that the gate must be reusable, or stateless, as mentioned above. To realize this, we adopt a dyadic signaling convention, in which the presence of a "one" is represented not by arrival of a single particle, but by a gas particle

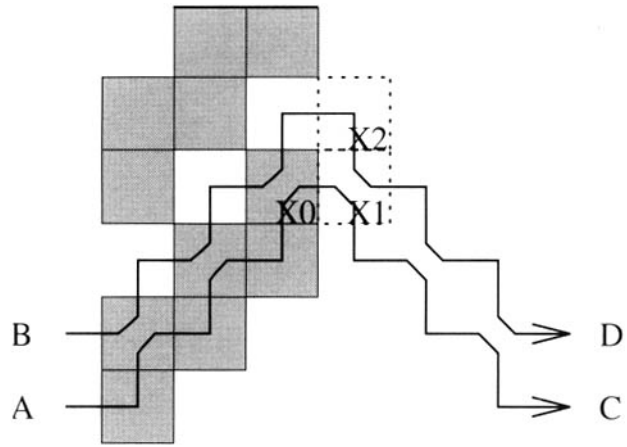

FIGURE 4 A simple, non-reusable, universal logic gate. Crystal is shown by shaded squares. Signal paths are shown as wiggly arrows. Mirror locations are implied by the paths. A and B enter as shown, with A preceding B in time. If A is gas, it crystallizes at X1, releasing heat. If A is heat, it evaporates X0, becoming gas. If A has crystallized at X1, then if B is gas it crystallizes at X2, yielding heat, and if B is heat, it re-evaporates X1, yielding gas. If A has evaporated X0 and B is gas, B re-crystallizes X0 yielding heat, and if B is heat, it does not interact and remains heat. Recall both gas and heat particles diffuse freely over the crystal.

followed four time steps later by a heat particle. This allows us to use crystallization events to implement interactions without leaving permanent changes in the structure of the gate: If a gas particle enters an input and leaves a crystal bit behind, then the corresponding heat particle follows, cleaning up the crystallization event, and leaving the gate in its original state.

We implement a switch gate in the RA model as follows. We first set up an initial condition with a few crystal bits forming a simple aggregate, with only one potential crystallization site, X1. A second site, X2, will become a potential crystallization site if and only if X1 becomes occupied by crystal. There are input paths for signals A and B. The path for signal B is routed through X1, and that for A through X2.

The action of the gate is as follows: We route the gas particle of B through X1, and delay the heat particle of B for now. If B is true (that is, the gas and heat portions of B are indeed present), the gas crystallizes at X1, and X2 now becomes a potential crystallization site. If B is false, nothing happens and X2 is not a potential crystallization site. Now we arrange, by appropriate choice of delays, for both particles of A to pass through X2 while X1 is (in the $B = 1$ case) occupied by crystal. If B is false, A passes through X2 unchanged (i.e., with its heat portion following its gas portion). If B is true, the gas portion of A

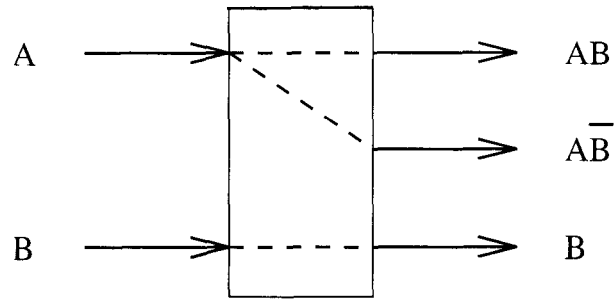

FIGURE 5 The schematic diagram of a switch gate. If the input B is true, the signal A exits the top output. If B is false, A exits the middle output.

crystallizes at X2 yielding heat, and the heat portion of A evaporates the crystal at X2, yielding gas again. Thus, B being true essentially reverses the order of gas and heat particles exiting X2. When both the gas and heat portions of A have had a chance to interact at X2, we send the heat portion of the B signal through X1. If B is a one, the heat signal encounters the crystal particle at X1 and evaporates it, thus restoring the crystal aggregate to its original state and leaving the gate ready for reuse. Note if B is false nothing happens, which also leaves the gate ready for reuse.

The particles for A exit X2 in the same direction regardless of the state of B, but in different temporal orders. To obtain the two outputs $A\bar{B}$ and AB on different spatial paths, we use η fields which change with time to switch the gas and heat particles onto two different paths according to their timing. We then delay the heat on the AB output path, to conform with our dyadic signaling convention that heat follows gas. A schematic diagram of this interdependence of events is shown in figure 6.

4.3.2 Interconnect and Parity. There are some subtleties involved in being able to route any output of any gate to any input of any other gate. In particular, there is a parity defined on particles: Since they are constrained to move alternately vertically and horizontally, we may draw a checkerboard on our lattice and separate the particles into those starting at time zero on a red square ("red" particles), and those starting on a black square ("black" particles).[1] Since the first move is horizontal for all particles, any red particle will move horizontally off its red square, and will always move horizontally off any red square. Similarly any black particle will always move horizontally off a black square. These types of particles cannot be interconverted, and a path followed by one type cannot be followed by the other. Each gate input follows a specific path, thus requiring a

[1]Note, this discussion is independent of the checkerboard updating scheme discussed in section 2.

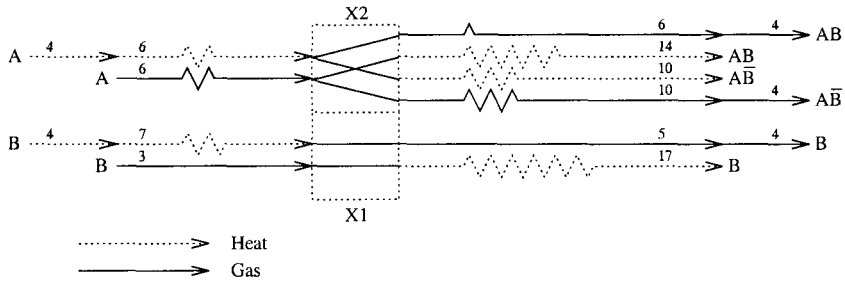

FIGURE 6 A schematic representation of the switch gate. Arrows indicate signal paths for heat and gas particles, with delays annotated. Signals A and B enter simultaneously from the left at time zero. The heat particle for each signal trails the gas particle by four time steps. If B is true, X1 is occupied from time three through time 11. In this case, A crystallizes and evaporates from site X2 at times six and ten, respectively. The signal paths are indicated by the top two lines in X2. If B is false, X1 is never occupied. In this case, both A and its cleanup signal pass through without interacting, and exit at the appropriate times on a different path. The signal paths in this case are indicated by the bottom two lines in X2. Output delays are chosen to place the signals at the gate outputs at a time independent of input values and signal path. Note that all paths have an identical length of 20.

particular color particle. Likewise, each gate output produces a particular color particle. So if we had a gate output producing a black particle and another gate's input expecting a red particle, there would be no way of directly connecting that input to that output. To solve this, we constrain all our inputs and outputs to be on black squares of the checkerboard.

Even with all inputs and outputs on black squares, we have a synchronization problem: Our gates require synchronous signal arrivals, so sometimes we need to delay individual signals to satisfy this requirement. Note delay loops are only available in multiples of four time steps (see fig. 3). Consider any two input signals to our circuit. If they are ever to interact in any gate, they must not only start on the same color square, but they must also be routable to each other in a multiple of four time steps; otherwise they will never be able reach the inputs of the gate simultaneously. So we introduce an additional constraint that all inputs to our circuit be routable to each other in multiples of four time steps. This constraint can be met simply by placing all inputs at time zero on a sublattice of a checkerboard: a square sublattice with separation distance two (see fig. 7). All inputs are now separated by paths whose lengths are multiples of four, and in fact particles can travel from one site of the sublattice to an adjacent one in exactly four time steps. This also guarantees that all outputs of our circuit will be routable to each other in multiples of four time steps (since gate departure

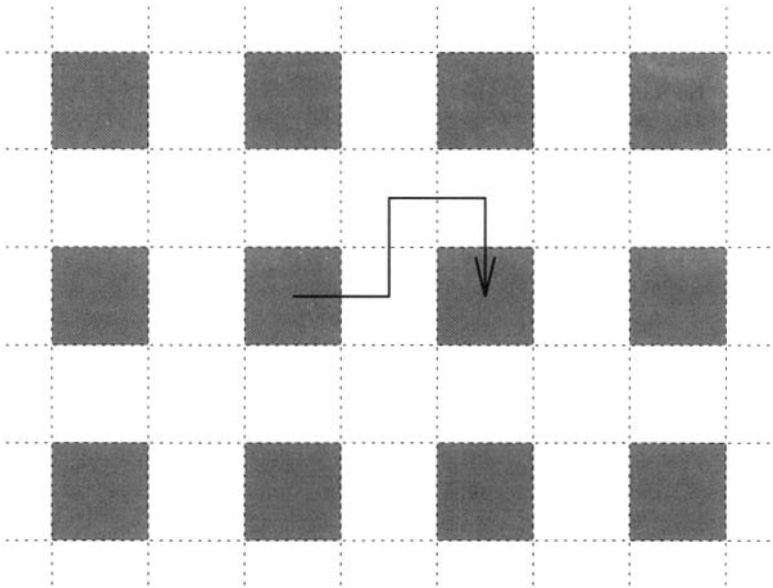

FIGURE 7 The RA lattice, with the circuit input/output sublattice shown in shaded squares. A particle is shown traveling from one sublattice site to an adjacent one in four time steps.

times as well as arrival times are synchronous). Note that by restricting our inputs to be on this sublattice, we have somewhat reduced the usable volume of space. This simplifies our discussion, however, and introduces only a constant factor slowdown and size increase.

4.3.3 Details of the Switch Gate Implementation. The details of the RA model switch gate are shown in figure 8, which depicts the initial state of the lattice and the paths taken by the signals. As mentioned above we set up an initial condition with a few crystal bits forming a simple aggregate. X1 and X2 are indicated by heavy dashed lines. One can see that X1 is initially a potential crystallization site, whereas X2 is a potential crystallization site if and only if X1 is occupied by crystal. The "extra" crystal bits prevent spurious crystallization at undesired locations.

The paths taken by the particles are shown as the wavy lines, and are determined by the placement of mirrors (indicated by the various shades of gray as shown in the legend). At the left are the two input signal paths; at the right are the three outputs. Note that all inputs and outputs lie on the sublattice described in section 4.3.2, thus guaranteeing routability.

One can verify by "walking through" the paths by hand, that the actual delay times are as specified in figure 6. If B is true, X1 is occupied between times three and 11. In this case, if A is also true, a heat particle exits X2 at time six, and a gas at time ten. If B is false and A is true, a *gas* particle exits X2 at time six and a *heat* at time ten. The gas and the heat so produced travel to a toggling mirror, arriving at times 10 and 14, respectively. The toggling mirror begins by deflecting gas in the down direction and heat in the up direction at time zero, and toggling gas and heat directions every four steps thereafter. Hence, if B is true and a heat arrives at time 10 and a gas at time 14, they are both deflected in the up direction, since the mirror toggles in between them at time 12. Conversely, if B is false, the gas arriving at time ten and the heat arriving at time 14 are both deflected *downward*, again since the mirror toggles at time 12. The extra eight steps of heat delay in the upper exit path are to delay the heat so that it follows the gas, in accordance with our signaling convention.

The longest path through the gate requires 16 time steps, so we have imposed extra delay on some signal paths so that the propagation delay through the gate is exactly 16 on all paths. This will be convenient when building more complex circuits, since we will not have to worry about delaying signals to compensate for differences in gate delay. The paths in the figure are longer than the gate propagation time so that both the gas and heat particles can be shown entering and exiting the gate.

4.3.4 The Reverse Switch Gate. In order to build general logic circuits we will need to use switch gates in both the forward and reverse directions [4]. The switch gate used in reverse is a three-input two-output gate which performs the inverse logic operation. The three inputs must of course be appropriately correlated for this to be possible. Since the dynamics of the RA model is reversible, we can build the reverse gate by sending the signals through the outputs of the original gate, but with the opposite signaling convention: heat preceding gas. Converting between forward and reverse signaling conventions is simple: We need only impose an extra delay of eight steps on the gas so that it follows the heat by four time steps. We may convert back to the forward signaling convention by similarly delaying the heat.

4.3.5 Crossing Wires. We showed above that, observing parity constraints in gate placement, any output can be connected to any input. However, we have yet to show that signals can cross—a necessary detail for connecting any output to any input. Consider two signals, one traveling in a diagonal downward and to the right, the other traveling in a diagonal upward and to the right. The first signal will travel purely in one channel (channel 1 in fig. 3(a)), encountering no mirrors. The second signal must flip channels at each time step and thus encounter a mirror at every site. Thus, for these signals to cross there must both be mirrors and be no mirrors at the two lattice sites they will both encounter. A way to implement this is to delay one signal by four time steps, allowing the

first signal to propagate past the relevant two sites, then toggle the mirrors. The second signal now encounters the correct mirror configuration when it occupies the relevant two sites. A "cross gate" so constructed is shown in figure 9. The shaded sites are the ones where the mirrors are toggled. Note that we have delayed the first signal by four time steps after its encounter with the toggling sites, so that both signals leave the cross gate synchronously.

4.3.6 A Sample Circuit. The RA model has been implemented on a special-purpose cellular automata machine, the CAM8 [9]. A detailed description of this implementation can be found in D'Souza and Margolus [3]. For constructing the logic gates we have discussed in this chapter, we use the CAM8 implementation of the RA model, but with the modification that the initial state and the dynamics of the pseudorandom bits are precisely specified. In the original model the permutation of the η bit fields was simply a displacement (shift) in $\hat{x}$ and $\hat{y}$ by a prespecified amount at each update step. Here, where we wish to exercise detailed control over the microscopic dynamics, we choose the displacements more restrictively. We displace the η bit fields for both gas and heat by half the size of the space in the horizontal direction only. Further, this displacement occurs

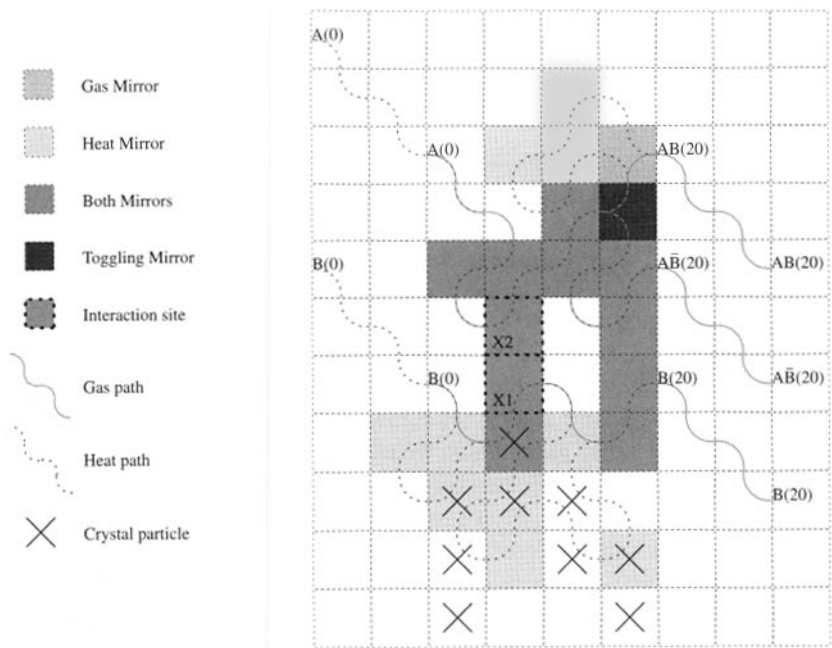

FIGURE 8 A detailed picture of an RA switch gate. Signal entry and exit times are given in parentheses after the signal name.

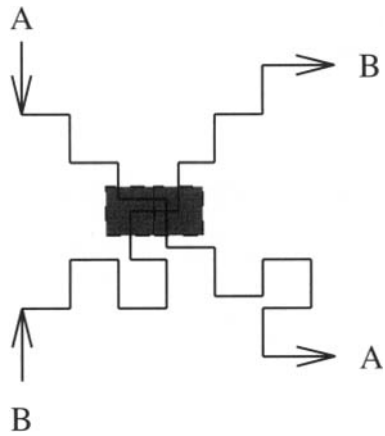

FIGURE 9 A "cross gate." The signal traveling upward is delayed by four time steps while the signal traveling downward passes through the two shaded lattice sites. The mirrors at the shaded sites are then toggled. The upward signal now encounters the correct mirror configuration to pass through as indicated. The first signal is delayed by four steps after passing through the shaded sites so that both signals leave this gate synchronously.

only on every fourth time step. This allows us to implement the toggling mirrors easily. We place all the circuitry and one set of mirrors in the left half of the space and place a second set of mirrors, with the toggling mirrors complemented, in the right half. Note that since the shifts of the pseudorandom bit planes are specified as part of the initial condition, we are still just manipulating our initial condition in order to effect computation in the RA model.

To see the detailed action in the CAM8 simulation, consider a single switch gate. The operation of this gate in three of the four input cases is shown in figure 10. Note that A takes the topmost output path if and only if $B = 1$.

We have shown how to implement a reusable, universal logic gate, how to route and delay signals, and how to let signals cross so that output from any can serve as input to any other gate. Thus, we can build any Boolean logic circuit we wish. As a simple example of connecting gates together we construct an identity circuit from two back to back switch gates, the second running in reverse, followed by a cross gate. The schematic of the circuit is shown in figure 11. The CAM8 implementation is pictured in figure 12. The black squares are the stationary crystal bits. The lightly shaded squares mark the trace of where the signals have traveled (the wires). The signals (heavily shaded squares) exit the gate at the right. Note that in the space between the two switch gates, the gas particles are delayed by eight so the signals enter the reverse gate with the opposite signal

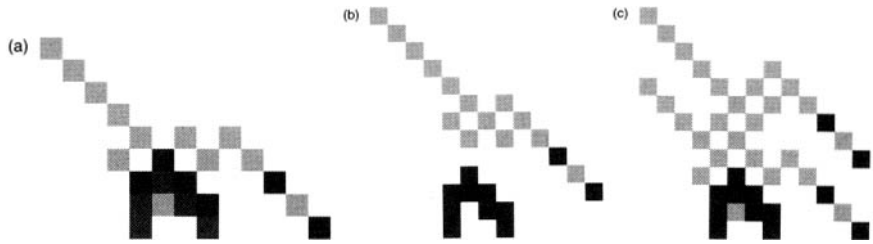

FIGURE 10 Three cases in the operation of a switch gate. (a) $A = 0$, $B = 1$. (b) $A = 1$, $B = 0$. (c) $A = 1$, $B = 1$. Particles are heavily shaded, crystal is black, and paths are lightly shaded. The path shading is only for every alternate time step, to make it simpler to resolve distinct paths by eye. The null case, $A = 0$, $B = 0$ is omitted.

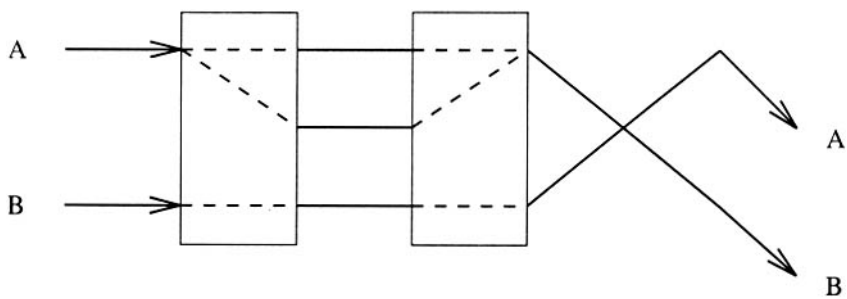

FIGURE 11 A circuit composed of two switch gates—one forward, one reverse—followed by a signal crossover.

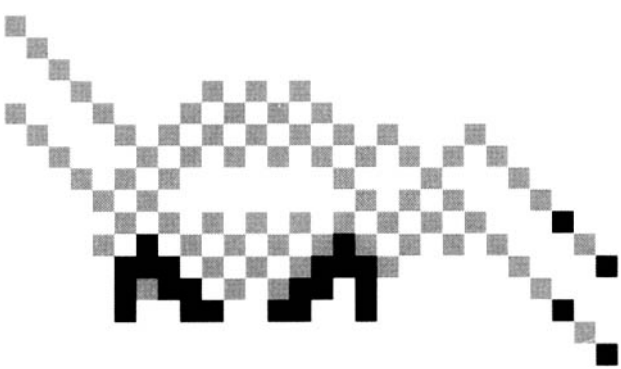

FIGURE 12 A CAM8 implementation of an identity gate (composed of two switch gates back to back) and a crossover. The shaded squares are the wires, visualized every other time step. The signals have exited the circuit at the right.

convention. In order not to clutter the figure we did not reinvert the signals, so they leave with the inverse signal convention of heat first.

5 DISCUSSION AND CONCLUSIONS

5.1 SUMMARY

We have investigated the computational capabilities of a specific physical model of cluster growth, where we have control over all of the microscopic degrees of freedom. The dynamics can compute any sequence of digital logic operations if the system is initialized to a precisely specified state. Thus the RA model can simulate any other digital dynamics.

The moving particles are the logic signals, the paths they take are the wires. Synchronization of signal arrival times at specific sites requires careful layout of mirrors which route and delay signals. By routing particles through specified interaction sites, we can build logic gates from conditional crystallization and evaporation events. These constructs are relatively straightforward to implement, allowing us to build a simple logic gate as shown in section 4.2. This gate is changed by the interaction—it can be used only once. Showing that the RA model can support reusable gates adds complication. We have many degrees of freedom; choosing an appropriate signaling convention is crucial. We choose a dyadic signaling convention which allows us to construct reusable logic gates, as discussed in section 4.3.3.

5.2 COMPUTATION IN REAL PHYSICAL SYSTEMS

We have exhibited computations in a discrete lattice system that depend on exactly synchronous updating and complete control of all microscopic degrees of freedom. Have we abstracted anything useful for computation in real physical systems?

The RA model captures some essential aspects of real crystallization. Perhaps the most relevant to computation is the conditional nature of crystallization (i.e., the presence of nucleation sites only at the perimeter of a growing crystal along with the absence of heat in the local environment). This conditional interaction allows us to build logical primitives that do not depend on exact synchronization.

Computation in the RA model can be accomplished with exact control of the microscopic degrees of freedom. However, in real physical systems, including electronic computers, we typically have control only of the macroscopic degrees of freedom. For the macroscopic dynamics of a system to be universal, the microscopic dynamics must necessarily be universal: If we cannot compute with complete control over the system, we cannot hope to compute with less control. Thus we can consider the construction given in this chapter as a "warmup" for

addressing the "larger" issue of macroscopic universality. The analogy to thermodynamics is straightforward. A gas can do mechanical work moving a piston despite the fact we know nothing of the individual gas particles; We know only aggregate quantities. The question then is, can we get *computational* work out of a system with control only of the macroscopic degrees of freedom? With this type of understanding we might be able to compute with a growing bacterial colony or a growing crystal aggregate, and exert detailed control over the structures produced [1].

A first step in understanding macroscopic computation in the RA model is to test the robustness of the system's ability to compute when it is subjected to an actual stochastic dynamics. For instance, we can study the situation where the particles follow truly random walks or where the interactions are probabilistic, yet we maintain control over the other degrees of freedom. If we replace each single gas and heat particle with a dilute cloud of particles we may be able to conditionally crystallize and uncrystallize with high probability. Directing the macroscopic motion of clouds of particles may require us to add momentum conservation to the RA model, making it even more realistic. Even so restoring the gates and signals to their starting state would require some dissipative cleanup process. This would entail a constant throughput of "power" (i.e., the addition and removal of particles).

ACKNOWLEDGMENTS

This work was supported by DARPA under contract number DABT63-95-C-0130, and by the Merck/MIT Graduate Research Fellowship Program. We wish to thank the organizers of the SFI Constructive CA workshop, which provided inspiration for this work. We also thank Jeremy Zucker for a careful reading of this manuscript.

REFERENCES

[1] Abelson, H., T. F. Knight, Jr., G. J. Sussman, and friends. "Amorphous Computing." MIT AI Laboratory. ⟨`http://swissnet.ai.mit.edu/projects/amorph-new/amorph-new.html`⟩, 1995-present.

[2] Bennett, C. H. "Logical Reversibility of Computation." *IBM J. Res. & Dev.* **17** (1973): 525–532.

[3] D'Souza, R. M., and N. H. Margolus. "A Thermodynamically Reversible Generalization of Diffusion Limited Aggregation." *Phys. Rev. E.* **60(1)** (1999): to appear.

[4] Fredkin, E., and T. Toffoli. "Conservative Logic." *Intl. J. Theor. Phys.* **21(3/4)** (1982): 219–253.

[5] Kaufman, H., A. Vespignani, B. B. Mandelbrot, and L. Woog. "Parallel Diffusion Limited Aggregation." *Phys. Rev. E* **52** (1995): 5602–5609.

[6] Keyes, R. W., and R. Landauer. "Minimal Energy Dissipation in Logic." *IBM J. Res. & Dev.* **14** (1970): 152–157.

[7] Landauer, R. "Irreversibility and Heat Generation in the Computing Process." *IBM J. Res. & Dev.* **3** (1961): 183–191.

[8] Margolus, N. H. "Physics-Like Models of Computation." *Physica D* **10** (1984): 81–95.

[9] Margolus, N. H. "CAM-8: A Computer Architecture Based on Cellular Automata." In *Pattern Formation and Lattice-Gas Automata*, edited by A. Lawniczak and R. Kapral. Providence, RI: American Mathematical Society, 1996.

[10] Moriarty, K., J. Machta, and R. Greenlaw. "Parallel Algorithm and Dynamic Exponent for Diffusion-Limited Aggregation." *Phys. Rev. E* **55(5)** (1997): 6211–6218.

[11] Nagatani, T. "Unsteady Diffusion-Limited Aggregation." *J. Phys. Soc. Jpn.* **61(5)** (1992): 1437–1440.

[12] Voss, R. F. "Multiparticle Fractal Aggregation." *J. Stat. Phys.* **36(5/6)** (1984): 861–872.

[13] Witten, T. A., and L. M. Sander. "Diffusion-Limited Aggregation, a Kinetic Critical Phenomenon." *Phys. Rev. Lett.* **47(19)** (1981): 1400–1403.

Universal Cellular Automata Based on the Collisions of Soft Spheres

Norman Margolus

Fredkin's Billiard Ball Model (BBM) is a continuous classical mechanical model of computation based on the elastic collisions of identical finite-diameter hard spheres. When the BBM is initialized appropriately, the sequence of states that appear at successive integer time steps is equivalent to a discrete digital dynamics.

Here we discuss some models of computation that are based on the elastic collisions of identical finite-diameter *soft* spheres: spheres which are very compressible and hence take an appreciable amount of time to bounce off each other. Because of this extended impact period, these Soft Sphere Models (SSMs) correspond directly to simple lattice gas automata—unlike the fast-impact BBM. Successive time steps of an SSM lattice gas dynamics can be viewed as integer-time snapshots of a continuous physical dynamics with a finite-range soft-potential interaction. We present both two-dimensional and three-dimensional models of universal CAs of this type, and then discuss spatially efficient computation

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

using momentum conserving versions of these models (i.e., without fixed mirrors). Finally, we discuss the interpretation of these models as relativistic and as semiclassical systems, and extensions of these models motivated by these interpretations.

1 INTRODUCTION

Cellular automata (CA) are spatial computations. They imitate the locality and uniformity of physical law in a stylized digital format. The finiteness of the information density and processing rate in a CA dynamics is also physically realistic. These connections with physics have been exploited to construct CA models of spatial processes in Nature and to explore artificial "toy" universes. The discrete and uniform spatial structure of CA computations also makes it possible to "crystallize" them into efficient hardware [17, 21].

Here we will focus on CAs as realistic spatial models of ordinary (non-quantum-coherent) computation. As Fredkin and Banks pointed out [2], we can demonstrate the computing capability of a CA dynamics by showing that certain patterns of bits act like logic gates, like signals, and like wires, and that we can put these pieces together into an initial state that, under the dynamics, exactly simulates the logic circuitry of an ordinary computer. Such a CA dynamics is said to be *computation universal*. A CA may also be universal by being able to simulate the operation of a computer in a less efficient manner—never reusing any logic gates for example. A universal CA that can perform long iterative computations within a fixed volume of space is said to be a *spatially efficient* model of computation.

We would like our CA models of computation to be as realistic as possible. They should accurately reflect important constraints on physical information processing. For this reason, one of the basic properties that we incorporate into our models is the microscopic reversibility of physical dynamics: there is always enough information in the microscopic state of a physical system to determine not only what it will do next, but also exactly what state it was in a moment ago. This means, in particular, that in reversible CAs (as in physics) we can never truly erase any information. This constraint, combined with energy conservation, allows reversible CA systems to accurately model thermodynamic limits on computation [3, 8]. Conversely, reversible CAs are particularly useful for modeling thermodynamic processes in physics [4]. Reversible CA "toy universes" also tend to have long and interesting evolutions [5, 17].

All of the CAs discussed in this chapter fall into a class of CAs called lattice gas automata (LGA), or simply lattice gases. These CAs are particularly well suited to physical modeling. It is very easy to incorporate constraints such as reversibility, energy conservation, and momentum conservation into a lattice gas. Lattice gases are known which, in their large-scale average behavior, reproduce the normal continuum differential equations of hydrodynamics [11, 12]. In a

lattice gas, particles hop around from lattice site to lattice site. These models are of particular interest here because one can imagine that the particles move continuously between lattice sites in between the discrete CA time steps. Using LGAs allows us to add energy and momentum conservation to our computational models, and also to make a direct connection with continuous classical mechanics.

Our discussion begins with the most realistic classical mechanical model of digital computation, Fredkin's Billiard Ball Model [10]. We then describe related classical mechanical models which, unlike the BBM, are isomorphic to simple lattice gases at integer times. In the BBM, computations are constructed out of the elastic collisions of very incompressible spheres. Our new two-dimensional and three-dimensional models are based on elastically colliding spheres that are instead very compressible, and hence take an appreciable amount of time to bounce off each other. The universality of these soft sphere models (SSMs) depends on the finite extent in time of the interaction, rather than its finite extent in space (as in the BBM). This difference allows us to interpret these models as simple LGAs. Using the SSMs, we discuss computation in perfectly momentum-conserving physical systems (cf. Moore and Nordahl [20]), and show that we can compute just as efficiently in the face of this added constraint. The main difficulty here turns out to be reusing signal-routing resources. We then provide an alternative physical interpretation of the SSMs (and of all mass and momentum conserving LGAs) as relativistic systems, and discuss some alternative relativistic SSM models. Finally, we discuss the use of these kinds of models as semiclassical systems which embody realistic quantum limits on classical computation.

2 FREDKIN'S BILLIARD BALL MODEL

In figure 1, we summarize Edward Fredkin's classical mechanical model of computation, the Billiard Ball Model. His basic insight is that a location where balls may or may not collide acts like a logic gate: we get a ball coming out at certain places only if another ball didn't knock it away! If the balls are used as signals, with the presence of a ball representing a logical "1" and the absence a logical "0," then a place where signals intersect acts as a logic gate, with different logic functions of the inputs coming out at different places. Figure 1(a) illustrates the idea in more detail. For this to work right, we need synchronized streams of data, with evenly spaced time slots in which a 1 (ball) or 0 (no ball) may appear. When two 1s impinge on the collision "gate," they behave as shown in the figure, and they come out along the paths labeled **AB**. If a 1 comes in at **A** but the corresponding slot at **B** is empty, then that 1 makes it through to the path labeled $\mathbf{A\bar{B}}$ (**A** and not **B**). If sequences of such gates can be connected together with appropriate delays, the set of logic functions that appear at the outputs in figure 1(a) is sufficient to build any computer.

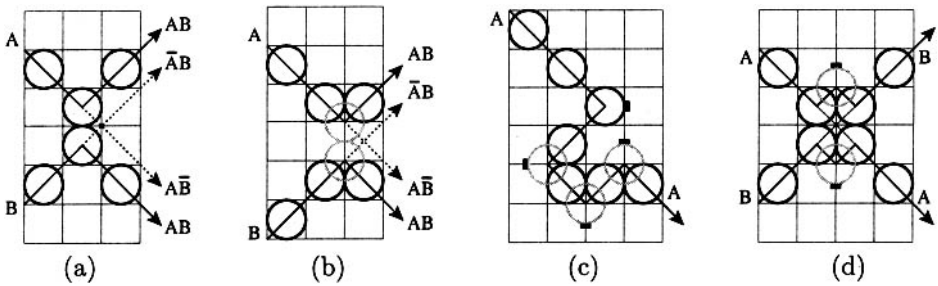

FIGURE 1 The Billiard Ball Model. Balls are always found at integer coordinates at integer times. (a) A collision that does logic. Two balls are initially moving toward each other to the right. Successive columns catch the balls at successive integer times. The dotted lines indicate paths the balls would have taken if only one or the other had come in (i.e., no collision). (b) Balls can collide at half-integer times (gray). (c) Billiard balls are routed and delayed by carefully placed mirrors as needed to connect logic-gate collisions together. Collisions with mirrors can occur at either integer or half-integer times. (d) Using mirrors, we can make two signal paths cross as if the signals pass right through each other.

In order to guarantee composability of these logic gates, we constrain the initial state of the system. All balls are identical and are started at integer coordinates, with the unit of distance taken to be the diameter of the balls. This spacing is indicated in the figure by showing balls centered in the squares of a grid. All balls move at the same speed in one of four directions: up-right, up-left, down-right, or down-left. The unit of time is chosen so that at integer times, all freely moving balls are again found at integer coordinates. We arrange things so that balls always collide at right angles, as in figure 1(a). Such a collision leaves the colliding balls on the grid at the next integer time. Figure 1(b) shows another allowed collision, in which the balls collide at half-integer times (shown in gray) but are still found on the grid at integer times. The signals leaving one collision gate are routed to other gates using fixed mirrors, as shown in figure 1(c). The mirrors are strategically placed so that balls are always found on the grid at integer times. Since zeros are represented by no balls (i.e., gaps in streams of balls), zeros are routed just as effectively by mirrors as the balls themselves are. Finally, in figure 1(d), we show how two signal streams are made to cross without interacting—this is needed to allow wires to cross in our logic diagrams. In the collision shown, if two balls come in, one each at **A** and **B**, then two balls come out on the same paths and with the same timing as they would have if they had simply passed straight through. Needless to say, if one of the input paths has no ball, a ball on the other path just goes straight through. And if both inputs have no ball, we will certainly not get any balls at the outputs, so the zeros go straight through as well.

Clearly any computation that is done using the BBM is reversible, since if we were to simultaneously and exactly reverse the velocities of all balls, they would exactly retrace their paths, and either meet and collide or not at each intersection, exactly as they did going forward. Even if we don't actually reverse the velocities, we know that there is enough information in the present state to recover any earlier state, simply because we *could* reverse the dynamics. Thus, we have a classical mechanical system which, viewed at integer time steps, performs a discrete reversible digital process.

The digital character of this model depends on more than just starting all balls at integer coordinates. We need to be careful, for example, not to wire two outputs together. This would result in head-on collisions which would not leave the balls on the grid at integer times! Miswired logic circuits, in which we use a collision gate backward with the four inputs improperly correlated, would also spoil the digital character of the model. Rather than depending on correct logic design to assure the applicability of the digital interpretation, we can imagine that our balls have an interaction potential that causes them to pass through each other without interacting in all cases that would cause problems. This is a bit strange, but it does conserve energy and momentum and is reversible. Up to four balls, one traveling in each direction, can then occupy the same grid cell as they pass through each other. We can also associate the mirror information with the grid cells, thus completing the BBM as a CA model. Unfortunately this is a rather complicated CA with a rather large neighborhood.

The complexity of the BBM as a CA rule can be attributed to the nonlocality of the hard-sphere interaction. Although the BBM interaction can be softened—with the grid correspondingly adjusted—this model depends fundamentally upon information interacting at a finite distance. A very simple CA model based on the BBM, the BBMCA [13, 17] avoids this nonlocality by modeling the front and back edges of each ball, and using a sequence of interactions between edge particles to simulate a billiard ball collision. This results in a reversible CA with just a four-bit neighborhood (including all mirror information!), but this model gives up exact momentum conservation, even in simulating the collision of two billiard balls.

In addition to making the BBMCA less physical, this loss of conservation makes BBMCA logic circuits harder to synchronize than the original BBM. In the BBM, if we start a column of signals out, all moving up-right or down-right, then they all have the same horizontal component of momentum. If all the mirrors they encounter are horizontal mirrors, this component remains invariant as we pass the signals through any desired sequence of collision "gates." We don't have to worry about synchronizing signals—they all remain in a single column moving uniformly to the right. In the BBMCA, in contrast, simulated balls are delayed whenever they collide with anything. In a BBMCA circuit with only horizontal mirrors (or even without any mirrors), the horizontal component of momentum is not conserved, the center of mass does not move with constant horizontal velocity, and appropriate delays must be inserted in order to bring together

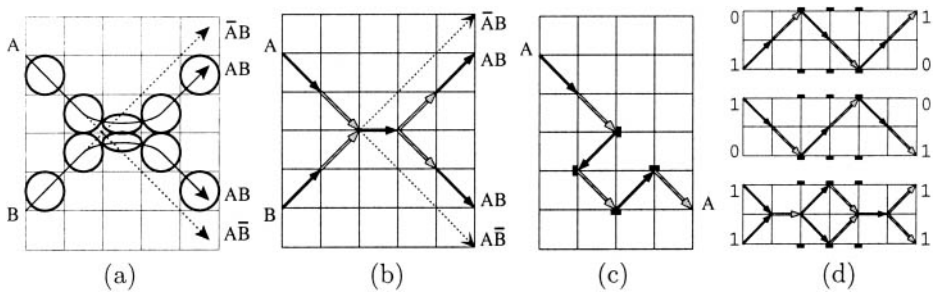

FIGURE 2 A soft sphere model of computation. (a) A BBM-like collision using very compressible balls. The springiness of the balls is chosen so that after the collision, the balls are again at integer sites at integer times. The logic is just like the BBM, but the paths are deflected inward, rather than outward. (b) Arrows show the velocities of balls at integer times. During the collision, we consider the pair to be a single mass, and draw a single arrow. (c) We can route and delay signals using mirrors. (d) We can make signals cross.

signals that have gotten out of step. The BBMCA has energy conservation, but not momentum conservation.

It turns out that it is easy to make a model which is very similar to the BBM, which has the same kind of momentum conservation as the BBM, and which corresponds isomorphically to a simple CA rule.

3 A SOFT SPHERE MODEL

Suppose we set things up exactly as we did for the BBM, with balls on a grid, moving so that they stay on the grid, but we change the collision, making the balls very compressible. In figure 2(a), we illustrate the elastic collision of two balls in the resulting Soft Sphere Model (SSM). If the springiness of the balls is just right (i.e., we choose an appropriate interaction potential), then the balls find themselves back on the grid after the collision. If only one or the other ball comes in, they go straight through. Notice that the output paths are labeled exactly as in the BBM model, except that the **AB** paths are deflected inwards rather than outwards (cf. Appendix to Margous [13]). If we add BBM-style hard-collisions with mirrors,[1] then this model can compute in the same manner as the BBM, with the same kind of momentum conservation aiding synchronization.

[1] All of the 90° turns that we use in our SSM circuits can also be achieved by soft mirrors placed at slightly different locations.

In figure 2(b), we have drawn an arrow in each grid cell corresponding to the velocity of the center of a ball at an integer time. The pair of colliding balls is taken to be a single particle, and we also draw an arrow at its center. We've colored the arrows alternately gray and black, corresponding to successive positions of an incoming pair of logic values. We can now interpret the arrows as describing the dynamics of a simple lattice gas, with the sites of the lattice taken to be the corners of the cells of the grid.

In a lattice gas, we alternately move particles and let them interact. In this example, at each lattice site we have room for up to eight particles (1s): we can have one particle moving up-right, one down-right, one up-left, one down-left, one right, one left, one up, and one down. In the movement step, all up-right particles are simultaneously moved one site up and one site to the right, while all down-right particles are moved down and to the right, etc. After all particles have been moved, we let the particles that have landed at each lattice site interact—the interaction at each lattice site is independent of all other lattice sites.

In the lattice gas pictured in figure 2(b), we see on the left particles coming in on paths **A** and **B** that are entering two lattice sites (black arrows) and the resulting data that leaves those sites (gray arrows). Our inferred rule is that single diagonal particles that enter a lattice site come out in the same direction they came in. At the next step, these gray arrows represent two particles entering a single lattice site. Our inferred rule is that when two diagonal particles collide at right angles, they turn into a single particle moving in the direction of the net momentum. Now a horizontal black particle enters the next lattice site, and our rule is that it turns back into two diagonal particles. If only one particle had come in, along either **A** or **B**, it would have followed our "single diagonal particles go straight" rule, and so single particles would follow the dotted path in the figure. Thus our lattice gas exactly duplicates the behavior of the SSM at integer times.

From figure 2(c) we can infer the rule with the addition of mirrors. Along with particles at each lattice site, we allow the possibility of one of two kinds of mirrors—horizontal mirrors and vertical mirrors. If a single particle enters a lattice site occupied only by a mirror, then it is deflected as shown in the diagram. Signal crossover takes more mirrors than in the BBM (fig. 2(d)). Our lattice gas rule is summarized in figure 3(a). For each case shown, 90° rotations of the state shown on the left turn into the same rotation of the state shown on the right. In all other cases, particles go straight. This is a simple reversible rule, and (except in the presence of mirrors) it exactly conserves momentum. We will discuss a version of this model later without mirrors, in which momentum is always conserved.

The relationship between the SSM of figure 3(a) and a lattice gas can also be obtained by simply shrinking the size of the SSM balls without changing the grid spacing. With the right time constant for the two-ball impact process, tiny particles would follow the paths indicated in figure 3(b), interacting at grid-corner lattice sites at integer times. The BBM cannot be turned into a lattice

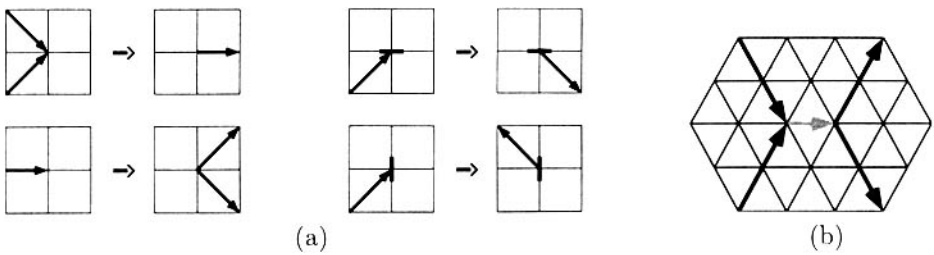

FIGURE 3 (a) A simple lattice gas rule captures the dynamics of the soft sphere collision. Two particles colliding at right angles turn into a single new particle of twice the mass for one step, which then turns back into two particles. A mirror deflects a particle through 90°. In all other cases, particles go straight. (b) A soft sphere collision on a triangular lattice.

gas in this manner, because the BBM depends upon the finite extent of the interaction in space, rather than in time.

Notice that in establishing an isomorphism between the integer-time dynamics of this SSM and a simple lattice gas, we have added the constraint to the SSM that we cannot place mirrors at half-integer coordinates, as we did in order to route signals around in the BBM model in figure 2. This means, in particular, that we can't delay a signal by one time unit—as the arrangement of mirrors in figure 3(c) would if the spacing between all mirrors were halved. This doesn't impair the universality of the model, however, since we can easily guarantee that all signal paths have an even length. To do this, we simply design our SSM circuits with mirrors at half-integer positions and then rescale the circuits by an even factor (four is convenient). Then all mirrors land at integer coordinates. The separation of outputs in the collision of figure 3(b) can be rescaled by a factor of four by adding two mirrors to cause the two **AB** outputs to immediately collide a second time (as in the bottom image of fig. 3(d)). We will revisit this issue when we discuss mirrorless models in section 5.

4 OTHER SOFT SPHERE MODELS

In figure 3(b), we show a mass- and momentum-conserving SSM collision on a triangular lattice, which corresponds to a reversible lattice gas model of computation in exactly the same manner as discussed above. Similarly, we can construct SSMs in three dimensional. In figure 4(a), we see a mass and momentum conserving SSM collision using the face diagonals of the cubes that make up our three-dimensional grid. The resulting particle (gray) carries one bit of information about which of two possible planes the face diagonals that created it resided in. In a corresponding diagram showing colliding spheres (a three-dimensional

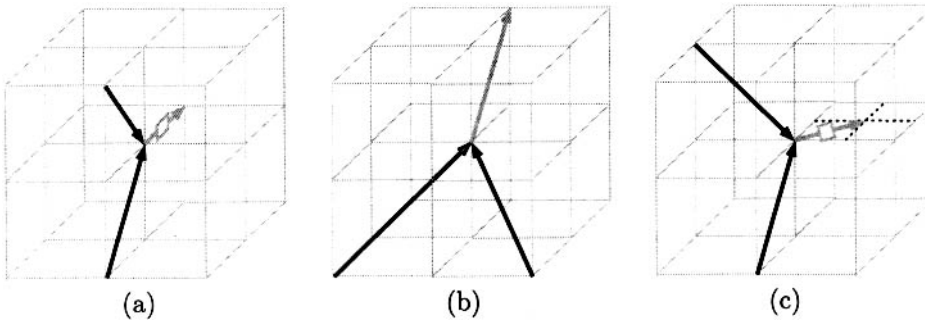

FIGURE 4 Three-dimensional Soft Sphere Models. (a) Collisions using cube edges and cube-face diagonals. Each edge particle carries one bit of information about which of two planes the diagonal particles that created it were in. (b) Collisions using face and body diagonals. Two body-diagonal particles collide only if they are both coplanar with a face diagonal. The resulting face-diagonal particle doesn't carry any extra planar information, since there is a unique pair of body-diagonal particles that could have produced it. (c) Collisions using only face diagonals, with two speeds. If particles are confined to a single plane, this is equivalent to the triangular lattice model of figure 3(b). Again the slower particle must carry an extra bit of collision-plane information.

version of fig. 3(a)), we would see that this information is carried by the plane along which the spheres are compressed. This model is universal within a single plane of the three-dimensional space, since it is just the two-dimensional square-lattice SSM discussed above. To allow signals to get out of a single plane, mirrors can be applied to diagonal particles to deflect them onto cube-face diagonals outside of their original plane.

A slightly simpler three-dimensional scheme is shown in figure 4(b). Here we only use body and face diagonals, and body diagonals only collide when they are coplanar with a face diagonal. Since each face diagonal can only come from one pair of body diagonals, no collision-plane information is carried by face-diagonal particles. For mirrors, we can restrict ourselves to reflecting each body diagonal into one of the three directions that it could have been deflected into by a collision with another body diagonal. This is an interesting restriction, because it means that we can potentially make a momentum-conserving version of this model without mirrors, using only signals to deflect signals.

Finally, the scheme shown in figure 4(c) uses only face diagonals, with the heavier particle traveling half as fast as the particles that collide to produce it. As in figure 4(a), the slower particle carries a bit of collision-plane information. To accommodate the slower particles, the lattice needs to be twice as fine as in figures 4(a) and 4(b), but we've only shown one intermediate lattice site for clarity. Noting that three coplanar face diagonals of a cube form an equilateral triangle, we see that this model, for particles restricted to a single plane, is ex-

actly equivalent to the triangular-lattice model pictured in figure 3(b). As in the model pictured in figure 4(b), the deflection directions that can be obtained from particle-particle collisions are sufficient for three-dimensional routing, and so this model is also a candidate for mirrorless momentum-conserving computation in three dimensions.

5 MOMENTUM CONSERVING MODELS

A rather unphysical property of the BBM, as well as of the related soft sphere models we have constructed, is the use of immovable mirrors. If the mirrors moved even a little bit, they would spoil the digital nature of these models. To be perfectly immovable, as we demand, these mirrors must be infinitely massive, which is not very realistic. In this section, we will discuss SSM gases which compute without using mirrors, and hence are perfectly momentum conserving.

The issue of computation universality in momentum-conserving lattice gases was discussed in Moore and Nordahl [20], where it was shown that some two-dimensional LGAs of physical interest can compute any logical function. This paper did not, however, address the issue of whether such LGAs can be spatially efficient models of computation, reusing spatial resources as ordinary computers do. There is also a new question about the generation of entropy (undesired information) which arises in the context of reversible momentum conserving computation models, and which we will address. With mirrors, any reversible function can be computed in the SSM (or BBM) without leaving any intermediate results in the computer's memory [10]. Is this still true without mirrors, where even the routing of signals requires an interaction with other signals? We will demonstrate mirrorless momentum-conserving SSMs that are just as efficient spatially as an SSM *with* mirrors, and that don't need to generate any more entropy than an SSM with mirrors. In the process we will illustrate some of the general physical issues involved in efficiently routing signals without mirrors.

5.1 REFLECTIONS WITHOUT MIRRORS

We begin our discussion by replacing a fixed mirror with a constant stream of particles (ones), aimed at the position where we want a signal reflected. This is illustrated in figure 5(a). Here we show the two-dimensional square-lattice SSM of figure 3(a), with a signal $\mathbf{A}$ being deflected by the constant stream. Along with the desired reflection of $\mathbf{A}$, we also produce two undesired copies of $\mathbf{A}$ (one of them complemented). This suggests that perhaps every bend in every signal path will continuously generate undesired information that will have to be removed from the computer.

Figure 5(b) shows a more promising deflection. The only thing that has changed is that we have brought in $\bar{\mathbf{A}}$ along with $\mathbf{A}$, and so we now get a 1 coming out the bottom regardless of what the value of $\mathbf{A}$ was. Thus signals

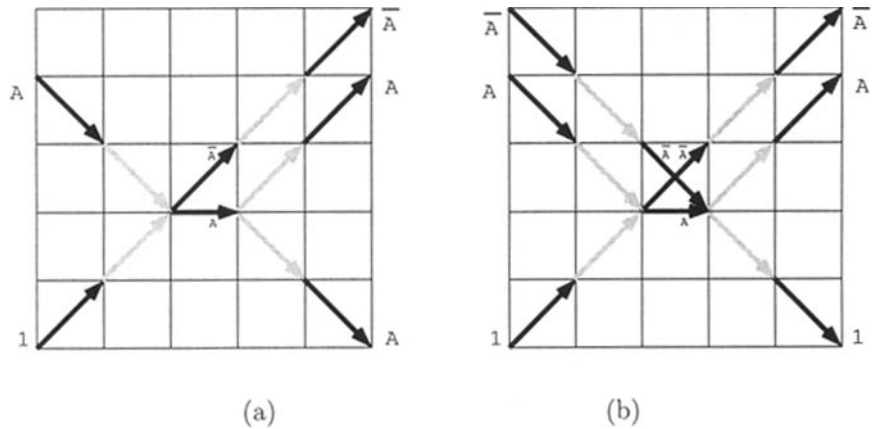

FIGURE 5 Using streams of balls as mirrors. (a) A stream of 1s (balls) diverts a signal **A**, but also makes two copies of the signal. (b) If dual-rail (complementary) signaling is used, signals can be cleanly reflected.

that are represented in complementary form (so-called "dual rail" signals) can be deflected cleanly. This makes sense, since each signal now carries one unit of momentum regardless of its value, and so the change of momentum in the deflecting mirror stream can now also be independent of the signal value.

5.2 SIGNAL CROSSOVER

An important use of mirrors in the BBM and in SSMs is to allow signals to cross each other without interacting. While signals can also be made to cross by leaving regular gaps in signal streams and delaying one signal stream relative to the other, this technique requires the use of mirrors to insert compensating delays that resynchronize streams. If we're using streams of balls to act as mirrors, we have a problem when these mirror streams have to cross signals, or even each other.

We can deal with this problem by extending the noninteracting portion of our dynamics. In order to make our SSMs unconditionally digital, we already require that balls pass through each other when too many try to pile up in one place. Thus it seems natural to also use the presence of extra balls to force signals to cross. The simplest way to do this is to add a rest particle to the model—a particle that doesn't move. At a site "marked" by a rest particle, signals will simply pass through each other. This is mass and momentum conserving, and is perfectly compatible with continuous classical mechanics. Notice that we don't actually have to change our SSM collision rule to include this extra noninteracting case, since we gave the rule in the form, "these cases interact, and in all other

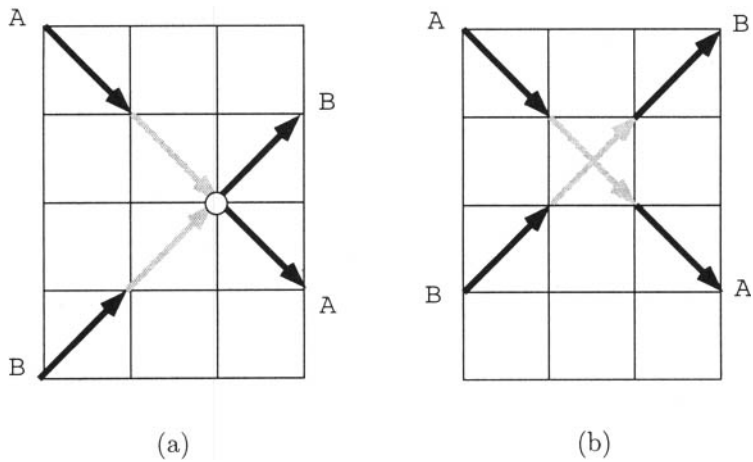

FIGURE 6 Signals that cross. (a) The circle indicates a rest particle. Two signals cross at a rest particle without interacting. (b) Signals can also cross between lattice sites, where no interaction is possible.

cases particles go straight." Figure 6(a) shows an example of two signal paths crossing over a rest particle (indicated by a circle).

Figure 6(b) shows an example of a signal crossover that doesn't require a rest particle in the lattice gas version of the SSM. Since LGA particles only interact at lattice sites, which are the corners of the grid, two signals that cross as in this figure cannot interact. Such a crossover occurs in figure 5(b), for example. Without the LGA lattice to indicate that no interaction can take place at this site, this crossover would also require a rest particle. To keep the LGA and the continuous versions of the model equivalent, we will consider a rest particle to be present implicitly wherever signals cross between lattice sites.

5.3 SPATIALLY EFFICIENT COMPUTATION

With the addition of rest particles to indicate signal crossover, we can use the messy deflection of figure 5(a) to build reusable circuitry and so perform spatially efficient computation. The paths of the incoming "mirror streams" can cross whatever signals are in their way to get to the point where they are needed, and then the extra undesired "garbage" output streams can be led away by allowing them to cross any signals that are in their way. Since every mirror stream (which brings in energy but no information) and every garbage stream (which carries away both energy and entropy) crosses a surface that encloses the circuit, the number of such streams that we can have is limited by the area of the enclosing surface. Meanwhile, the number of circuit elements (and

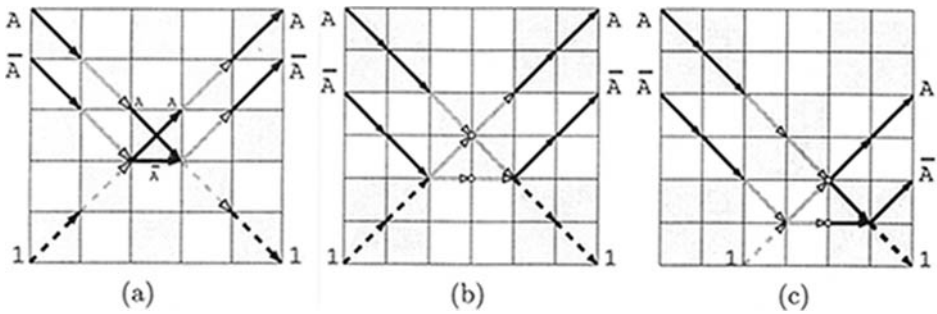

FIGURE 7 A signal routing constraint. (a) When signal pairs are deflected by a stream of 1s, each component of the pair remains on the same checkerboard region of the space. (b) If we spread the signals so that pairs are twice as far apart, we can also rescale the mirror collision. (c) After rescaling, we can move the "mirror" to what would have originally been a half-integer position, and so avoid this constraint.

hence also the demand for mirror and garbage streams) grows as the volume of the circuit [3, 8, 10]. This is the familiar surface to volume ratio problem that limits heat removal in ordinary heat-generating physical systems: the rate of heat generation is proportional to the volume, but the rate of heat removal is only proportional to the surface area. We have the same kind of problem if we try to bring free energy (i.e., energy without information) into a volume.

Using dual-rail signaling, we've seen that we have neat collisions available that don't corrupt the deflecting mirror streams. We do not, however, avoid the surface to volume problem unless these clean mirror streams can be reused: otherwise each reflection involves bringing in a mirror stream all the way from outside of the circuit, using it once, and then sending the reflected mirror stream all the way out of the circuit. Thus if we can't reuse mirror streams, the maximum number of circuit elements we can put into a volume of space grows like *the surface area rather than like the volume!* We will show that (at least in two-dimensional) mirror streams can be reused, and consequently momentum conservation doesn't impair the spatial efficiency of computations.

5.4 SIGNAL ROUTING

Even though we can reflect dual-rail signals and make them cross, we still have a problem with routing signals (actually two problems, but we'll discuss the second problem when we confront it). Figure 7(a) illustrates a problem that stems from not being able to reflect signals at half-integer locations. Every reflection leaves the top **A** signal on the dark checkerboard we've drawn—it can't connect to an input on the light checkerboard. We can fix this by rescaling the circuit, spreading all signals twice as far apart (fig. 7(b)). Now the implicit crossover in

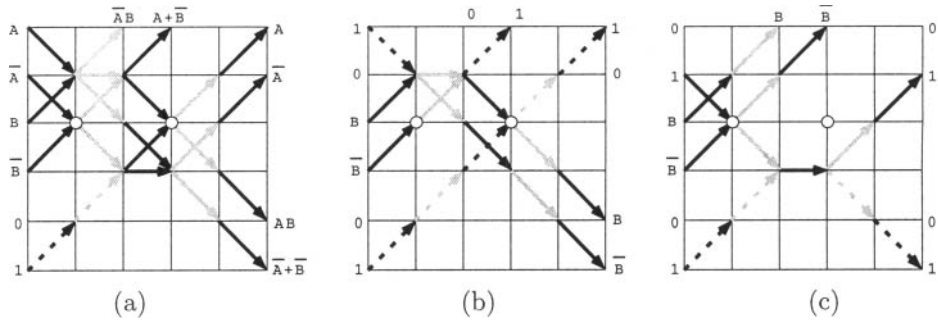

FIGURE 8 A switch gate using dual-rail signaling. (a) The general case. The **A** signal either deflects **B** and $\bar{\mathbf{B}}$ or not, doing most of the work. We've highlighted the constant stream of ones by using dotted lines. (b) The case **A**=1. **B** and $\bar{\mathbf{B}}$ are reflected down, and the one is reflected the opposite way. (c) The case **A**=0. There is no interaction with **B** or $\bar{\mathbf{B}}$, and they go straight.

the middle of figure 7(a) must be made explicit. Notice also that the horizontal particle must be stretched—it too goes straight in the presence of a rest particle. Now we can move the reflection to a position that was formerly a half-integer location (fig. 7(c)), and the **A** signal is deflected onto the white checkerboard.

5.5 DUAL-RAIL LOGIC

We've seen that dual-rail signals can be cleanly routed. In order to use such signals for computation, we need to be able to build logic with dual-rail inputs and outputs. We will now see that if we let two dual-rail signals collide, we can form a switch gate [10], as shown in figure 8(a). The switch gate is a universal logic element that leaves the control input **A** unchanged, and routes the controlled input **B** to one of two places, depending on the value of **A**. Since each dual-rail signal contains a 1, and since all collisions conserve the number of 1s, all dual-rail logic gates need an equal number of inputs and outputs. Thus our three-output switch gate needs an extra input which is a dual-rail constant of 0.

The switch gate (fig. 8(a)) is based on a reflection of the type shown in figure 5(b). If **A**=1 (fig. 8(b)), the **B** and $\bar{\mathbf{B}}$ pair are reflected downward; if **A**=0 there is no reflection and they go straight. The $\bar{\mathbf{A}}$ signal reflects off the constant-one input as in figure 5(a), to regenerate the **A** and $\bar{\mathbf{A}}$ outputs. Notice that if a rest particle were added in figure 8(a) at the intersection of the **A** and **B** signals, the switch would be stuck in the *off* position: **B** and $\bar{\mathbf{B}}$ would always go straight through, and **A** and $\bar{\mathbf{A}}$ would get reflected by the constant one, and come out in their normal position.

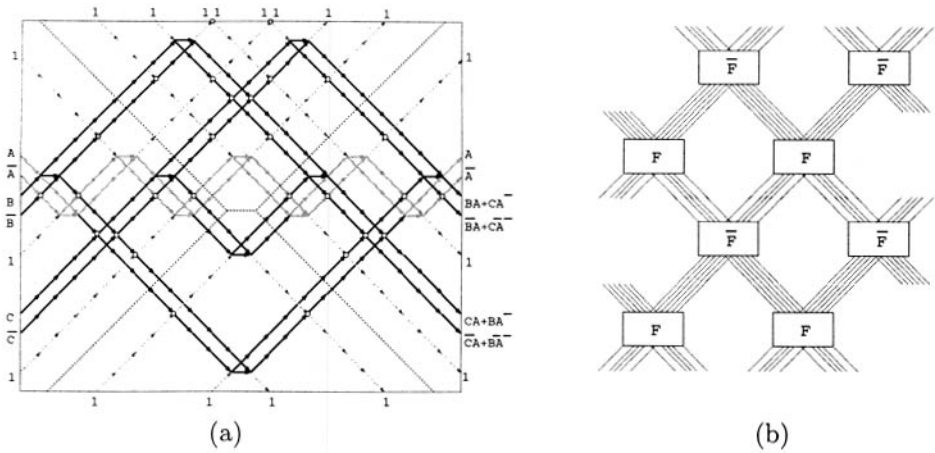

FIGURE 9 (a) A Fredkin gate. We construct a Fredkin gate out of four switch gates, two used forward, and two backward. Constant 1s are drawn in lightly using dotted arrows. The path of the control signal **A** is shown in solid gray. If we added constant streams of 1s along the four paths drawn as dotted lines without arrows, then the constant streams would be symmetrical about diagonal axes. (b) Because of the diagonal symmetry of this Fredkin gate construction, we can make an array of them, as indicated here, and reuse the constant streams of 1s that act as signal mirrors. The upside down Fredkin gates are also Fredkin gates, but with the sense of the control inverted.

5.6 A FREDKIN GATE

In order to see that momentum conservation doesn't impair the spatial efficiency of SSM computation, we first illustrate the issues involved by showing how mirror streams can be reused in an array of Fredkin gates [10].

A Fredkin gate has three inputs and three outputs. The **A** input, called the control, appears unchanged as the **A** output. The other two inputs either appear unchanged at corresponding outputs (if **A**=1), or appear interchanged at the corresponding outputs (if **A**=0). We construct a Fredkin gate out of four switch gates, as shown in figure 9(a). The first two switch gates are used forward, the last two switch gates are used backward (i.e., flipped about a vertical axis). The control input **A** is colored in solid gray, and we see it wend its way through the four switch gates. Constant 1s are shown using dotted gray arrows. In the case **A**=0, all four switch gates pass their controlled signals straight through, and so **B** and **C** interchange positions in the output. In the case **A**=1, all four switch gates deflect their controlled signals, and so **B** and **C** come out in the same positions they went in.

Now notice the bilateral symmetry of the Fredkin gate implementation. We can make use of this symmetry in constructing an array of Fredkin gates that

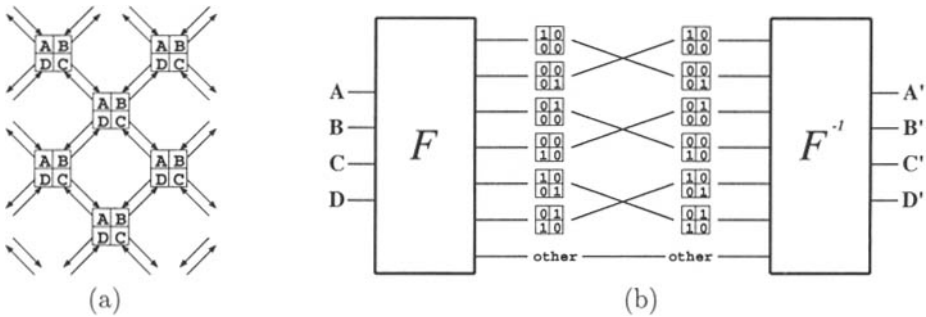

FIGURE 10 Emulating the BBMCA using an SSM. (a) The BBMCA can be implemented as a two-dimensional array of identical blocks of logic, each of which processes four bits at a time. The bits that are grouped together in one step go to four different diagonally adjacent blocks in the next step. (b) We construct a circuit out of switch gates to implement the BBMCA logic block. The first half of the circuit (F) produces a set of outputs that are each one only if the four BBMCA bits have some particular pattern. The second half (F^{-1}) is the mirror image of the first. In between, the cases that interchange are wired to each other.

reuse the constant 1 signals. If we add an extra stream of constant 1s along the four paths drawn as arrowless dotted lines (making these lie on the lattice involves rescaling the circuit), then the set of constant streams coming in or leaving along each of the four diagonal directions is symmetric about some axis. This means that we can make a regular array of Fredkin gates and upside-down Fredkin gates, as is indicated in figure 9(b), with the constants all lining up. These constants are passed back and forth between adjacent Fredkin gates, and so don't have to be supplied from outside of the array. Since an upside-down Fredkin gate is still a Fredkin gate, but with the sense of the control inverted, we have shown that constant streams of ones can be reused in a regular array of logic.

We still have not routed the inputs and outputs to the Fredkin gates, and so we have another set of associated mirror streams that need to be reused. The obvious approach is to create a regular pattern of interconnection, thus allowing us to again solve the problem globally by solving it locally. But a regular pattern of interconnected logic elements that can implement universal computation is just a universal CA: we should simply implement a universal CA that doesn't have momentum conservation!

5.7 IMPLEMENTING THE BBMCA

The BBMCA is a simple reversible CA based on the BBM, with fixed mirrors [13, 15, 17]. It can be implemented as a regular array of identical logic

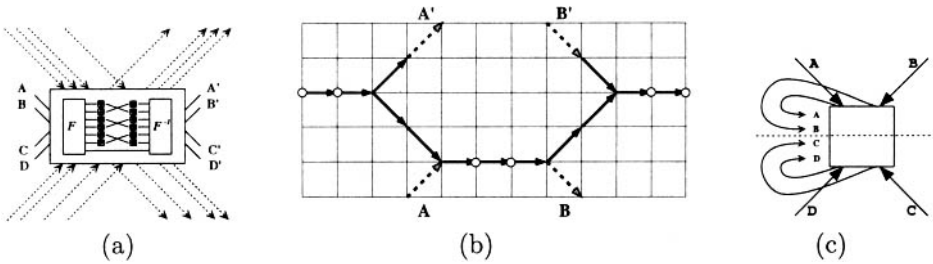

FIGURE 11 Symmetrizing signal paths so that adjacent BBMCA logic blocks can share their mirror constants. (a) The BBMCA block circuit is bilaterally symmetric, with an equal number of constants flowing in or out along each of the four diagonal directions. (b) Symmetric pairs of constant 1s can be shifted vertically in order to align the "mirror streams" so that the blocks can be arrayed. (c) The wiring of the four BBMCA signal inputs (and outputs) to each block is also bilaterally symmetric, so the same alignment techniques should apply.

blocks, each of which takes four bits of input, and produces four bits of output (fig. 10(a)). Each logic block exchanges one bit of data with each of the four blocks that are diagonally adjacent. The four bits of input can be thought of as a pattern of data in a 2×2 region of the lattice, and the four outputs are the next state for this region. According to the BBMCA rule, certain patterns are turned into each other, while other patterns are left unchanged. This rule can be implemented by a small number of switch gates, as is indicated schematically in figure 10(b). We first implement a demultiplexer F, which produces a value of 1 at a given output if and only if a corresponding 2×2 pattern appears in the inputs. Patterns that don't change under the BBMCA dynamics only produce 1s in the outputs labeled "other." The demultiplexer is a combinational circuit (i.e., one without feedback). The inverse circuit F^{-1} is simply the mirror image of F, obtained by reflecting F about a vertical axis. In between F and F^{-1} we wire together the cases that need to interchange. This gives us a bilaterally symmetric circuit which implements the BBMCA logic block in the same manner that our circuit of figure 9(a) implemented the Fredkin gate. Note that the overall circuit is its own inverse, as any bilaterally symmetric combinational SSM circuit must be.

Now we would like to connect these logic blocks in a uniform array. We will first consider the issue of sharing the mirror streams associated with the individual logic blocks, and then the issue of sharing the mirror streams associated with interconnecting the four inputs and outputs. In figure 11(a) we see a schematic representation of our BBMCA block. It is a combinational circuit, with signals flowing from left to right. The number of signal streams flowing in along one diagonal direction is equal to the number flowing out along the same direction—

this is true overall because it's true of every collision! In particular, since the four inputs and outputs are already matched in the diagram, the mirror streams must also be matched—there are an equal number of streams of constant 1s coming in and out along each direction. The input streams will not, however, in general be aligned with the output streams. If we can align these, then we can make a regular array of these blocks, with mirror-stream outputs of one connected to the mirror-stream inputs of the next.

In figure 11(b) we show how to align streams of ones. Due to the bilateral symmetry of the BBMCA circuit, every incoming stream that we would like to shift up or down on one side is matched by an outgoing stream that needs to be shifted identically on the other side. Thus we will shift streams in pairs. To understand the diagram, suppose that **A** and **B** are constant streams of ones, with **B** going into a circuit below the diagram, and **A** coming out of it. Now suppose that we would like to raise **A** and **B** to the positions labeled **A′** and **B′**. If a constant stream of horizontal particles is provided midway in between the two vertical positions, then we can accomplish this as shown. The constant horizontal stream splits at the first position without a rest particle. It provides the shifted **A′** signal, and a matching stream of ones collides with the original **A** signal. The resulting horizontal stream is routed straight across until it reaches **B**, where an incoming stream of ones is needed. Here it splits, with the extra stream of ones colliding with the incoming **B′** signal to restore the original horizontal stream of ones, which can be reused in the next block of the array of circuit blocks to perform the same function. The net effect is that the mirror streams **A** and **B** coming out of and into a circuit have been replaced with new streams that are shifted vertically. By reserving some fraction of the horizontal channels for horizontal constants that stream across the whole array, and reserving some channels for horizontal constants that connect pairs of streams being raised, we can adjust the positions of the mirror streams as needed. Note that a mirror pair can be raised by several smaller shifts rather than just one large shift, in case there are conflicts in the use of horizontal constants. Exactly the same arrangement can be used to lower **A′** and **B′** going into and out of a circuit above the diagram. If we flip the diagram over, we see how to handle pairs of streams going in the opposite directions.

Now we note that the wiring of the four signal inputs and outputs in our BBMCA array also has bilateral symmetry, about a horizontal axis (fig. 11(c)). Thus it seems that we should be able to apply the same technique to align the mirror streams associated with this routing, in order to complete our construction. But there is a problem.

5.8 SIGNAL ROUTING REVISITED

So far, we have only constructed circuits without feedback—all signal flow has been left to right. Because of the 90° rotational symmetry of the SSM, we might expect that feedback isn't a problem. When we decided to use dual-rail signaling,

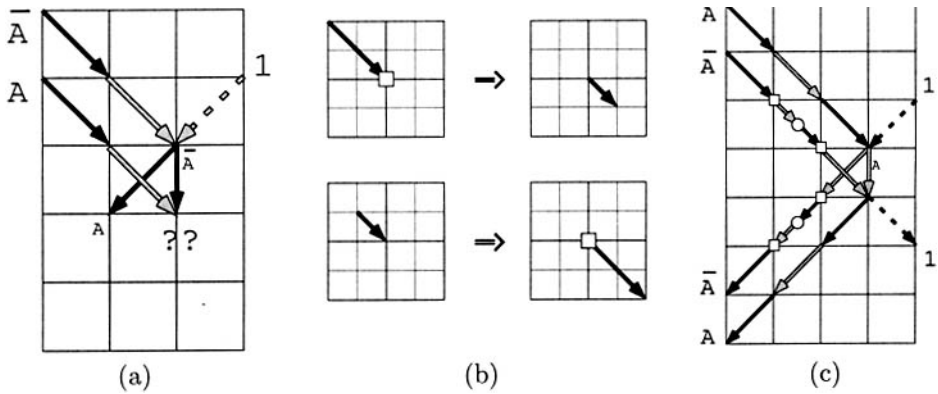

FIGURE 12 Reflecting signals back. (a) Dual rail pairs that are synchronized by column only reflect correctly off "horizontal mirrors." If we try to bounce them off a "vertical mirror," signals from different times interact. (b) This can be fixed by providing a way to slow down a signal. The rule for adding slower particles involves refining the lattice to admit a half-speed double-mass diagonal particle, and adding a single-mass rest particle (square block in the diagram). When a soft sphere collides with an equally massive sphere at rest, the first slows down as the second speeds up (giving a net speed of 1/2), and then the sphere that was at rest proceeds. (c) Using our "non-interaction" rest particle to extend the lifetime of the half-speed diagonal particle, we can change a column-synchronized pair into a row-synchronized pair, and then back.

however, we broke this symmetry. The timing of the dual-rail signal pairs is aligned vertically and not horizontally. In figure 12(a), we see the problem that we encounter when we try to reflect a right-moving signal back to the left. A signal that passed the input position labeled $\mathbf{A}$ at an even time step collides with an unrelated signal that passed input $\bar{\mathbf{A}}$ at an odd time step. These two signals need to be complements of each other in order to reconstitute the reflecting mirror stream. Thus we only know how to reflect signals vertically, not horizontally!

We will discuss two ways of fixing this problem. Both involve using additional collisions in the SSM. The first method we describe is more complicated, since it adds additional particles and velocities to the model, but is more obvious. The idea is that we can resynchronize dual-rail pairs by delaying one signal. We do this by introducing an interacting rest particle (distinct from our previously introduced noninteracting rest particle) with the same mass as our diagonally moving particles. The picture we have in mind is that if there is an interacting rest particle in the path of a diagonally moving particle, then we can have a collision in which the moving particle ends up stationary, and the stationary particle ends up moving. During the finite interval while the particles are colliding, the mass is doubled and so (from momentum conservation) the velocity is halved. By picking

the head-on impact interval appropriately, the new stationary particle can be deposited on a lattice site, so that the model remains digital. This is illustrated in figure 12(b). Here the square block indicates the interacting rest particle. This is picked up in a collision with a diagonal-moving particle to produce a half-speed double-mass particle indicated by the short arrow. Note that adding this delay collision requires us to make our lattice twice as fine to accommodate the slower diagonal particles. It adds five new particle states to our model (four directions for the slow particle, and an interacting rest particle). The model remains, however, physically "realistic" and momentum conserving.

Figure 12(c) illustrates the use of this delay to reflect a rightgoing signal leftward. We insert a delay in the $\bar{\mathbf{A}}$ path both before the mirror-stream collision and afterward, in order to turn the plane of synchronization 180°, turning it 90° at a time. Notice that we use a noninteracting rest particle (round) to extend the lifetime of the half-speed diagonal particle.

In addition to complicating the model, this delay technique adds an extra complication to showing that momentum conservation doesn't impair spatial efficiency. Signals are delayed by picking up and later depositing a rest particle. In order to reuse circuitry, we must include a mechanism for putting the rest particle back where it started before the next signal comes through. Since we can pick this particle up from any direction, this should be possible by using a succession of constant streams coming from various directions, but these streams must also be reused. We won't try to show that this can be done here—we will pursue an easier course in the next section.

It would be simpler if the moving particle was deposited at the same position that the particle it hit came from, so that no cleanup was needed. Unfortunately, this necessarily results in no delay. Since the velocity of the center of mass of the two particles is constant, if we end up with a stationary particle where we started, the other particle must exactly take the place of the one that stopped.

5.9 A SIMPLER EXTENSION

We can complete our construction without adding any extra particles or velocities to the model. Instead, we simply add some cases to the SSM in which our normal soft-sphere collisions happen even when there are extra particles nearby. The cases we will need are shown in figure 13. In this diagram, we show each forward collision in one particular orientation—collisions in other orientations and the corresponding inverse collisions also apply. The first case is the SSM collision with nothing else around. The second case is a forward and backward collision simultaneously—this will let us bounce signals back the way they came. The third case has at least two spectators, and possibly a third (indicated by a dotted arrow). The collision proceeds normally, and all spectators pass straight through. This case will allow us to separate forward and backward moving signals. As usual, all other cases go straight. In particular, we will depend for the first time

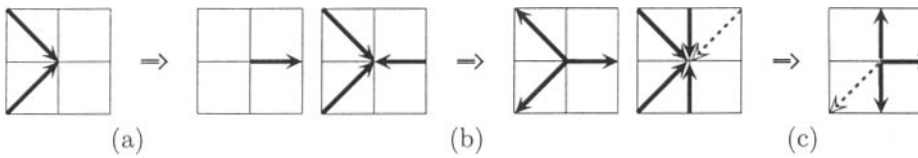

FIGURE 13 A two-dimensional square-lattice SSM. Particles go straight unless they interact. One sample orientation is shown for each interacting case. The inverse cases also apply. (a) Basic collision. (b) Same collision and its inverse operating in two opposite directions simultaneously. (c) Same collision as (a), but with "spectator particles" present. The dotted-arrow particle may or may not be present, and may come from below instead (i.e., flipped orientation). Spectators go straight.

on head-on colliding particles going straight. We have not used any head-on collisions in our circuits thus far, and so we are free to define their behavior here.

Figure 14(a) shows how two complementary sets of dual-rail signals can be reflected back the way they came. We show the signals up to the moment where they come to a point where they collide with the two constant streams. In the case where $\mathbf{A}=1$, we have four diagonal signals colliding at a point, and so everything goes straight through. In particular, the constant streams have particles going in both directions (passing through each other), and the signal particles go back up the $\mathbf{A}$ paths without interacting with oncoming signals. In the case where $\mathbf{A}=0$, we use our new "both directions" collision, which again sends all particles back the way they came. Thus we have succeeded in reversing the direction of a signal stream.

Figure 14(b) shows a mirror with all signals moving from left to right. We've added in vertical constant streams in two places, which don't affect the operation of the "mirror." These paths have a continual stream of particles in both the up and down directions, and so these particles all go straight (head-on collisions). In figure 14(c), we've just shown signals coming into this mirror backward (with the forward paths drawn in lightly). This mirror doesn't reflect these backward-going signals, and so they go straight through. The vertical constants were needed to break the symmetry, so that it's unambiguous which signals should interact. This separation uses the extra spectator-particle cases added to our rule in figure 13(c). As we will discuss, in a triangular-lattice SSM the separation at mirrors doesn't require any vertical constants at the mirrors (see section 5.10).

Finally, figure 15 shows how we can arrange to always have two complementary dual-rail pairs collide whenever we need to send a signal backward. Figure 15(a) shows an SSM circuit with some number of dual-rail pairs. In each pair, the signals are synchronized vertically, with the uncomplemented signal on top. Figure 15(b) shows the same gate flipped vertically. The collisions that implement the circuit work perfectly well upside down, but both the inputs and the outputs are complemented by this inversion. For example, in figure 15(c),

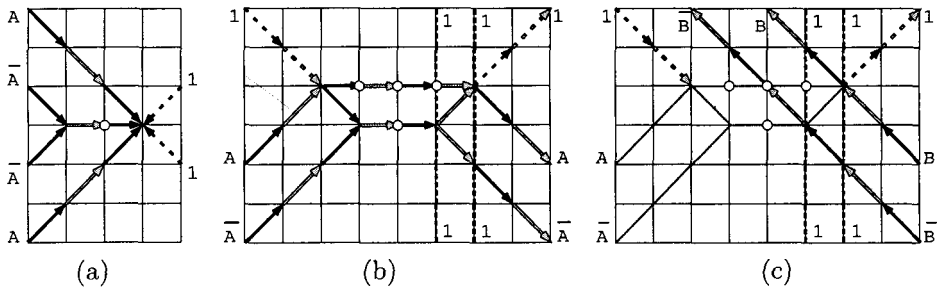

FIGURE 14 A way to reflect signals back, without refining the lattice or adding extra particles. (a) If we have two dual-rail signal pairs (one the complement of the other), then they can be bounced straight backward along the same paths they came in on by bringing all the signals to one point at which two mirror signals impinge. In either case (**A**=0 and **A**=1), the constant 1s that reflect these signals are also reversed along their paths. (b) The thick vertical dotted line segments indicate constants of one that are moving in both directions at the indicated locations. This is otherwise a normal reflection of a signal—the extra vertical streams don't interfere with the operation of the "mirror." (c) If a backward propagating signal comes in from the right (**B** and $\bar{\mathbf{B}}$), then it is not reflected by this forward mirror—such a mirror separates the backward moving stream from the forward stream.

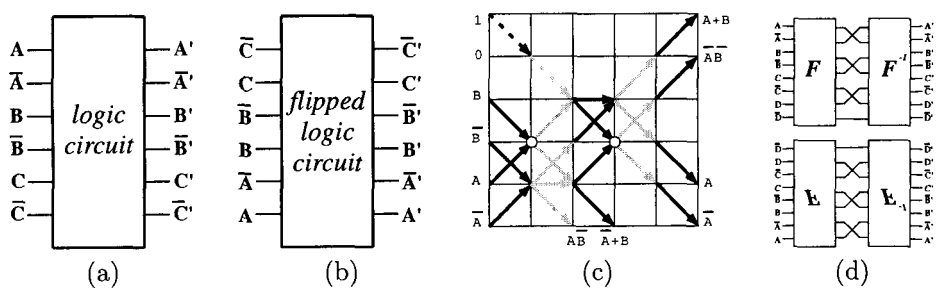

FIGURE 15 DeMorgan inversion. (a) A logic circuit with dual-rail inputs. In each input pair, the complemented signal lies below the uncomplemented one. (b) If this circuit is flipped vertically, the operation of the circuit is unchanged, but it operates upon inputs that are complemented (according to our conventions) and produces outputs that are also complemented. (c) The switch gate of figure 8(a), flipped vertically. Inputs and outputs have been relabeled to call the top signal in each dual-rail pair "uncomplemented." (d) The BBMCA logic circuit of figure 10(b) has been mirrored vertically to produce a vertically symmetric circuit which has complementary pairs of dual-rail pairs.

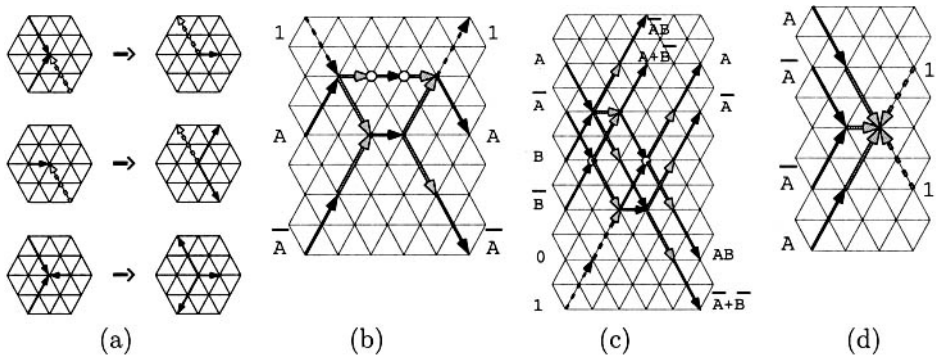

FIGURE 16 An SSM gas on the triangular lattice which allows signal feedback. (a) Two speed-2 particles collide and turn into a speed-1 particle with twice the mass. This decays back into two speed-2 particles. If an extra "spectator" speed-2 particle comes in as shown with a dotted arrow (or the flip of these cases), it passes straight through. Collisions can happen both forward and backward simultaneously. In all other cases, particles go straight. (b) Constants act as mirrors for dual-rail signals. (c) This is a switch gate. Other combinational circuits from the square-lattice SSM can be similarly stretched vertically to fit onto the triangular lattice. (d) The third collision case in the rule makes signals bounce back the way they came. Backward-going signals will separate at a mirror such as is shown in (b).

we have turned a switch gate upside down. If we relabel inputs and outputs in conventional order, then we see that this gate performs a logical OR where the original gate performed an AND. In figure 15(d), we take our BBMCA logic block of figure 10(b) and add a vertically reflected copy. This pair of circuits, taken together, has both vertical and horizontal symmetry. Given quad-rail inputs (dual rail inputs along with their dual-rail complements), it produces corresponding quad-rail outputs, which can be reflected backward using the collision of figure 14(a), and separated at mirrors, as shown in figure 14(c). Now note that the constant-lifting technique of figure 11(b) works equally well even if all of the constant streams have 1s flowing in both directions simultaneously, by virtue of the bidirectional collision case of figure 13(b). Thus we are able to apply the constant-symmetrizing technique to mirror streams that connect the four signals between our BBMCA logic blocks (fig. 11(c)), and complete our construction.

5.10 OTHER LATTICES

All of this works equally well for an SSM on the triangular lattice, and is even slightly simpler, since we don't need to add extra constant streams at mirrors where forward and backward moving signals separate (as we did in fig. 14(c)). The complete rule is given in figure 16(a): the dotted arrow indicates a position

where an extra "spectator" particle may or may not come in. If present, it passes straight through and doesn't interfere with the collision. In figures 16(b) and 16(c), we see how mirrors and switch gates (and similarly any other square-lattice SSM combinational circuit) can simply be stretched vertically to fit onto the triangular lattice. A back reflection, where signals are sent back the way they came, is shown in figure 16(d).

This of course also means that the corresponding three-dimensional model (fig. 4(c)) can perform efficient momentum-conserving computation, at least in a single plane. If we have a dual-rail pair in one plane of this lattice, and its dual-rail complement directly below it in a parallel plane, this combination can be deflected cleanly in either of two planes by a pair of constant mirror streams. Thus it seems plausible that this kind of discussion may be generalized to three dimensions, but we won't pursue that here.

6 RELATIVISTIC CELLULAR AUTOMATAS

We have presented examples of reversible lattice gases that support universal computation and that can be interpreted as a discrete-time sampling of the classical-mechanical dynamics of compressible balls. We would like to present here an alternative interpretation of the same models as a discrete-time sampling of relativistic classical mechanics, in which kinetic energy is converted by collisions into rest mass and then back into kinetic energy.

For a relativistic collision of some set of particles, both relativistic energy and relativistic momentum are conserved, and so:

$$\sum_i E_i = \sum_i E'_i, \quad \sum_i E_i \vec{v}_i = \sum_i E'_i \vec{v}'_i,$$

where the unprimed quantities are the values for each particle before the collision, and the primed quantities are after the collision. These equations are true regardless of whether the various particles involved in the collision are massive or massless. Now we note that for *any mass- and momentum-conserving lattice gas*,

$$\sum_i m_i = \sum_i m'_i, \quad \sum_i m_i \vec{v}_i = \sum_i m'_i \vec{v}'_i,$$

and so we need only reinterpret what is normally called "mass" in these models as relativistic energy in order to interpret the collisions in such a lattice gas as being relativistic. If all collisions are relativistically conservative, then the overall dynamics exactly conserves relativistic energy and momentum, regardless of the frame of reference in which the system is analyzed. Normal nonrelativistic systems have separate conservations of mass and nonrelativistic energy—a property that the collisions in most momentum-conserving lattice gases lack. Thus we might argue that the relativistic interpretation is more natural in general.

In the collision of figure 3(b), for example, we might call the incoming pair of particles "photons," each with unit energy and unit speed. The two photons collide, and the vertical components of their momenta cancel, producing a slower moving ($v = 1/\sqrt{2}$) massive particle ($m = \sqrt{2}$) with energy 2 and with the same horizontal component of momentum as the original pair. After one step, the massive particle decays back into two photons. At each step, relativistic energy and momentum are conserved.

As is discussed elsewhere [16, 17], macroscopic relativistic invariance could be a key ingredient in constructing CA models with more of the macroscopic richness that Nature has. If a CA had macroscopic relativistic invariance, then every complex macroscopic structure (assuming there were any!) could be set in motion, since the macroscopic dynamical laws would be independent of the state of motion. Thus complex macroscopic structures could move around and interact and recombine.

Any system with macroscopic relativistic symmetry is guaranteed to also have the relativistic conservations of energy and momentum that go along with it. As Fredkin has pointed out, a natural approach to achieving macroscopic symmetries in CAs is to start by putting the associated microscopic conservations directly into the CA rule—we certainly can't put the continuous symmetries there! Momentum and mass-conserving LGA models effectively do this.

Of course, merely reinterpreting the microscopic dynamics of lattice gases relativistically doesn't make their macroscopic dynamics any richer. One additional microscopic property that we can look for is the ability to perform computation, using space as efficiently as is possible: this enables a system to support the highest possible level of complexity in a finite region. Microscopically, SSM gases have both a relativistic interpretation and spatial efficiency for computation. What we would really like is a dynamics in which both of these properties persist at the macroscopic scale.

If we are trying to achieve macroscopic relativistic invariance along with efficient macroscopic computational capability, we can see that one potential problem in our "bounce back" SSM gases (figs. 13 and 16) is a defect in their discrete rotational symmetry. Dual-rail pairs of signals aligned in one orientation can't easily interact with dual-rail pairs that are aligned in a 60° (triangular lattice) or 90° (square lattice) rotated orientation. If this causes a problem macroscopically, we can always try adding individual signal delays to the model, as in figure 12(b). This may have macroscopic problems as well, however, since turning signals with the correct timing requires several correlated interactions. Of course the reason we adopted dual-rail signaling to begin with was to avoid mixing logic-value information with signal momentum—every dual-rail signal has unit momentum and can be reflected without "measuring" the logic value. Perhaps we should simply decouple these two quantities at the level of the individual particle, and use some other degree of freedom (other than presence or absence of a particle) to encode the logic state (e.g., angular momentum). An example of a model which decouples logic values and momentum is given in figure 17.

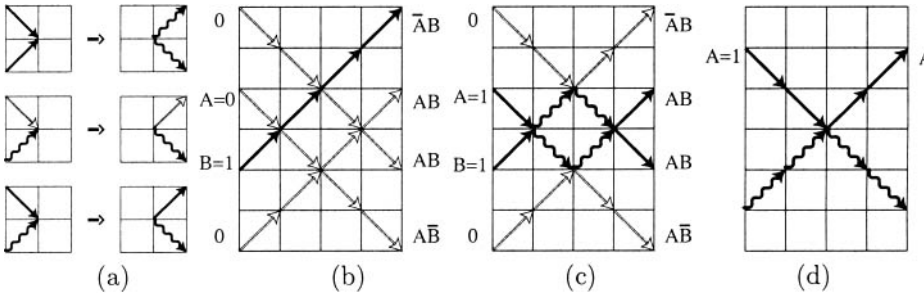

FIGURE 17 A computation-universal LGA in which ones and zeros have the same momentum. (a) We use three kinds of particles that interact. The three cases shown, plus rotations, inversions and the time reversal of these cases, are all of the interactions. In all other cases, particles don't interact. (b) We use the solid-black particles to represent "ones" and the dotted particles to represent "zeros." If only a single one comes into a "collision," none of the interaction cases applies, and so everything goes straight. (c) If two ones collide, they turn into a third kind of particle (shown as a wavy arrow), which is deflected by a zero (and deflects the zero). The inverse interaction recreates the two ones, displaced inwards from their original paths (as in an SSM collision). (d) The wavy particle also deflects (and is deflected by) a one.

In figure 17(a), we define a rule which involves three kinds of interacting particles. Figures 17(b) and 17(c) show how an SSM-style collision gate can be realized, using one kind of particle to represent an intermediate state. Single ones go straight, whereas pairs of ones are displaced inwards. Both ones and zeros are deflected by the wavy "mirror" particles, which can play the role of the mirror streams in our earlier constructions. Deflecting a binary signal conserves momentum without recourse to dual rail logic, and without contaminating the mirror stream. Adding rest particles to this model allows signals to cross (since the rule is, "in all other cases particles don't interact"). Models similar to this "proto-SSM" would be interesting to investigate on other lattices, in both two dimensions and three dimensions.

The use of rest particles to allow signals to cross in this and earlier rules raises another issue connected with the macroscopic limit. If we want to support complicated macroscopic moving structures that contain rest particles, we have to have the rest particles move along with them! (Or perhaps use moving signals to indicate crossings.) If we want to make rest particles "move," they can't be completely noninteracting. Thus we might want to extend the dynamics so that rest particles can both be created and destroyed. This could be done by redefining some of the noninteracting collision cases that have not been used in our constructions—we have actually used very few of these cases. These collisions would be different from the springy collision of figure 3(a). Even a single-particle colliding with a rest particle can move it (as in fig. 12(b) for example).

These are all issues that can be approached both theoretically, and by studying large-scale simulations [17].

7 SEMICLASSICAL MODELS OF DYNAMICS

The term *semiclassical* has been applied to analyses in which a classical physics model can be used to reproduce properties of a physical system that are fundamentally quantum mechanical. Since the finite and extensive character of entropy (information) in a finite physical system is such a property [1], all CA models can in a sense be considered semiclassical. It is interesting to ask what other aspects of quantum dynamics can be captured in classical CA models.

One such aspect is the relationship in quantum systems between energy and maximum rate of state change. A quantum system takes a finite amount of time to evolve from a given state to a different state (i.e., a state that is quantum mechanically orthogonal). There is a simple relationship between the energy of a quantum system in the classical limit and the maximum rate at which the system can pass through a succession of distinct (mutually orthogonal) quantum states. This rate depends only on how much energy the system has. Suppose that the quantum mechanical average energy E (which is the energy that appears in the classical equations of motion) is measured relative to the system's ground-state energy, and in units where Planck's constant h is one. Then the maximum number of distinct changes that can occur in the system per unit of time is simply $2E$, and this bound is always achieved by some state [18].

Now suppose we have an energy-conserving LGA started in a state with total energy E, where E is much less than the maximum possible energy that we can fit onto the lattice. Suppose also that the smallest quantity of energy that moves around in the LGA dynamics is a particle with energy "one." Then with the given energy E, the maximum number of spots that can possibly change on the lattice in one time step is $2E$ (just as in the quantum case): E smallest energy particles can each leave one spot and move to another, each causing two changes if none of them lands on a spot that was just vacated by another particle. Since the minimum value of ΔE is 1 in this dynamics, and the minimum value of Δt is 1 since this is our integer unit of time, it is consistent to think of this as a system in which the minimum value of $\Delta E \Delta t$ is 1 (which for a quantum system would mean $h = 1$). Thus simple LGAs such as the SSM gases reproduce the quantum limit in terms of their maximum rate of dynamical change.

This kind of property is interesting in a physical model of computation, since simple models that accurately reflect real physical limits allow us to ask rather sharp questions about quantifying the physical resources required by various algorithms (cf. Frank [8]).

8 CONCLUSION

We have described soft sphere models of computation, a class of reversible and computation-universal lattice gases which correspond to a discrete-time sampling of continuous classical mechanical systems. We have described models in both two dimensions and three dimensions that use immovable mirrors, and provided a technique for making related models without immovable mirrors that are exactly momentum conserving while preserving their universality and spatial efficiency. In the context of the two-dimensional momentum-conserving models, we have shown that it is possible to avoid entropy generation associated with routing signals. For all of the momentum conserving models we have provided both a nonrelativistic and a relativistic interpretation of the microscopic dynamics. The same relativistic interpretation applies generally to mass and momentum conserving lattice gases. We have also provided a semiclassical interpretation under which these models give the correct physical bound on maximum computation rate.

It is easy to show that reversible LGAs can all be turned into quantum dynamics which reproduce the LGA state at integer times [14]. Thus SSM gases can be interpreted not only as both relativistic and nonrelativistic systems, but also as both classical and as quantum systems. In all cases, the models are digital at integer times, and so provide a link between continuous physics and the dynamics of digital information in all of these domains, and perhaps also a bridge linking informational concepts between these domains.

ACKNOWLEDGMENTS

This work was supported by DARPA under contract number DABT63-95-C-0130. This work was stimulated by the SFI Constructive CA workshop.

REFERENCES

[1] Baierlein, R. *Atoms and Information Theory: An Introduction to Statistical Mechanics.* San Francisco: W. H. Freeman, 1971.

[2] Banks, E. "Information Processing and Transmission in Cellular Automata." Tech. Rep. MAC TR-81, Massachusetts Institute of Technology Project MAC, Cambridge, MA, 1971.

[3] Bennett, C. H. "The Thermodynamics of Computation—a Review." In *Proceedings of the Physics of Computation Conference*, edited by E. Fredkin, R. Landauer, and T. Toffoli, 905–940.

[4] D'Souza, R. M., and N. H. Margolus. "Thermodynamically Reversible Generalization of Diffusion Limited Aggregation." *Phys. Rev. E* **60(1)** (1999): 264–274.

[5] D'Souza, R. M. "Macroscopic Order from Reversible and Stochastic Lattice Growth Models." Ph.D. thesis (Physics), Massachusetts Institute of Technology, Cambridge, MA, August 1999.

[6] Farmer, D., T. Toffoli, and S. Wolfram, eds. In *Cellular Automata.* Amsterdam: North-Holland, 1984.

[7] Feynman, R. P. *Feynman Lectures on Computation,* edited by J. G. Hey and R. W. Allen. Reading, MA: Addison-Wesley, 1996.

[8] Frank, M. P. "Reversibility for Efficient Computing." Ph.D. thesis, Massachusetts Institute of Technology AI Laboratory, Cambridge, MA, 1999.

[9] Fredkin, E., R. Landauer, and T. Toffoli, eds. *Proceedings of the Physics of Computation Conference.*

[10] Fredkin, E., and T. Toffoli. "Conservative Logic." In *Proceedings of the Physics of Computation Conference*, 219–253.

[11] Frisch, U., B. Hasslacher, and Y. Pomeau. "Lattice-Gas Automata for the Navier-Stokes Equation." *Phys. Rev. Lett.* **56** (1986): 1505–1508.

[12] Hardy, J., O. de Pazzis, and Y. Pomeau, "Molecular Dynamics of a Classical Lattice Gas: Transport Properties and Time Correlation Functions." *Phys. Rev. A* **13** (1976): 1949–1960.

[13] Margolus, N. "Physics-like Models of Computation." In *Cellular Automata*, edited by D. Farmer, T. Toffoli, and S. Wolfram, 81–95. Amsterdam: North-Holland, 1984.

[14] Margolus, N. "Quantum Computation." In *New Techniques and Ideas in Quantum Measurement Theory*, edited by D. Greenberger, 487–497. New York: New York Academy of Sciences, 1986.

[15] Margolus, N. "Physics and Computation." Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, 1987. Reprinted as *Tech. Rep. MIT/LCS/TR-415*, MIT Lab. for Computer Science, Cambridge MA, 1988.

[16] Margolus, N. "A Bridge of Bits." In *Proceedings of the Workshop on Physics and Computation—PhysComp '92*, 253–257. Los Alamitos, CA: IEEE Computer Society Press, 1993.

[17] Margolus, N. "Crystalline Computation." In *Feynman and Computation*, edited by Hey, 267–305. Reading MA: Perseus Books, 1998.

[18] Margolus, N., and L. Levitin. "The Maximum Speed of Dynamical Evolution." *Physica D* **120(1/2)** (1998): 188–195.

[19] Matzke, D., ed. *Proceedings of the Workshop on Physics and Computation—PhysComp '92*, Los Alamitos, CA: IEEE Computer Society Press, 1993.

[20] Moore, C., and M. G. Nordahl. "Lattice Gas Prediction is P-Complete." Working Paper 97-04-034, Santa Fe Institute, Santa Fe, NM, 1997.

[21] Margolus, N. "An Embedded DRAM Architecture for Large-Scale Spatial-Lattice Computations." In *The 27th Annual International Symposium on Computer Architecture. IEEE Computer Society* (2000): 149–160.

[22] Margolus, N. "Universal CA's Based on the Collisions of Soft Spheres." In *Collision Based Computation*, edited by Andrew Adamatzky. London: Springer Verlag, to appear.

[23] Wolfram, S. *Theory and Applications of Cellular Automata.* World Scientific, 1986.

Emerging Markets and Persistent Inequality in a Nonlinear Voter Model

Nienke A. Oomes

1 INTRODUCTION

Since it is one of the few spin systems that can be studied analytically, the Voter Model has been extensively discussed in the interacting particle systems literature.[1] In the original interpretation of this model, voters choose their political positions with probabilities equal to the voting frequency of their "friends." One of the main results is that, in one and two dimensions, the system clusters—i.e., converges to a homogeneous steady state—while heterogeneity can persist only in dimensions higher than two.

This chapter develops an economic model that is similar to the Voter Model, in that agents decide between "economic positions," conditional on the economic choices of their trade partners. The choices considered here are market production and nonmarket production, where the payoffs associated with market production for a given agent are a function of the amount of market goods produced by others. Intuitively, the more people are producing for the market, the more

[1] For an overview, see Liggett [14, ch. V] and Durrett [9, ch. III].

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

potential trade partners exist, hence the higher the expected payoff associated with market production. Similarly, the smaller the extent of the market, the lower the expected gains from trade, hence the smaller the incentive to produce for the market.[2]

When each agent is assumed to have an equal probability of trading with any other agent in his or her trade network, the payoffs associated with market production are linearly increasing in the network's total market output. However, this linearity in *payoffs* does not necessarily imply that the *conditional probability* of working for the market is linearly increasing in total market production, as the Voter Model would have it. As it turns out, this follows only if agents believe, mistakenly, that their trade partners will decide to work for the market with a probability that is exactly proportional to their current market output.

Clearly, a more general approach is obtained by allowing for different types of expectations agents may have about the production decisions of their trade partners. A model that allows for such an approach is the so-called *Nonlinear Voter Model* (NLVM), studied by Molofsky et al. [15]. An interesting aspect of this model is that, unlike the Voter Model, it is able to generate persistent heterogeneity even in two dimensions.

In an economic context, this result is relevant both for studying emerging markets, and for analyzing the persistence of inequality. Will markets eventually take over all types of nonmarket production, or can both forms of production coexist? Is it possible for a "culture of poverty" to persist in the midst of plenty? While the model presented here is obviously a considerable simplification of any real economy, it might give us some insight into these questions.

The remainder of this chapter is organized as follows. First, the underlying assumptions of the model, along with its game-theoretical representation, are presented in sections 2 and 3. Then, in section 4, the NLVM is discussed, with a focus on the conditions under which it generates persistent inequality. The major finding here is that the parameter values that allow for this to happen are not realistic from an economic point of view. Finally, in section 5, it is demonstrated that, in a more general NLVM that includes the Ising model as a special case, persistent inequality may arise for more realistic economic parameter values. Section 6 concludes, and suggests some directions for future research.

2 DESCRIPTION OF THE MODEL

Consider an artificial economy that is inhabited by N agents. The agents live on a two-dimensional lattice $S \subseteq \mathbf{Z}^2$, and are indexed by their coordinates $x = (i, j)$. Each period, they independently decide whether or not to engage in market production. Decisions depend on endowments, technology, trade structure, and preferences, which are described below.

[2]This is essentially the argument put forward by Adam Smith in the first three chapters of his *Wealth of Nations* (1776).

2.1 ENDOWMENTS

Each agent is endowed with one homogeneous unit of labor time per period, of which $\eta_t(x)$ is allocated to market production, and $1 - \eta_t(x)$ is allocated to nonmarket activities. The latter might be thought of as farming, household work, voluntary work, or even crime. What is important, however, as will be described below, is that the payoffs associated with nonmarket production are assumed to be independent of the production decisions of other agents, and that these payoffs should be high enough so as to allow agents to be self-sufficient, i.e., to survive outside the market.

The variable $\eta_t(x)$ is assumed to have binary support $\Omega = \{0, 1\}$, implying that agents specialize in either market or nonmarket production.[3] This assumption is less restrictive than it may appear at first, since t may be thought of as a relatively short period of time, such as a day. In that sense, part-time work is possible, for instance, by allowing agents to switch every other day between market and nonmarket work. At any given moment in time, however, the state of the economy η_t is a binary configuration with state space $\{0, 1\}^N$.

2.2 TECHNOLOGY

The production processes for both market and nonmarket activities are constant returns to scale, with market production assumed to be twice as efficient. Normalizing the marginal productivity of market work to one, this gives a pair of production functions for each site x:

$$y_t(x) = \eta_t(x)\,, \tag{1}$$

$$h_t(x) = \frac{1}{2}[1 - \eta_t(x)]\,, \tag{2}$$

where $y_t(x)$ is the amount of market goods produced at site x, and $h_t(x)$ is the amount of nonmarket goods produced and consumed at site x during period t.

2.3 TRADE STRUCTURE

The trade structure between agents is represented as a doubly stochastic matrix $\mathbf{P}$, the elements $p(x, y)$ of which denote the probability that x trades with y. The trade structure is exogenous, and is assumed to satisfy the restriction that each agent x trades, on average, an equal amount of her market goods with any other agent within her "trade network" $N_x \in S$. This implies that, within a given trade network, all agents have an equal probability of trading with each other:

$$p(x, y) = \begin{cases} 0 & \text{if } y \notin N_x\,, \\ |N_x|^{-1} & \text{if } y \in N_x\,. \end{cases} \tag{3}$$

[3] A more general setup is discussed in Oomes [16].

TABLE 1 Payoff Matrix for the Two-Player Game.

	$\eta_t(y) = 0$	$\eta_t(y) = 1$
$\eta_t(x) = 0$	0.5, 0.5	0.5, 0
$\eta_t(x) = 1$	0, 0.5	1, 1

2.4 PREFERENCES

From the perspective of the consumer, market and nonmarket goods are considered perfect substitutes, meaning that they yield an equal amount of utility per unit consumed. These preferences can be represented by a linear utility function:

$$U_t(x) = h_t(x) + c_t(x)\,, \tag{4}$$

where $h_t(x)$ is the amount of nonmarket goods, and $c_t(x)$ the amount of market goods consumed at site x during period t.

Since nonmarket goods, by definition, have no market value, market goods can only be obtained when they are exchanged for other market goods. Therefore, the amount of market goods an agent is able to consume depends, first of all, on whether this agent is producing market goods herself, and secondly, on the amount of market production by her trade partners:

$$c_t(x) = \eta_t(x) \sum_{y \in S} p(y, x)\eta_t(y)\,. \tag{5}$$

Combined with the production function for nonmarket goods, this gives a mapping from the production configuration η_t to the welfare distribution U_t, where, for each agent x, utility is defined as

$$U_t(x) = [1 - \eta_t(x)]0.5 + \eta_t(x) \sum_{y \in S} p(y, x)\eta_t(y)\,. \tag{6}$$

3 GAME-THEORETICAL REPRESENTATION

The assumption that each agent trades an equal amount of her market goods with any other agent in her trade network can be alternatively interpreted so as to mean that, in each period, each agent x is matched with a random trade partner y. These trade partners, then, can be considered to play a "game," the payoff matrix of which is represented in table 1. This game has the structure of a "coordination game," which Cooper [5] describes as having the following characteristics:

1. The actions of players are "strategic complements," implying that an increase in the level of activity of one agent creates an incentive for increased activity by the other agent.[4]
2. The existence of multiple, Pareto-ranked Nash equilibria, which implies that self-fulfilling pessimistic beliefs may give rise to a Pareto-suboptimal equilibrium outcome, the realization of which is termed a *coordination failure.*[5]
3. The equilibria are regular in that small variations in the payoffs of the game do not result in large changes in the number of equilibria.

It is important to realize that the two-player game is, in fact, an evolutionary game, in which N agents have to choose their optimal strategies simultaneously and independently, without knowing in advance who their "opponent" is going to be. A strategy $p_y = \Pr(\eta_t(y) = 1) = \sum_{y \in S} p(y,x)\eta_t(y)$ may, therefore, be interpreted as a *belief* on the part of agent x concerning the probability of being matched with *some* $y \in N_x$ who is engaged in market production and, therefore, is able to trade.

Given that the random opponent y plays strategy p_y, the expected payoffs for agent x are:

$$\begin{aligned} U_{t,0}(x) &= 0.5\,, \\ U_{t,1}(x) &= p_y\,, \end{aligned} \tag{7}$$

where $U_{t,0}(x)$ denotes the payoff associated with strategy $\eta_t(x) = 0$ and $U_{t,1}(x)$ denotes the expected payoff associated with strategy $\eta_t(x) = 1$. Letting $p_x = \Pr(\eta_t(x) = 1)$ denote the strategy of agent x, this gives as the expected utility for x:

$$U_t(x) = (1 - p_x)0.5 + p_x p_y\,. \tag{8}$$

3.1 NASH EQUILIBRIUM

Since the model has the structure of a game, it seems appropriate to use the notion of a Nash equilibrium as the solution concept for this model. Before defining a Nash equilibrium; however, we first need to define the notion of a "best response."

Definition 1. *A* ***best response*** *for x is the strategy p_x that maximizes x's utility conditional on y playing strategy p_y. A* ***best response correspondence*** *for x is a mapping from p_y to p_x such that $p_x(p_y)$ gives x's best response to any strategy p_y.*

[4] On the notion of "strategic complements," see Cooper and John [6]. In Molofsky et al. [15], this characteristic is referred to as "positive frequency dependence."

[5] An allocation is Pareto optimal if no other feasible allocation exists that would make at least one agent better off, without making any other agent worse off. The notion of a Nash equilibrium is defined below.

Given the utility function above, the best response for x in this model is:

$$p_x(p_y) = \text{argmax } \{(1-p_x)0.5 + p_x p_y\} . \tag{9}$$

Since the game is symmetric, the best response correspondence for y is of the same form.

Definition 2. *A **Nash Equilibrium** is a strategy profile $\{p_x^*, p_y^*\}$, such that p_x^* and p_y^* are mutual best responses.*

When $p_x^*, p_y^* \in \{0,1\}$ (i.e., agents choose one action or another with probability one), this is called a Nash equilibrium in pure strategies, or "pure Nash equilibrium." Similarly, when $p_x^*, p_y^* \in \langle 0,1\rangle$ (i.e., agents randomize between actions; or believe each other to randomize), we have a Nash equilibrium in mixed strategies, also called a "mixed Nash equilibrium."

For the static two-player game above, it can easily be checked that the strategy profiles $\{0,0\}$ and $\{1,1\}$ each constitute a Nash equilibrium in pure strategies: if an opponent is expected to produce for the market, the best response is to produce for the market as well; and vice versa for nonmarket production. The equilibrium $\{0,0\}$ is Pareto suboptimal, since by moving to $\{1,1\}$ both agents would be better off. There also exists a (weak) Nash equilibrium in mixed strategies, where both players randomize between market and nonmarket production with probability one half.[6]

3.2 BEST RESPONSE DYNAMICS

The concept of a best response appears to presuppose that agents know the probability with which their random trade partners make choices. However, while it is true that expectations should be correct in a Nash equilibrium, outside of equilibrium they may be right or wrong, rational or irrational, optimistic or pessimistic. Without further assumptions on expectations formation, the model does not say anything about the way in which agents come to *learn* others' best responses. What is missing is a theory of equilibrium selection, and an explanation of how it is possible that a large number of agents could possibly coordinate on any given equilibrium.

As evolutionary game theorists have pointed out,[7] the strict rationality assumptions that might be reasonable consistency requirements in one-shot 2-by-2 games, or economies with a small number of "representative agents," become quite untenable when applied to games with a large number of players. If players repeatedly encounter similar situations, it might be more reasonable to assume

[6] To see that this constitutes a Nash equilibrium, observe that, when one agent randomizes (or is believed to randomize) with probability 0.5, the expected payoff for the other agent is the same whatever her response; hence, randomizing with probability 0.5 is one of many best responses.

[7] For example, Samuelson [17].

that agents are *myopic*; i.e., they base their expectations of others' behavior on the immediate past.

This implies that the Nash equilibrium can be interpreted as the steady state (fixed point) of a Markov random field, which must satisfy, $\forall\, x, y \in S$,

$$\Pr(\eta_{t+1}(x) = 1) = \Phi(\hat{\eta}_t(y))\,, \tag{10}$$

where

$$\begin{aligned}\hat{\eta}_t(y) &= \sum_{y \in S} p(y,x)\eta_t(y) \\ &= |N_x|^{-1} \sum_{y \in N_x} \eta_t(y)\end{aligned} \tag{11}$$

is simply the average amount of market production in the trade network. Since agents are homogeneous with respect to their endowments, technologies, and preferences, it seems natural to assume that they also form their expectations in the same way, hence $\Phi(\cdot)$ should be the same for all agents.[8]

An interesting question that arises in this case is whether, in spite of the assumed homogeneity of agents, it is possible for asymmetric outcomes to persist. That is, even though the model predicts that any Nash equilibrium of the two-player game must be symmetric (i.e., $p_x^* = p_y^*$), is it possible in an N-player game for different subsets of the population to consistently coordinate on different equilibria? In other words, is it possible for market and nonmarket modes of production to coexist in an economy, or will one of the two eventually take over?

This question is relevant, not just in light of the problems faced by emerging market economies and economies in transition, but also given the fact that an unequal distribution of market and nonmarket work implies persistent inequality between agents. That is, since each equilibrium is associated with different levels of "utility," corresponding to different levels of consumption, agents who manage to coordinate on the market equilibrium are clearly better off than agents who coordinate on the nonmarket equilibrium.

In order to explore the implications of different assumptions on $\Phi(\cdot)$ for the existence, uniqueness, and stability of steady states, the next section will present a nonparametric approach, based on the Nonlinear Voter Model (NLVM) studied by Molofsky et al. [15]. While this model, unlike the Voter Model, is able to generate persistent inequality in two dimensions, the conditions under which this is possible are argued not to be economically plausible. In section 5, however, a slightly modified NLVM will be presented for which inequality can persist under more reasonable conditions.

[8] An interesting direction for future work would be to endogenize the choice of $\Phi(\cdot)$, for instance, along the lines of Crutchfield [7]. In this chapter, however, expectations will be taken as exogenous.

TABLE 2 LVM and MVM as Special Cases of the NLVM.

k	p_k	NLVM	LVM	MVM
0	p_0	0	0.0	0
1	p_1	p_1	0.2	0
2	p_2	p_2	0.4	0
3	p_3	$1-p_2$	0.6	1
4	p_4	$1-p_1$	0.8	1
5	p_5	1	1.0	1

4 THE NONLINEAR VOTER MODEL

The Nonlinear Voter Model (NLVM) is a nonparametric approach and, as such, very well suited for studying the implications of different expectational assumptions, without requiring us to specify a functional form for $\Phi(\cdot)$.

An economic interpretation of the NLVM is as follows. At the beginning of each period, each agent judges the state of the economy by counting the number of market workers in a random sample of size K. It then uses the decision rule: hire with probability p_k if the sample contains k market workers, where

$$p_k = \Pr\left(\eta_{t+1}(x) = 1 \mid \sum_{y \in K} \eta_t(y) = k\right). \tag{12}$$

In order to reduce the parameter space, Molofsky et al. [15] propose to impose two restrictions. First of all, they impose a symmetry restriction:

$$p_k = 1 - p_{K-k}. \tag{13}$$

Secondly, it is assumed that $p_0 = 0$; i.e., if no one in the sample is currently involved in market production, the percentage of market workers in one's trade network is predicted to be below 50%, and so the best response is to stay out of the market. By the symmetry restriction, this also implies $p_1 = 1$.

Together, the two restrictions imply that the model has $(K-1)/2$ free parameters. In order to be able to represent this space in a two-dimensional graph, it is assumed that $K = 5$. The resulting probabilities are given in table 2. As this table shows, two special cases of the NLVM are the Linear Voter Model (LVM) and the Majority Voter Model (MVM), both of which also have parametric representations. The LVM, which was introduced independently by Clifford and Sudbury [4] and by Holley and Liggett [13], assumes that

$$\Pr(\eta_{t+1}(x) = 1) = \Phi(\hat{\eta}_t(y)) = \hat{\eta}_t(y). \tag{14}$$

A well-known property of this model is that, in one and two dimensions, it converges to steady states with all zeros or all ones (i.e., the pure Nash equilibria)

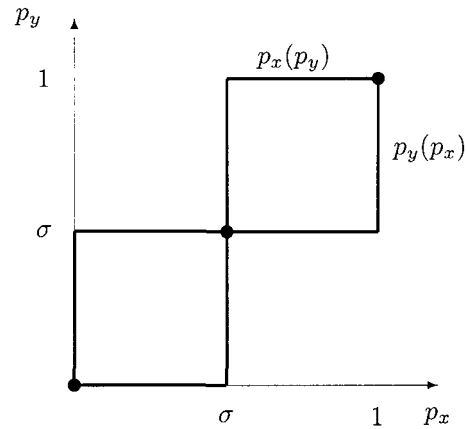

FIGURE 1 Best Response Correspondences for the Majority Voter Model.

while, in infinite spaces with three or more dimensions, additional steady states (i.e., mixed Nash equilibria) exist.[9] This result applies to any trade network structure, finite or infinite, that constitutes a connected graph.[10]

The MVM, on the other hand, assumes that agents will produce for the market if the majority of their trade partners was involved in market production in the previous period, and will remain autarkic if the majority of their trade partners was autarkic in the previous period. If there is a match, agents randomize with probability one half. This gives as the best response correspondence:

$$\Pr(\eta_{t+1}(x) = 1) = \begin{cases} = 0 & \text{if } \hat{\eta}_t(y) < 0.5\,, \\ \in [0,1] & \text{if } \hat{\eta}_t(y) = 0.5\,, \\ = 1 & \text{if } \hat{\eta}_t(y) > 0.5\,. \end{cases} \tag{15}$$

As figure 1 shows, the best response correspondences of any two agents intersect in three places, implying that there are three symmetric Nash Equilibria: two pure and one mixed. However, since agents respond myopically to each other, only the pure Nash equilibria can be stable steady states of the system (this is explained in more detail below). Hence, it is impossible for inequality to persist in both the LVM and the MVM.

[9]For proofs, see Liggett [14, ch. V] and Durrett [9, ch. III].

[10]More specifically, the proofs generalize to any model in which x adopts the position of y with some probability $p(y,x)$, in which case mixed strategy equilibria exist when the random walk with kernel p is recurrent, and do not exist when it is transient.

4.1 NLVM WITH GLOBAL TRADE

When trade networks are "global," i.e., $N_x = S$, each agent has an equal probability to trade with any other agent. Assuming that the lattice is infinite, then the law of large numbers implies that the unconditional probability of being a market worker, $\Pr(\eta_t(x) = 1)$, identically equals the percentage of market workers in the entire economy, $\hat{\eta}_t$. Given that the number of possible ways in which k market workers can be distributed over a sample of size k equals $\binom{K}{k}$, this implies that the expected amount of market production in the next period is

$$\hat{\eta}_{t+1} = \sum_{k=0}^{K} \binom{K}{k} \hat{\eta}_t^k (1 - \hat{\eta}_t)^{K-k}. \tag{16}$$

For $K = 5$, this yields

$$\begin{aligned} \hat{\eta}_{t+1} = {} & 5p_1\hat{\eta}_t(1 - \hat{\eta}_t)^4 + 10p_2\hat{\eta}_t^2(1 - \hat{\eta}_t)^3 + 10(1 - p_2)\hat{\eta}_t^3(1 - \hat{\eta}_t)^2 \\ & + 5(1 - p_1)\hat{\eta}_t^4(1 - \hat{\eta}_t) + \hat{\eta}_t^5 . \end{aligned} \tag{17}$$

For the linear voter model, $(p_1, p_2) = (0.2, 0.4)$, it can be checked that this function reduces to $\hat{\eta}_{t+1} = \hat{\eta}_t$, and, hence, coincides with the 45-degree line. This implies that, in a perfectly globalized, infinite economy, any production level $\hat{\eta}$ constitutes a steady state.

The linear case, however, is a rather special case. It can be verified analytically that all other, nonlinear models have either three or five steady states, which may or may not be stable. The three steady states common to all models are the pure Nash equilibria $\hat{\eta} = 0$ and $\hat{\eta} = 1$, and the mixed Nash equilibrium $\hat{\eta} = 0.5$. To check for stability of these steady states, define

$$\begin{aligned} J(\hat{\eta}_{t+1}) = {} & \frac{\delta\hat{\eta}_{t+1}}{\delta\hat{\eta}_t} \\ = {} & 5p_1(1 - \hat{\eta}_t)^4 + 20(p_2 - p_1)\hat{\eta}_t(1 - \hat{\eta}_t)^3 \\ & + 30(1 - 2p_2)\hat{\eta}_t^2(1 - \hat{\eta}_t)^2 \\ & + 20(p_2 - p_1)\hat{\eta}_t^3(1 - \hat{\eta}_t) + 5p_1\hat{\eta}_t^4 . \end{aligned} \tag{18}$$

Evaluated at the steady states, this gives the eigenvalues

$$\begin{aligned} J(0) &= J(1) = 5p_1 , \\ J(0.5) &= -\left(\frac{1}{8}\right)(15p_1 + 10p_2 - 15) . \end{aligned} \tag{19}$$

Stability requires $|J(\hat{\eta})| < 1$. This implies that $\hat{\eta} = 0$ and $\hat{\eta} = 1$ are stable for $p_1 < 0.2$, while $\hat{\eta} = 0.5$ is stable for $\{p_1, p_2 : 7 < 15p_1 + 10p_2 < 23\}$.

As $|J(\hat{\eta})| > 1$, the steady states lose stability and additional steady states may emerge. As Molofsky et al. [15] show, this implies that there are five possible "regimes," which are plotted in the phase diagram of figure 2, and which are labeled as follows:

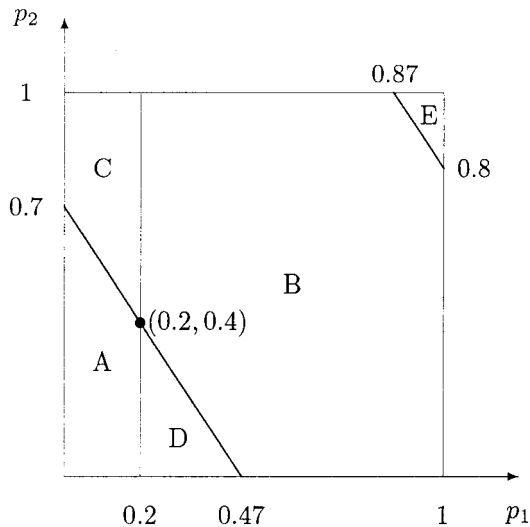

FIGURE 2 Phase Diagram for NLVM with Global Trade. (A) clustering, (B) ergodicity, (C) multiple outcomes, (D) phase separation, and (E) periodicity.

(A) clustering: only $\{0\}$ and $\{1\}$ are stable;
(B) ergodicity: only $\{0.5\}$ is stable;
(C) multiple outcomes: $\{0\}, \{0.5\}, \{1\}$ are all stable;
(D) phase separation: $\{0\}, \{0.5\}, \{1\}$ are all unstable, but there exist two additional steady states;
(E) periodicity: $\{0\}, \{0.5\}, \{1\}$ are all unstable, but there exist stable periodic solutions.

Figures 3 and 4 illustrate the dynamics of the different regimes, which may be interpreted as the best response of the average agent to the average amount of market goods produced during the previous period.

As figure 3 shows, the Majority Voter Model, for which $(p_1, p_2) = (0, 0)$, provides an example of the "clustering" regime, in that it converges to one of the pure Nash equilibria. The case $(p_1, p_2) = (0.5, 0.5)$, on the other hand, which might be called the Random Voter Model, provides an example of the "ergodic" regime in which only the mixed strategy Nash equilibrium is stable. Finally, $(p_1, p_2) = (1, 1)$ illustrates the possibility of a periodic steady state.

Figure 4 shows the existence of nontrivial steady states. In a case such as $(p_1, p_2) = (0, 1)$, it is possible for both pure and mixed Nash equilibria to be

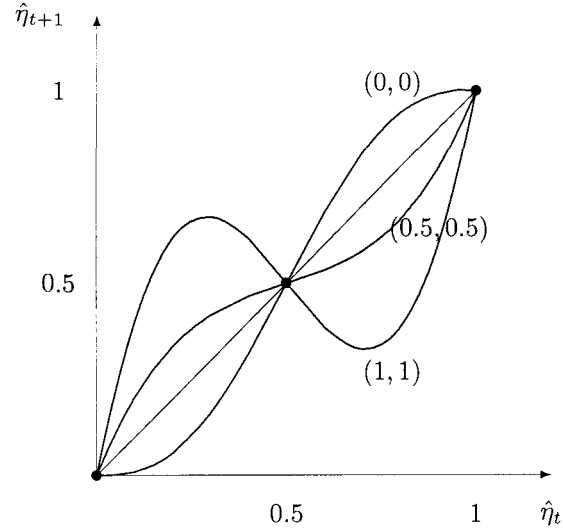

FIGURE 3 NLVM examples with three steady states.

stable (the "multiple outcome regime"), while the case $(0.3, 0.1)$ provides an example of "phase separation."

4.2 NLVM WITH LOCAL TRADE

The analytical results above are based on the assumption that trade networks are global, i.e., that geography does not matter. Even in today's "global village," however, geographic proximity is still an important determinant of trade flows.[11] The usual explanation for this fact is in terms of transportation costs, although there may be other explanations such as cultural barriers, political constraints, or, as Eichengreen and Irwin [11] suggest, path dependency in the selection of one's trade partners. Whatever the cause, however, it is interesting to explore the consequences of "localized trade" for the spatiotemporal distributions of market and nonmarket work.

To focus on an extreme case, we will replace the random sampling assumption by the assumption that each agent "samples" her nearest neighbors only. Technically, a "nearest neighborhood," or "local trade network," may be defined

[11]This is true even when one controls for spatial correlations in income. As Deardorff [8] notes, for instance: "It has long been recognized that bilateral trade patterns are well described empirically by the so-called gravity equation, which relates trade between two countries positively to both of their incomes and negatively to the distance between them, usually with a functional form that is reminiscent of the law of gravity in physics" (p. 7). For empirical evidence, see, e.g., Bergstrand [2] and the multiple articles in Frankel [12].

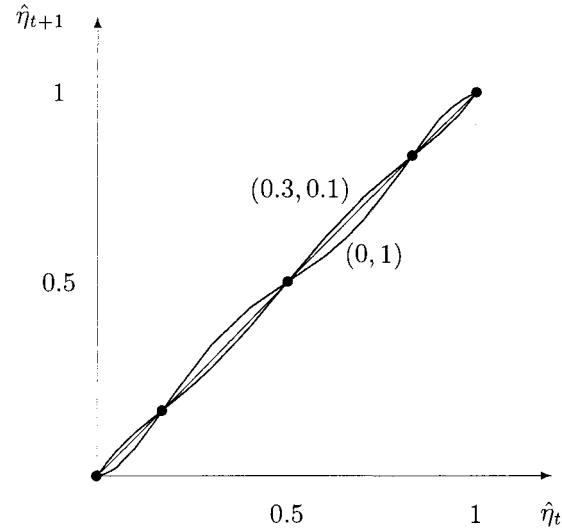

FIGURE 4 NLVM examples with five steady states.

as $N_x = \{y : |y - x| \leq \epsilon\}$, where $|y - x|$ denotes the distance of the shortest path between y and x. Following Molofsky et al. [15], we will consider the case where $\epsilon = 1$. The resulting trade network is sometimes referred to as a "Von Neumann neighborhood" or "common border neighborhood."[12] On a lattice this implies that every agent has five potential trade partners, including herself.[13]

While the local NLVM cannot, in general, be solved analytically, it was implemented by David Griffeath on his WinCA software, as well as on Norm Margolis' Cellular Automata Machine (CAM8) at the Santa Fe Institute. Griffeath's experiments, reported in the Molofsky paper, took place on a 640-by-480 lattice, starting from an initial configuration with all ones (market workers) on the West side, and all zeros (nonmarket workers) on the East side. This allowed for a relatively quick assessment of whether the pure Nash equilibria were stable or, in case they were unstable, how fast they would break down.

In order to focus on the interactions around the East/West border, a mixed boundary condition was imposed, which wraps the top edge to the bottom but

[12]Although this is obviously an extreme assumption, it is interesting to note that common borders are often found to be highly significant in explaining trade patterns (e.g., Balassa [1]).

[13]One may wonder whether it is necessary or realistic to assume that agents are consumers of their own goods. This is partly an empirical issue, which could be adjudicated, for instance, by measuring the fraction of a country's GDP, or of an industry's output, that is consumed by this country or industry itself. It is to be expected that this fraction is not only positive, but might in fact be larger than the fraction consumed by each given trade partner, especially at the country level. The implications of this are discussed in section 5.

disables interaction between the left and right edges. Following the suggestion by Durrett and Levin [10], the global trade case (which can be regarded as a "mean field approximation") was used as "simulation guide." This suggested that a phase separation regime could be found by setting $p_2 = 0$, and varying p_1 between 0 and 1. The results are reported in Molofsky et al. [15] and illustrated in figure 5.

The pure Nash equilibria appear to be stable for values of p_1 as high as 0.27, instead of the threshold 0.2 suggested by the global trade case. For these parameter values, the NLVM essentially behaves like a local Majority Voter Model with $(p_1, p_2) = (0, 0)$, which is known to evolve by what is called *curvature-driven surface tension.* This means that, when two clusters are separated by some curved boundary, the one with the highest frequency continues to grow at a rate which decreases with the curvature of the boundary. In other words, pockets of nonmarket production which are completely surrounded by market production gradually disappear, and vice versa. Started from a vertical boundary, this implies that, for a long time, neither type of economy is able to grow. Panel (a) portrays the situation after 5,000 updates. Eventually, however, one type must become predominant by chance, and will take over the entire lattice. Therefore, inequality cannot persist.

When p_1 is large enough, on the other hand, each of the two halves "invades" the other at a linear rate, and the agents quickly converge to the mixed Nash equilibrium, where everyone essentially flips a coin every period. This is shown in panel (b), for the case $p_1 = 0.35$ after 1,000 updates. Since the lattice is finite, the pure Nash equilibria $\{0\}$ and $\{1\}$ are still the only absorbing states, and therefore must be reached eventually. However, since $\{0\}$ or $\{1\}$ are unstable, even the slightest deviation from these steady states will push the system towards the metastable mixed Nash equilibrium $\{0.5\}$, where it will remain for eons of time. In this sense, the system can be called "ergodic." While random noise does imply "inequality," this inequality is not spatially persistent.

The analogue to the phase separation regime, with two nontrivial steady states, is found for $0.27 < p_1 < 0.35$. Panel (c) shows the case of $p_1 = 0.31$ after 2,000 updates. As in the "ergodic" case, market and nonmarket workers are quickly mixed, but this time a predominant market equilibrium emerges in the West, and a predominant nonmarket equilibrium in the East. Started from more general initial configurations, the economy converges to a steady state with about 75 to 80 percent of workers engaged in market production if market workers initially dominated, and vice versa for nonmarket workers. Started from symmetric randomness (Bernouilli product measures with density 0.5), the system self-organizes into clusters of these two nontrivial steady states, which themselves appear to evolve by curvature-driven surface tension. This is shown in panel (d) after 5,000 updates.

In contrast with what the global trade model (mean field approximation) suggested, Griffeath found no evidence of a multiple outcome regime. However,

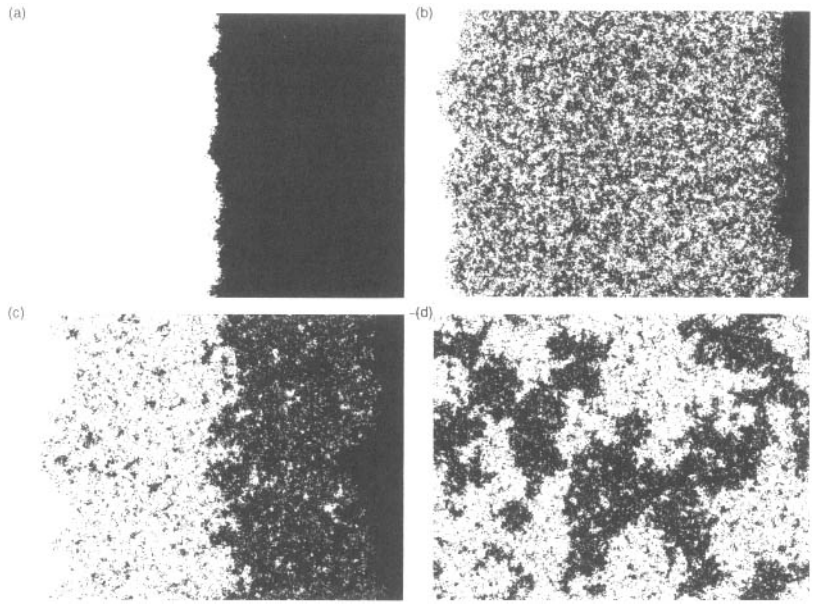

FIGURE 5 Phase Transition of the NLVM for $p_2 = 0$. (a) $p_1 = 0.27$; (b) $p_1 = 0.35$; (c) $p_1 = 0.31$; (d) $p_1 = 0.31$.

he did find a tiny area in the upper right corner of the phase space corresponding to a (quasi-)periodic regime.[14]

Based on detailed simulation of the entire $(p1, p2)$ parameter space, Griffeath estimated that the boundary separating the clustering and ergodic regions runs from about $(0.024, 1)$ down to the linear voter point $(0.2, 0.4)$, after which a bifurcation encloses the tiny phase separation region. On the basis of his simulations, however, he was unable to find the exact bifurcation point.[15]

In my view, the most mathematically as well as economically plausible candidate is the point on the phase boundary for which $p1 = p2$. This point is critical in that, as p_2 falls below p_1, the probability of working for the market is no longer monotonically increasing in the amount of market production in the neighborhood. Intuitively, when the probability of working for the market

[14]See Molofsky et al. [15] for a description of this regime.

[15]As Griffeath wrote in an early summary of his findings: "A particularly subtle issue is whether the boundary bifurcation occurs at $(0.2, 0.4)$, as in the mean-field model, or for a smaller value of p_2. In spite of the mathematical plausibility of the former alternative, our simulation data would seem to indicate the latter. As we move away from the lower edge, the discernible width of the phase separation region decreases at a rate which precludes any detectable instance of this phase with $p_2 > 0.3$, say. Assuming the boundaries vary smoothly, it is hard to see how this triangle could extend all the way to the linear voter point."

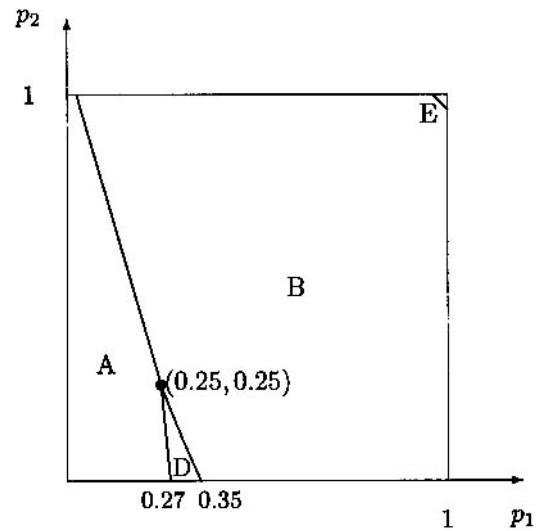

FIGURE 6 Estimated Phase Diagram for the NLVM with Local Trade.

increases as the amount of market production in the neighborhood grows from 0 to 20 percent, but then decreases as neighborhood market production grows further from 20 to 40 percent, it should not be too much of a surprise that distributions with market production around 20 percent can be metastable.

If it is true that the transition occurs at the onset of nonmonotonicity, then, assuming the estimate of $(0.024, 1)$ is correct, this would imply that the bifurcation occurs at $(14/55, 14/55)$, or approximately $(0.25, 0.25)$.[16] This point is plotted in figure 6.

5 THE ISING MODEL

Economically speaking, it seems plausible that expectations on future production levels should be monotonically increasing in current production levels. That is, the more agents are producing for the market today, the larger the expected probability that at least half of them are producing for the market tomorrow. If this is accepted, an important implication of the above analysis is that there exist no economically plausible parameters in the NLVM that allow for persistent

[16]Interestingly, the point $(0.25, 0.25)$ lies on the line $p_2 = 1 - 3p_1$, which is the prediction generated by a "dyad approximation" (see Molofsky et al. [15]). However, the dyad approximation itself is falsified by Griffeath's observation that the process is nonergodic for $p_2 = 1$ and p_1 slightly above 0, as well as for $p_2 = 0$ and p_1 slightly above $1/3$.

inequality. In the long run, the economy must be either completely "marketized," or must be fully "autarkic."

Note, however, that the NLVM was initially restricted by assuming that $p_0 = 0$ and $p_1 = 1$. Clearly, a more general NLVM would be obtained by dropping this assumption. On the other hand, in order to increase economic plausibility, it might be reasonable to restrict the NLVM to cases where $\Phi(\hat{\eta}_t(y))$ is monotonically increasing in $\hat{\eta}_t(y)$.

As an example of such a case, consider the following type of expectations:

$$\Pr\ (\eta_{t+1}(x) = 1) = \Pr\ (\hat{\eta}_t(y) > 0.5 + \mu\epsilon_t(x))\,. \tag{20}$$

This specification says that agents expect average market production to remain the same, while allowing for unpredictable deviations to occur in the form of some mean zero noise, measured by the term $\epsilon_t(x)$. The parameter μ is introduced to allow for variations in the impact of this noise term on agent's decisions.

To further study this model, we need to specify a specific distribution for $\epsilon_t(x)$. In order to preserve symmetry, this distribution would have to be symmetric around zero. In discrete choice econometrics, two common distributions used in such cases are the normal distribution ("probit") and the logistic distribution ("logit"). For our purposes, the latter assumption is the more interesting one, since it yields a best response function that is equivalent to another well-known spin system, called the Ising model:

$$\begin{aligned}\Pr\ (\eta_{t+1}(x) = 1) &= \Pr\ \left(\epsilon_t(x) < \frac{1}{\mu}(\hat{\eta}_t(y) - 0.5)\right) \\ &= \frac{1}{1 + e^{-2\beta(\hat{\eta}_t(y) - 0.5)}}\,,\end{aligned} \tag{21}$$

where $\beta = \frac{1}{2\mu}$.[17] In statistical mechanics, β is usually interpreted as the "inverse temperature." As Brock and Durlauf [3] suggest, in a socio-economic context it may be interpreted as the "intensity of choice": as β increases, agents' behavior is determined more by their intentions to behave optimally, and less by noise. In the limit, as $\beta \to \infty$, this implies that the best response function reduces to the equilibrium strategy of the Majority Voter Model, which assumes that there is no noise.

5.1 ISING MODEL WITH GLOBAL TRADE

As before, the behavior of the model under global trade (the "mean field approximation") can be used as simulation guide. Assuming that $\Pr(\eta_{t+1}(x) = 1) = \hat{\eta}_{t+1}$, the evolution of the system is given by the difference equation

$$\hat{\eta}_{t+1} = \frac{1}{1 + e^{-2\beta[\hat{\eta}_t - 0.5]}}\,. \tag{22}$$

[17] In the Ising model, the states of "agents" are typically denoted by $\omega_t(x) \in \{-1, +1\}$, as opposed to $\eta_t(x) \in \{0, 1\}$. Hence, in order to see that the two models are equivalent, a transformation $\omega = 2(\eta - 0.5)$ is needed.

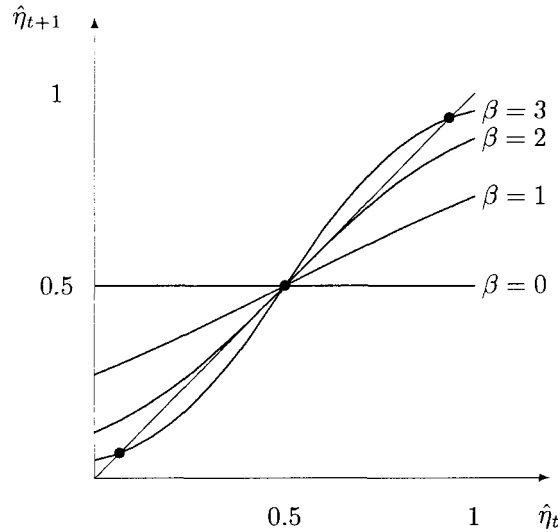

FIGURE 7 Best Response Functions for the Ising Model with Global Trade.

It can be easily checked that $\hat{\eta} = 0.5$ constitutes a steady state of this model, whereas $\hat{\eta} = 0$ or $\hat{\eta} = 1$ do not. The first derivative is given by:

$$\frac{\partial \hat{\eta}_{t+1}}{\partial \hat{\eta}_t} = 2\beta \left(1 + e^{-2\beta[\hat{\eta}_t - 0.5]}\right)^{-2}. \tag{23}$$

Evaluated at $\hat{\eta} = 0.5$, this yields $|J(0.5)| = \beta/2$, implying that this steady state becomes unstable for $\beta > 2$. As shown in figure 7, at this point two additional steady states emerge, which are stable. In terms of the NLVM, this corresponds to a phase transition from "ergodicity" to "clustering."

Apart from the fact that the steady states $\hat{\eta} = 0$ and $\hat{\eta} = 1$ no longer exist in this model, the mean field approximation of the $\beta = 3$ case is qualitatively similar to that of the $(p_1, p_2) = (0.3, 0.1)$ case in the NLVM. In both models, $\hat{\eta} = 0.5$ is an unstable steady state, but there exist two additional stable steady states, corresponding to a "largely market" and a "largely nonmarket" equilibrium. Following this analogy, this suggests that it may be possible for a phase separation regime to exist at the boundary between "clustering" and "ergodicity."

5.2 ISING MODEL WITH LOCAL TRADE

To test the hypothesis above, the Ising model was simulated on the Cellular Automata Machine (CAM8) at the Santa Fe Institute. Various numerical exper-

iments were carried out on a 512-by-512 square lattice, starting from symmetric random initial configurations.

As figure 8 illustrates, there does indeed exist a phase transition from "ergodicity" to "clustering," but not at $\beta = 2$, as predicted by the mean field approximation. As panel (a) shows, for β as high as 2.8, the system still looks essentially "ergodic," in the sense of being very close to random noise (even though, upon close inspection, tiny clusters can be detected). As β increases, signs of persistent inequality emerge. For $\beta = 3$, the clusters are small, but quite discernable and highly persistent (panel b). For $\beta = 3.1$, however, which is displayed in panel (c) after only 100 periods, the clusters eventually disappear, due to a process of surface-tension clustering which is shown in panels (d) through (f), for $\beta = 3.5$. For larger values of β, the system behaves increasingly more like the Majority Voter Model, as predicted by the mean field approximation.

While the behavior of the local trade model is thus qualitatively similar to that of the global trade model, the question remains as to why the critical value of β appears to be so much larger in the local case. One suggestion is that it is due to the fact that each agent is treated as her own "neighbor," implying that each period, an agent has at least a 20-percent chance of continuing to produce as before. It seems likely that this decreases the probability of clustering (since agents are less sensitive to the behavior of their neighbors), and hence a larger β is needed in order for to occur.

To capture the more general case where agents consume some fraction α of their own goods, the local trade structure may be redefined as a matrix $\mathbf{P}$ with elements

$$p(x,y) = \begin{cases} \alpha & \text{if} |y-x| = 0\,, \\ \frac{1}{4}(1-\alpha) & \text{if} |y-x| = 1\,, \\ 0 & \text{if} |y-x| > 1\,. \end{cases} \tag{24}$$

This gives the following best response function:

$$\Pr(\eta_{t+1}(x) = 1) = \frac{1}{1 + e^{-2\beta[\alpha\eta_t(x) + (1-\alpha)\hat{\eta}_t - 0.5]}}\,. \tag{25}$$

In any steady state, it must be true that $\eta_t(x) = \hat{\eta}_t$. Plugging this into the equation above, it can be seen that, in the global trade model, the steady state is independent of α. However, the global trade model is only a "first order" mean field approximation, in that it does not condition on each agent's current state. Taking this into account, the following "second order" approximation can be obtained:

$$\begin{aligned} \hat{\eta}_{t+1} &= \hat{\eta}_t \cdot \Pr(\eta_{t+1}(x) = 1 \mid \eta_t(x) = 1) \\ &\quad + (1 - \hat{\eta}_t) \cdot \Pr(\eta_{t+1}(x) = 1 \mid \eta_t(x) t = 0)\,; \qquad (26) \\ &= \hat{\eta}_t \left(1 + e^{-2\beta[\alpha + (1-\alpha)\hat{\eta}_t - 0.5]}\right)^{-1} \\ &\quad + (1 - \hat{\eta}_t) \left(1 + e^{-2\beta[(1-\alpha)\hat{\eta}_t - 0.5]}\right)^{-1} \qquad (27) \end{aligned}$$

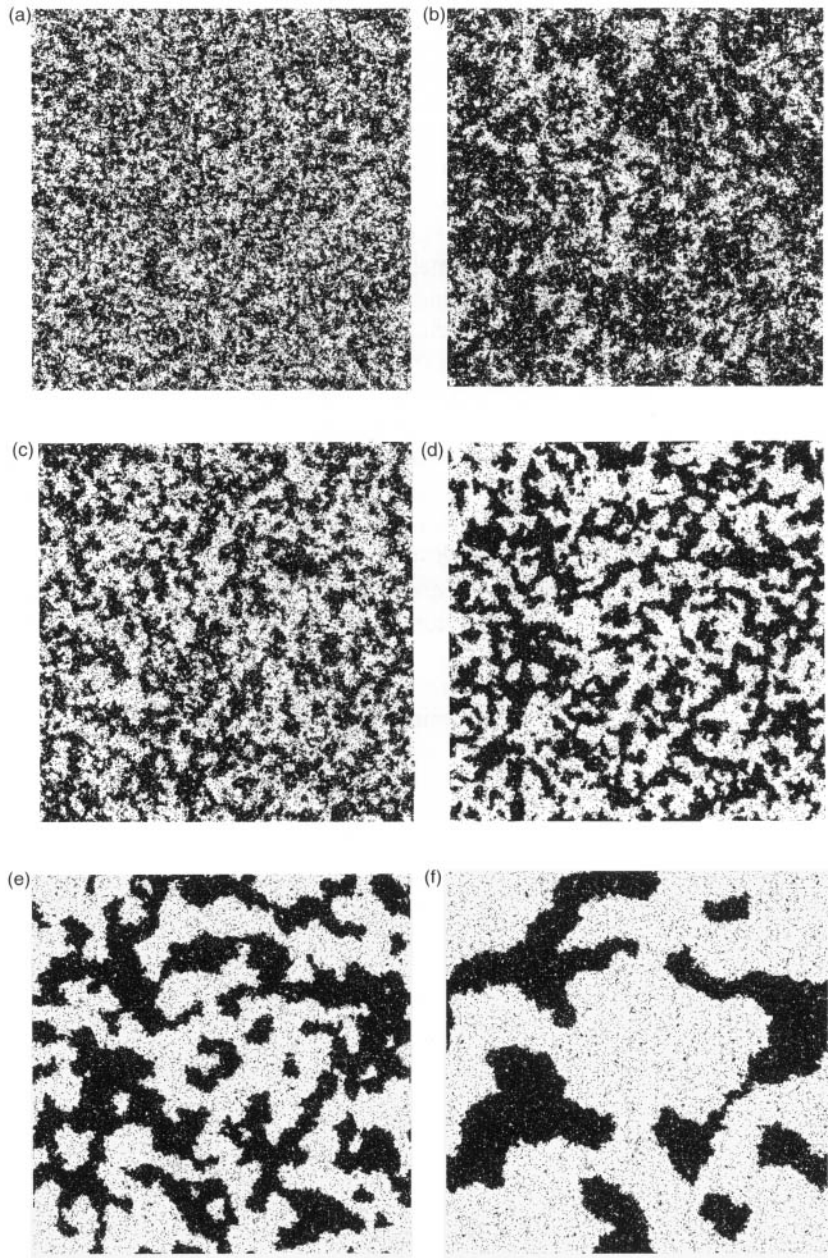

FIGURE 8 Phase Transition in the Ising Model. (a) $\beta = 2.8$, $t = 30,000$; (b) $\beta = 3$, $t = 50,000$; (c) $\beta = 3.1$, $t = 100$; (d) $\beta = 3.5$, $t = 100$; (e) $\beta = 3.5$, $t = 500$; (f) $\beta = 3.5$, $t = 3,000$.

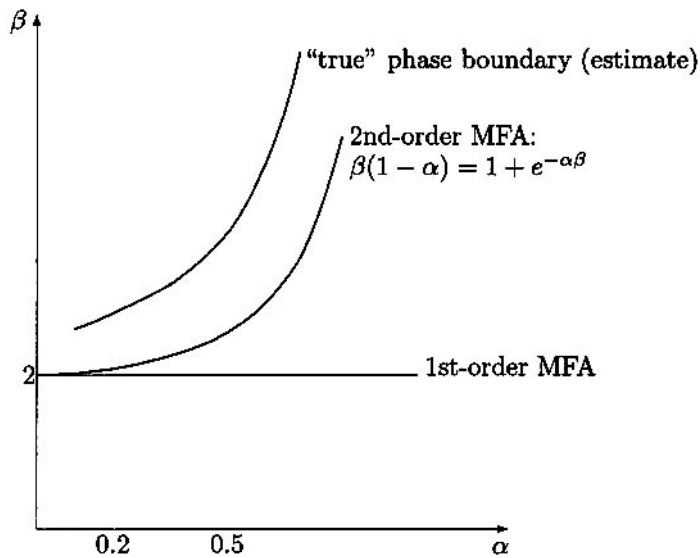

FIGURE 9 Phase Diagram for the Ising Model.

A phase transition now requires

$$|J(0.5)| = \left(1+e^{-\alpha\beta}\right)^{-1} + \left(1+e^{-\alpha\beta}\right)^{-2}\beta(1-\alpha)e^{-\alpha\beta} + \left(1+e^{\alpha\beta}\right)^{-1} + \left(1+e^{\alpha\beta}\right)^{-2}\beta(1-\alpha)e^{\alpha\beta} \geq 1, \tag{28}$$

which reduces to

$$\beta(1-\alpha) \geq 1 + e^{-\alpha\beta}. \tag{29}$$

This inequality confirms our intuition that, as α increases, larger values of β are required for clustering to occur. Moreover, as figure 9 shows, the second-order mean field approximation (MFA) is indeed quite close to the actual phase boundary estimated on the basis of numerical experiments.

These numerical experiments show that, for instance, for α around 0.5, the phase transition occurs around $\beta = 3.9$. Two examples of this are given in figure 10. Just as in the phase separation regime found by Molofsky et al. [15], the clusters that emerge here can become quite large, but they never take over the entire lattice.

The intuition behind this is as follows: While the probability of working for the market conditional on living in a nonmarket neighborhood is very small, occasionally agents do decide to take a chance (when they draw a large noise term) and start producing for the market. Once they do so, this increases the

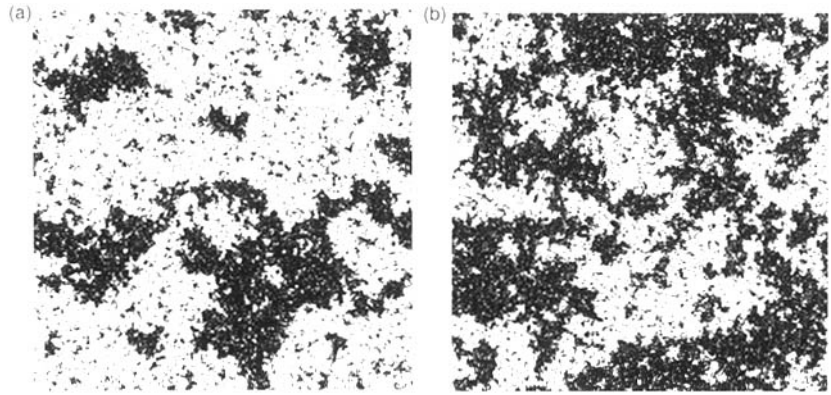

FIGURE 10 Persistence of Inequality in the Ising Model. (a) $\alpha = 0.5$, $\beta = 3.9$; (b) $\alpha = 0.51$, $\beta = 3.9$.

incentive of their neighbors to do the same. And once those neighbors become involved in market production, the neighbors of those neighbors are more likely to do so. Thus it is possible in principle for a cluster of market workers to develop within a nonmarket economy.

Of course, the probability that this happens in any given location is extremely small, since agents who become market producers by chance will tend to fall back to nonmarket production as soon as they find out that there are no trade partners. However, the probability that it happens *somewhere* within a nonmarket cluster increases with α (which makes agents fall back less quickly), and increases with the size of the cluster. This implies that, when β is large enough (relative to α) to produce large clusters, but small enough to allow for a reasonable amount of noise, it should be possible for the process to continue indefinitely, growing clusters within clusters within clusters.

6 CONCLUSION

The requirement that the probability for a given agent to be of a certain type is monotonically increasing in the frequency of this type, whether it be local or global frequency, seems to be a reasonable restriction in any model with positive frequency dependence. As has been shown in this chapter, under this restriction it is impossible for inequality to persist in the NLVM studied by Molofsky et al. [15]. However, persistent inequality was shown to arise in a more general NLVM that satisfies the restriction, and includes the Ising model as a special case.

So far, the analysis has been constrained to the two extreme cases of perfectly global and perfectly local trade. Plans for future work include the study of more general trade networks, the structure of which may be estimated on the basis of actual bilateral trade flow data, as well as exploring the implications of "globalization" for the persistence of inequality both in space and in time.

On the basis of results by Watts and Strogatz [18] it is to be expected that the marginal effects of globalization on "efficiency" are much higher than its marginal effects on reducing inequality. The reason for this is that, in a system with a regular and local interaction topology, the number of "degrees of separation" between agents (which could be interpreted as a measure of efficiency) grows linearly in the number of agents, while for a random and global topology this growth is only logarithmic. This suggests that adding only a few global traders to an initially localized economy is enough to obtain near-efficiency results, but not sufficient to eliminate locally persistent inequality. This hypothesis still remains to be tested.

ACKNOWLEDGMENTS

I would like to thank Steven Durlauf, Giorgio Fagiolo, David Griffeath, Mark Hopkins, Brian Krauth, Larry Samuelson, William Sandholm, and participants of the 1998 Santa Fe Institute Workshop in Computational Economics for valuable comments and suggestions. Special thanks to Cosma Shalizi for programming assistance, and to the Santa Fe Institute for its hospitality.

REFERENCES

[1] Balassa, B. "Intra-Industry Trade among Exporters of Manufactured Goods." In *Imperfect Competition and International Trade: The Policy Aspects of Intra-Industry Trade*, edited by D. Greenaway and P. Tharakan. Brighton: Wheatsheaf Press, 1986.

[2] Bergstrand, J. "The Gravity Equation in International Trade: Some Microeconomic Foundations and Empirical Evidence." *Rev. Econ. & Stat.* **67** (1985): 474–481.

[3] Brock, W., and S. Durlauf. "Discrete Choice with Social Interactions I: Theory." SSRI Working Paper 9521, University of Wisconsin-Madison, 1995.

[4] Clifford, P., and A. Sudbury. "A Model for Spatial Conflict." *Biometrika* **60** (1973): 581–588.

[5] Cooper, R. *Coordination Games. Complementarities and Macroeconomics.* Cambridge, MA: Cambridge University Press, 1999.

[6] Cooper, R., and A. John. "Coordinating Coordination Failures in Keynesian Models." *Quart. J. Econ.* **103(3)** (1988): 441–463.

[7] Crutchfield, J. "The Calculi of Emergence: Computation, Dynamics, and Induction." *Physica D* **75** (1994): 11–54.

[8] Deardorff, A. "Determinants of Bilateral Trade: Does Gravity Work in a Neoclassical World?" In *The Regionalization of the World Economy*, edited by J. Frankel, 7–22. Chicago, IL: University of Chicago Press, 1998.

[9] Durrett, R. *Lecture Notes on Particle Systems and Percolation.* Pacific Grove, CA: Wadsworth, 1988.

[10] Durrett, R., and S. Levin. "The Importance of Being Discrete (and Spatial)." *Theor. Pop. Biol.* **46** (1994): 363–394.

[11] Eichengreen, B., and D. Irwin. "The Role of History in Bilateral Trade Flows." In *The Regionalization of the World Economy*, edited by J. Frankel, 33–57. Chicago, IL: University of Chicago Press, 1998.

[12] Frankel, J., ed. *The Regionalization of the World Economy.* Chicago, IL: University of Chicago Press, 1998.

[13] Holley, R., and T. Liggett. "Ergodic Theorems for Weakly Interacting Systems and the Voter Model." *Ann. Prob.* **3** (1975): 643–663.

[14] Liggett, T. *Interacting Particle Systems.* New York: Springer-Verlag, 1985.

[15] Molofsky, J., R. Durrett, J. Dushoff, D. Griffeath, and S. Levin. "Local Frequency Dependence and Global Coexistence." *Theor. Pop. Biol.* **55** (1999): 270–282.

[16] Oomes, N. A. "Trade Networks and Persistent Unemployment." *J. Econ. Dynamics & Control* (2001): forthcoming.

[17] Samuelson, L. *Evolutionary Games and Equilibrium Selection.* Cambridge, MA: MIT Press, 1997.

[18] Watts, D., and S. Strogatz. "Collective Dynamics of 'Small-World' Networks." *Nature* **393(4)** (1998): 440–444.

Cellular Automata for Imaging, Art, and Video

Joy V. Hughes

1 INTRODUCTION

The techniques known as Cellular Automata (CA) can be used to create a variety of visual effects. As the state space for each cell, 24-bit photo realistic color was used. Several new state transition rules were created to produce unusual and beautiful results, which can be used in an interactive program or for special effects for images or videos.

This chapter presents a technique for applying CA rules to an image at several different levels of resolution and recombining the results. A "soft" artistic look can result.

The concept of "targeted" CAs is introduced. A targeted CA changes the value of a cell only if it approaches a desired value using some distance metric. This technique is used to transform one image into another, to transform an image to a distorted version of itself, and to generate fractals.

The author believes that the techniques presented can form the basis for a new artistic medium that is partially directed by the artist and partially emer-

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

gent. Images and animations from this work are posted on the World Wide Web at (http://www.scruznet.com/~hughes/CA.html).

2 24-BIT COLOR SPACE

All cellular automata (CA) operate on a space of discrete states. The simplest CAs, such as the Game of Life, use a 1-bit state space. Most modern personal computers represent color as a 24-bit value, allowing for approximately 16 million possible colors. The work presented in this chapter uses a 24-bit color space that is represented in a 32-bit-long integer.

This color space can be conceptualized as a three-dimensional bounded continuous vector space.

Often, it is desirable to work with in the HSV (Hue, Saturation, Value) color space. Some of the rules encode the value (luminance) of a cell in the otherwise unused 8 high-order bits of a 32-bit word. The hue and saturation can be estimated "on the fly" with simple, fast algorithms. The hue is represented as an angle on the color wheel.

For some rules, it is necessary to know the "distance" between two colors. Estimating the distance in perceptual space would be a difficult problem, as it would be dependent on the monitor used and the gamma exponent applied for a particular setup. In practice, a "Color Manhattan Distance" is used, where sum of the absolute values of differences between each component is used.

$$\text{Color Manhattan Distance} : |r2 - r1| + |g2 - g1| + |b2 - b1| .$$

This metric can be calculated very efficiently, without the need for a square root.

3 24-BIT RULES

Several new CA rules were created to take advantage of the huge range of possible rules over a 24-bit color space. A few selected rules are reviewed below.

3.1 BOX BLUR AND DIAGONAL BOX BLUR

The "Box Blur" is a 3×3 neighborhood rule that blurs pixels in either the horizontal or the vertical direction. The right and left neighbors are compared, and the color Manhattan distance between them is computed. The same calculation is performed for the top and bottom neighbors. The pair of pixels that are closest in color space are averaged, and the value of the center pixel is set to this average. The effect is to blur regions either horizontally or vertically, depending on their local color gradient. Areas that are blurred in one direction tend to be

blurred in the same direction again, as application of the rule tends to decrease the local gradient in the preferred direction, making it more likely that the same direction will be selected again.

"Diagonal Box Blur" works like "Box Blur," but blurs on the left and right diagonals instead of horizontally and vertically. This rule creates a checkerboard arrangement of colors, since information travels like a bishop in the game of chess, which cannot move from white squares to black squares or vice versa (Plate 1).

3.2 HUE BOIL

"Hue Boil" is a rule that selects two random pixels from a 2×2 (Margolis) neighborhood. The hue angles of the two pixels are compared, and the one with a hue angle less than 180 degrees to the right of the other copies the color of the other. This is a very high energy rule, as colors will change constantly anywhere the color is not constant. The visual effect is that of a roiling sea of colors. Zhabotinskyesque spirals can result (Plate 2).

3.3 MEDIAN MANHATTAN MOLD

"Median Manhattan Mold" operates over a 3×3 neighborhood. Three randomly selected neighbors of the center pixel are chosen, and their color Manhattan distances to the center pixel are computed. The color of the pixel with the median distance is copied to the center. The term "mold" is applied as this rule tends to result in irregularly shaped domains of color.

3.4 MUTANT CRYSTAL MOLD AND MUTANT CRYSTAL SMOKE

"Mutant Crystal Mold" is a relative of "Median Manhattan Mold." This rule was the result of a software bug, and as such its internal workings are not completely understood by the author. It is also the author's favorite visual effect. The scene tends to break up into small domains of color with smooth edges. In areas of sharp contrast, the colors tend to change to ones not found in the original image. Black-and-white or grayscale colors will not be transformed into saturated colors; however, a small region of color will expand across an otherwise monochrome image (Plate 3).

"Mutant Crystal Smoke" uses the same algorithm as "Mutant Crystal Mold," but is restricted to modify pixels only in the vertical direction. The effect resembles crystals growing underwater.

4 MULTIRESOLUTION CA

Most cellular automata operate over a fixed neighborhood of cells within a discrete space. The author has borrowed from the field of imaging to create a

multiresolution representation, or mip-map, of the grid of cells. The image is resized using a pyramidal resampling kernel to create a subimage at half the size of the original. The filter is applied recursively to give a series of subimages at one-quarter, one-eighth, and other integer powers of two. Because this step must be done at each time step, the implementation of the filter must be very fast. The coefficients of the kernel are all powers of two, so that bit shifting can be used to implement such a kernel efficiently.

Here is a pyramidal resampling kernel (weights shown are 16 times actual values):

$$\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$

When the resampled images are available, a CA rule is run on a specified number of levels. The same rule can be used on different levels, or different rules may be used. For example, a blur filter can be used on the highest resolution level, while sharpen is used on lower resolutions (Plate 4). The user has control over how many levels the rule or rules are applied to. The subimages prior to the iteration are stored using a double-buffering technique to be used in the recombination calculation.

A multiresolution CA rule is calculated as follows: The rule is applied first to the lowest-resolution (smallest) subimage. The image before iteration is subtracted from the image after iteration, giving the amount of change for each pixel (the "delta image"). The delta image is resampled to twice its size and added to the next higher-resolution subimage. The results of the iteration are thus "smoothed," while higher-resolution details are preserved. The rule is then applied to the result, and a new delta image calculated. These steps are repeated until the full-resolution image has been processed.

The entire process can be accomplished quite quickly, but, of course, the result will be slower than the simple application of a CA rule to an image. The rule must be iterated over the resampled hierarchy of images, which imposes a penalty of up to 33% in speed. The recombination calculations add a constant overhead as well (about 1/10 second on a 300 MHz PowerPC for a 400×400 pixel image). While this overhead is significant, interactive speed can still be achieved.

When multiresolution rules are applied to an image, a number of things can happen. Some rules generate a "softened" version of the single-resolution rule, while others display emergent properties that don't resemble the single-resolution rule at all.

5 TARGETED CAs

A CA can be constrained in its behavior to produce interesting visual effects. Each pixel in the image has a "target" color. When a rule is iterated over the im-

age, only those pixels with values that are closer to the target by some metric are updated. The target values are generally stored in a "target image," which may be the starting image, another image chosen by the user, or the image undergoing iteration. The target image may be modified using a "target transform," which causes each pixel in the iterating image to target a pixel in the target image whose coordinates are determined by some mathematical function.

If the target value for a given pixel does not change over time, the CA will eventually fall into a steady state, where repeated application of the rule does not alter the image. The result is often something close in appearance to the target image, but with visible differences depending on the rule, metric, and starting image. If the target value changes over time, or is dependent on the values of pixels within the image being iterated, the image may continue to change indefinitely.

5.1 TARGET METRICS

To determine the distance to a target color, a number of different metrics can be used, each of which produces its own unique visual effects. One simple metric is to use the luminance value of the pixel (a weighted average of 60% red, 30% green, and 10% blue). Use of this metric will result in an approximation of the target image with colors of the appropriate brightness, but possibly wildly different hues. Hue can be used, or hue can be offset by some desired angle. The result will be an image with the appropriate colors (or with a different "tint" if an offset is used), but with variable brightness. Another useful metric is the Manhattan color distance, which takes into account all three components of a color.

One metric that is quite useful for artistic effects is to try to approximate the inverse of a color, where each component of the original color is subtracted from the maximum possible value (255). This can produce particularly striking effects when self-targeting.

5.2 CA RULES FOR IMAGE TARGETING

Any of the rules described in section 4 can be used to approach a target, and some can be quite visually interesting (Mutant Crystal Mold is a favorite of the author). There are some rules that are only useful when trying to approach a target.

5.2.1 Random Neighbor Copy. One simple method is to choose a random pixel from the eight neighboring pixels, and copy its color into the center pixel if closer to the target color, using the target metric. If this rule is used, the result often appears to be a "posterized" version of the target, where single pixels expand to cover larger regions. The target image is approximated, but an exact match is not possible, since the colors are not modified, only copied (Plates 5 and 6).

A more accurate result can be obtained by choosing two or more neighboring pixels at random, averaging their values, and determining if the result is closer to the target.

5.2.2 Error Diffusion. Another rule specific to image targeting is error diffusion. The image is divided into pairs of pixels. A 2×2 (Margolis) neighborhood is used, and each 2×2 block is divided into two pairs of pixels. The blocks are offset by one pixel both horizontally and vertically every other time step, to allow a pixel to pair with all its neighbors. A horizontal, vertical, or diagonal arrangement of pixel pairs is chosen at random. For each pixel in a pair, the per component difference between the pixel color and the target color is determined. These "error values" are equalized between the two pixels by increasing the value of a component in one pixel while decreasing the value of the same component in the other by the same amount. For example, if one pixel is five units of red above its target value, and the other pixel is one unit of red above its target value, the red value of the first pixel will be decreased by two units while the red value of the second pixel will be increased by two units. The red values of both pixels will then both be three units above their target values.

Repeated application of error diffusion causes colors to appear to "flow," approximating the target image. High-frequency details appear first, then the image gradually approaches the colors of the target image. An exact match usually will not occur, since the total amount of each color in the image is conserved. There just might not be enough red to go around (Plate 7).

5.2.3 Color Mating. To calculate the new pixel value, two neighboring pixels are chosen. The red, green, and blue components of each pixel are mixed at random, and the recombined color is compared with the target value. This is a crude simulation of biological sexual reproduction. A "mutation" parameter can also be included, which randomly perturbs the pixel values by some amount. A target image can be approximated much better than with random neighbor copying, as many more colors can be constructed with recombination. If mutation is included, a target image can be reached with complete accuracy (although this might take an extremely long time) (Plate 8).

5.3 TARGET TRANSFORMS

The target color of a given pixel is assigned to the color of some pixel in the target image. The target image can be the original image at the start of iteration, a separate target image selected by the user, or the iterating image itself. The coordinates of the target pixel are chosen using the target transform, which is some mathematical function. The transform is represented by an image of pointers to target pixels, so the target function does not need to be recalculated each frame (unless it is time dependent).

The simplest target transform is the identity transform, where the coordinates of a target pixel are the same as those of the iterating pixel. The identity transform can be used for image transitions. Reflection or rotation can be used as a target transform. Rotation can be applied in a time-dependent manner, where the rotation angle changes overtime. This gives two distinct domains of behavior in the image. Near the center, the target pixels are changing slowly, and information can travel between pixels that have the same target values in different frames. Farther from the center, the speed of rotation of the target image exceeds the speed of light, and information cannot travel to pixels with the same target value.

Nonlinear functions can also be used, such as a spiral-type function, where a rotation angle is applied, and the angle increases with distance from the center. Turbulence is another interesting nonlinear function, where a number of small vortices are summed together, giving a wavy appearance (Plate 6).

A pixel's coordinates can be mapped to a value in the complex plane, and complex functions used to determine the target pixel's coordinates. Polynomial, exponential, and trigonometric functions have been used. These can produce strangely warped versions of the target image and, when self-targeting, fractals emerge.

5.4 APPLICATIONS OF TARGETED CA

5.4.1 Pseudo-Morphing. If the original image is used as the target image, and a target transform applied, the image will appear to change to a transformed version of itself. This can produce interesting artistic effects, such as a face transforming into a mirror image of itself, or a warped version of itself. While not technically the same as morphing, the visual effect can be similar (Plate 6).

5.4.2 Image Transitions. To accomplish an image transition, a target image is chosen which is different from the original image. As the CA iterates, the iterating image will appear more similar to the target image. This can be very interesting visually, as old forms seem to dissolve and new forms emerge (Plate 5).

5.4.3 Self-Targeting. A variety of interesting effects can result from setting the iterating image as a target. If reflection is used as a target transform, a symmetrical image will result. A rotation transform will result in an image with rotational symmetry. Angles which do not evenly divide the circle will produce a "resonance," which may produce regions of different symmetries. For instance, a ring of 13-fold symmetry might surround a central region of 5-fold symmetry. Sometimes different symmetries will alternate in an unstable way, with 3-fold and 4-fold symmetries dominating at different times (Plate 9).

An interesting result was obtained using a rotated target transform and the hue metric with an offset in the hue "angle." A resonance will occur in color

space and anglular space, and symmetries as high as 27-fold have been observed (Plate 10).

If a complex plane function is used for the target transform, Julia Sets (see fractals) will arise while self-targeting. Painting on the fractals will make the paint marks "echo" in other parts of the image (Plates 7 and 11).

5.4.4 False Color. The original image can be used as a target with no target transform. In this case, no change will be observed until the image is modified using paint tools. Then, the image will appear to "heal" the marks made on it. Paint will appear to flow into areas where the image has similar colors, leaving behind a false-colored image (Plate 12).

6 INTERACTION WITH THE CA

Mark Zimmer at Meta Creations developed an interactive program which allows the user to modify many of the internal properties of CA rules using sliders and buttons. Each rule has different properties which may be changed by the user, so a particular control may have different effects depending on the active rule. Images may be loaded in, or the current image may be saved out. The user may select from a table of possible rules and target transforms, and select the number of levels of resolution to which the rule should be applied.

The user can "paint" into the CA using a brush tool. This allows an artist to use the CA as an artistic tool that is partially directed and partially emergent (Plate 12).

7 VIDEO PROCESSING

The CA software developed at Meta Creations allows the user to save a sequence of images as a compressed digital movie. The frame rate is reduced significantly during storage, but playback can occur at full speed, possibly even faster than the CA rule iterates.

CA rules can be applied to each frame of a video. The user selects a movie, a rule, and the number of iterations that should be applied to each frame. The software automatically loads each frame, applies the appropriate number of iterations, and saves the result into a new movie.

8 CONCLUSION

Cellular automata can be used to create visual art which can be both strange and beautiful. A new artistic medium has been demonstrated that is neither totally directed by the artist nor totally algorithmic. 24-bit color spaces with new CA

rules, interactive tools, multiresolution modifications to rules, and the concept of targeted CAs are all ways of exploring the vast and visually fascinating world of cellular automata.

ACKNOWLEDGMENTS

This work was sponsored by Meta Creations from the fall of 1997 to the spring of 1998. Mark Zimmer (Chief Technology Officer) wrote the application framework under which my work was done, and a number of cellular automata effects. Meta Creations owns the code and images produced in this project.

Many thanks to Rudy Rucker for inviting me to the Cellular Automata Workshop at the Santa Fe Institute and providing me with a forum for this work.

This page intentionally left blank

Continuous-Valued Cellular Automata in Two Dimensions

Rudy Rucker

We explore a variety of two-dimensional continuous-valued cellular automata (CAs). We discuss how to derive CA schemes from differential equations and look at CAs based on several kinds of nonlinear wave equations. In addition we cast some of Hans Meinhardt's activator-inhibitor reaction-diffusion rules into two dimensions. Some illustrative runs of *CAPOW*, a CA simulator, are presented.

1 INTRODUCTION

A *cellular automaton*, or CA, is a computation made up of finite elements called cells. Each cell contains the same type of state. The cells are updated in parallel, using a rule which is homogeneous, and local.

In slightly different words, a CA is a computation based upon a grid of cells, with each cell containing an object called a *state*. The states are updated in discrete steps, with all the cells being effectively updated at the same time. Each cell uses the same algorithm for its update rule. The update algorithm

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

computes a cell's new state by using information about the states of the cell's nearby space-time neighbors, that is, using the state of the cell itself, using the states of the cell's nearby neighbors, and using the recent prior states of the cell and its neighbors.

The states do not necessarily need to be single numbers, they can also be data structures built up from numbers. A CA is said to be *discrete valued* if its states are built from integers, and a CA is *continuous valued* if its states are built from real numbers.

As Norman Margolus and Tommaso Toffoli have pointed out, CAs are well suited for modeling nature [7]. The parallelism of the CA update process mirrors the uniform flow of time. The homogeneity of the CA update rule across all the cells corresponds to the universality of natural law. And the locality of CAs reflect the fact that nature seems to forbid action at a distance.

The use of finite space-time elements for CAs are a *necessary evil* so that we can compute at all. But one might argue that the use of discrete states is an *unnecessary evil.* In the old days, speed and storage considerations made it impractical to carry out large CA computations using real numbers as the cell states, but today's desktop machines no longer have these limitations. The author and his students have developed a shareware software package for Windows called *CAPOW*, which we have used for exploring continuous-valued CAs [6]. The paper by Ostrov and Rucker [5] contains information about our investigations of one-dimensional continuous-valued CAs, and the present chapter presents some of the phenomena found in two-dimensional continuous-valued CAs.

In the CAs we have been investigating, we take "real number" to mean IEEE single-precision floating-point number, what the C language terms a *float* rather than a *double.* We have experimented with double-precision floating-point numbers, but their use does not seem to change the qualitative features of our simulations. Double-precision floating-point numbers have the drawback of requiring larger memory buffers and of cutting simulation speed. Because of considerations of speed and memory, we have been looking at relatively small two-dimensional CAs, with a dimension of 120 cells wide by 90 cells high.

In section 2 of this chapter we discuss how we derive CA schemes from sets of differential equations and section 3 presents some material relating to our specific methods of simulation. In section 4 we discuss some two-dimensional continuous-valued CAs that are based on reaction-diffusion systems that use an activator-inhibitor reaction. In section 5 we look at CAs based on linear and nonlinear wave equations and in section 6 we briefly consider the possibility of developing some "reaction wave" CAs. And section 7 suggests some paths for further investigations.

Before proceeding, let us confront three possible objections to the study of continuous-valued cellular automata.

TABLE 1 The longevity of a linear wave scheme using varying minimum sizes of real number.

Coarseness of "Reals"	Number of updates until a waves dies out
0.1	50 updates
0.01	200 updates
0.001	800 updates

Objection 1. *Since you are running your computation on a digital machine, your so-called continuous values are really discrete numbers, so you are doing nothing new.*

Over typical lab scales of minutes and hours, there is a qualitative difference between a CA whose state is only a few bits, and a CA whose state is a floating-point number. You can indeed simulate crude things like heat flow with only a few hundred discrete states, but numerical viscosity kills off subtler continuum behaviors like wave motion. With states that are single-precision floating-point numbers, simulation of a one- or two-dimensional wave will persist through millions of updates, but if we coarsen the grain down to something like ten bits of state, a wave simulation quickly dies out. Table 1 shows results obtained by simulating a one-dimensional wave with the *CAPOW* software, which contains a "State Grain" control for altering the coarseness of the real numbers used.

Objection 2. *When you use floating-point numbers, computational round-off destroys the possibility of having time-reversible rules.*

This issue was raised by Norman Margolus during the "Constructive CAs" conference. Margolus reasons that since physics is reversible, the CAs we use should be reversible as well. Margolus' concern about the *CAPOW* program was that its use of floating-point numbers for its "real numbers" would make the rules irreversible due to computational round-off.

Tests conducted since the conference reveal that the computational round-off is not noticeable enough to destroy the possibility of rule reversibility over the simulation runs that we use, typically on the order of a thousand to a hundred-thousand updates. Figure 1 shows an example of a one-dimensional wave equation rule being reversed.

Of course our rules can only be reversible when they are based on a reversible scheme such as the Wave Scheme introduced in section 3. The Diffusion Scheme which we use is inherently irreversible. Although it is possible to model diffusion in terms of the reversible motions of a deterministic gas of "heat particles," such

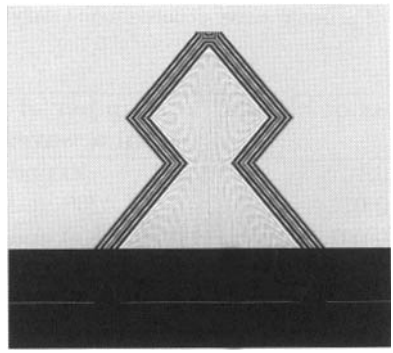

FIGURE 1 A one-dimensional wave schema being run with a *Wave* constant of 0.694. The rule was seeded with a single triangular spike and was time-reversed twice. The figure shows an instantaneous snapshot of the cells' intensity values at the bottom, while the upper part shows a space-time diagram of the intensity patterns, with the earlier times at the top and the later times at the bottom. This simulation was "time reversed" by exchanging the "past" and "future" cell buffers.

a strategy would seem to limit us to simulating systems much smaller and simpler than those investigated here.

Objection 3. *To study continuous-valued CAs is to repeat existing work in numerical analysis and finite element simulations.*

Finite element methods are indeed an inspiration for continuous-valued CAs. But the CA approach has a different mindset and leads to different kinds of investigations. The CA approach involves: an emphasis on *experiment* and observation rather than on theory and proof; an *artificial life* orientation in which one is actively on the lookout for unexpected and emergent properties of the simulation; and the use of *genetic algorithm* methods for effectively searching the large phase spaces of the rules. A historical distinction between CA work and finite element simulations is that the latter tend to be run on supercomputers, while CA programs are usually rapidly running, attractive, interactive shareware graphics programs for desktop machines. It is our hope that thinking in terms of continuous-valued CAs can lead to a productive unification of work from diverse fields.

2 CONTINUOUS-VALUED CAs AND DIFFERENTIAL EQUATIONS

Most of the continuous-valued rules we have investigated so far have been inspired by systems of differential equations. These include one- and two-dimensional forms of equations based on diffusion, wave motion, oscillators, activator-inhibitor reactions, and various combinations of these equations.

Our cell states typically include a real-number value u, and our CA update scheme typically has the form $uNew = Update(u, \ldots)$. Various neighbor-state values can appear as arguments to the Update rule. We write *uNew* rather than u on the left because in order to preserve parallelism we think in terms of first computing the *uNew* values for every cell before then starting to view these as the current u values.

In working with CA schemes of this nature, one needs to worry both about the numerical accuracy of a scheme and about its stability [5]. The accuracy relates to how well the CA is simulating an actual differential equation. The stability relates to whether or not the CA simulation goes completely out of control. When a rule enters an unstable regime it will generally produce arbitrarily large and small values. A good heuristic in seeing if a scheme is likely to be stable is that it should set *uNew* to something the size of u plus something the size of a constant times the difference between two cell values.

In order to discuss our practices in converting systems of equations into schemes for CA rules, let us suppose we are looking for a scheme to be used at a given cell C. In one-dimensional rules, we call the cells left and right neighbors L and R. In two-dimensional rules, we call the cell's neighbors *E, NE, N, NW, W, SW, S,* and *SE.* In two dimensions we distinguish between the Moore neighborhood of all eight neighbors, and the von Neumann neighborhood of only the four neighbors E, N, W, and S.

Table 2 lists some symbols we will use for various cell neighborhood values. We will use *uPast* and *uNew* for the value in the cell at, respectively, the prior and the following update. It is going to be useful to use the term *uTimeAvg* for the average of these two "time neighbors." And we will use *uNabeAvg* for the average state values in C's immediate neighbors, excluding C itself. In addition, we use uR and uL for the u values in the cell's right and left neighbors R and L, and use uE, uN, uW, and uS for the intensity values of the update cell's four von Neumann neighbors, and so on.

In a one-dimensional case where we only look at nearest neighbors, *uNabeAvg* is $(uL + uR)/2$. In a two-dimensional case where we use the von Neumann neighborhood, *uNabeAvg* is $(uE + uN + uW + uS)/4$. And in the two-dimensional Moore neighborhood case, we choose to weight the corner cells a bit less, and use a *uNabeAvg* of $((uE + uN + uW + uS) \times 0.75 + (uNE + uNW + uSW + uSE))/7$.

TABLE 2 Notation for cell neighborhood values.

Symbol	Meaning	Compute as:
u	Current value of cell state	
$uPast$	Prior value of cell state	
$uNew$	Next value of cell state	
$uNabeAvg$	Average of space neighbors	$(uL + uR)/2$ or $(uE + uN + uW + uS)/4$ or $((uE + uN + uW + uS) \times 0.75 + (uNE + uNW + uSW + uSE))/7$
$uTimeAvg$	Average of time neighbors	$(uPast + uNew)/2$

TABLE 3 Some CA approximations.

u_t	$uNew - u$
u_{tt}	$uTimeAvg - u$
$\nabla^2 u$	$uNabeAvg - u$

To convert a differential equation into a CA scheme, we write the equation in a form that uses expressions of the form u_t or u_{tt} as opposed to expressions of the form $\partial u/\partial t$ or $\partial^2 u/\partial t^2$. And we use the dimension-independent $\nabla^2 u$ for, in one dimension, u_{xx} or $\partial^2 u/\partial x^2$, and for, in two dimensions, $u_{xx} + u_{yy}$ or $(\partial^2 u/\partial x^2 + \partial^2 u/\partial x^2)$. And then we use the substitutions in table 3. Note that we do not worry about putting in any Δx, Δy, or Δt terms because values for these terms can end up being incorporated into one of the CA scheme's parameters.

Consider a diffusion equation with a parameter called *Diffusion*.

$$u_t = Diffusion \times \nabla^2 u$$

becomes (Diffusion Scheme)

$$uNew = u + Diffusion \times (uNabeAvg - u)\,.$$

Note that this scheme satisfies the stability heuristic mentioned at the start of this section; that is, $uNew$ is u plus something the order of a difference between the u values of two neighbors.

The Diffusion Scheme can be thought of as taking a weighted average of the cell and its neighbors. That is, we can write the Diffusion Scheme as $uNew = (1 - Diffusion) \times u + Diffusion \times uNabeAvg$. As the *Diffusion* parameter ranges from zero and unity, the weight shifts from the cell to its neighbors. If the number of neighbor cells is k, then using a *Diffusion* value of $k/(k+1)$ makes a straight average of the cell and its neighbors.

One should not use a *Diffusion* value greater than one, as this leads to instability. The reason for this becomes clear if we consider a situation where the *uNabeAvg* is zero. In this case, if *Diffusion* were greater than one, then a single update would change a positive *u*to a negative *uNew*.

Now consider a wave equation with a parameter called *Wave*.

$$u_{tt} = Wave \times \nabla^2 u$$

becomes (Wave Scheme)

$$uNew = (2 \times u - uPast) + 2 \times Wave \times (uNabeAvg - u)\,.$$

Our stability heuristic is satisfied because the first term on the right side is of the order of u, while the second term is the order of a difference in the u values of two neighbors.

In terms of the differential equations, the *Wave* parameter is actually $(c \times \Delta t/\Delta x)^2$, or the square of the speed of the wave in the medium times the square of the time step divided by the square of the space step [5]. For a cellular automatist it is more practical to just think of the single parameter *Wave*.

One significant thing to notice about the Wave Schema is that it is time reversible. As already mentioned in section 1, we can swap the positions of *uNew* and *uPast* to get a scheme that retrodicts the past instead of predicting the future. In practice, we reverse a running Wave Schema CA by exchanging the roles of the buffers that hold the cells' past values and the cells' future values.

We can also think of the right-hand side of the Wave Scheme as being two times a weighted average of the cell and its neighbors with the old value of the cell being subtracted off. That is, we can write the Wave Scheme as $uNew = 2 \times ((1 - Wave) \times u + Wave \times uNabeAvg) - uPast$. As the *Wave* parameter ranges from zero to one, the weight shifts from the cell to its neighbors, and values greater than one give instability.

Instead of deriving our schemes for continuous-valued CA rules from differential equations, it is also possible to develop new CA rules simply by playing with the schemes. Fermi et al. [3, 9] describe an early computer experiment in which they changed the one-dimensional wave equation rule by adding a *Nonlinearity* parameter and a factor that involves the squares of the differences between the neighboring cell values to produce this scheme. (See Weissert [10] for a history of the Fermi-Pasta-Ulam work through the mid 1970s.) As mentioned above, we write uR and uL to stand for the u values in the cell's right and left neighbors R and L.

(One-dimensional Quadratic Wave Scheme)

$$uNew = (2 \times u - uPast) + 2 \times Wave \times (uNabeAvg - u + Nonlinearity \times ((uR - u)^2 - (u - uL)2))$$

An analysis in Ostrov [5] establishes that in one dimension, this scheme corresponds to a nonlinear wave equation of the following form.

$$u_{tt} = Wave \times u_{xx} + 2 \times Nonlinearity \times u_x \times u_{xx} .$$

We will say more about nonlinear waves in section 6.

3 INVESTIGATING CONTINUOUS-VALUED CAs

As with other CAs, we simulate parallelism by maintaining separate buffers to hold the current cell values and the new cell values being computed. Since the Wave Schema CAs compute the future on the basis of values both from the present and the past cell values, we actually need to maintain three buffers for these rules.

In running CAs with continuous-valued state variables, one needs to prevent the state values from taking on unreasonably large values that can produce floating-point overflow. A simple way to do this is to pick a range of values that you will allow the variables to lie in, and to then clamp them to stay in this range, where to "clamp" a variable u to a range (Min, Max) means that if u is less than Min we set it to Min, and if u is greater than Max we set it to Max. The brute-force clamping approach protects the integrity of the simulation in times when the rules have no physical analog.

As an alternate approach to keeping variable values in range, one can "wrap" them instead of clamping them. That is, if u is slightly larger than Max, one changes it to $u - Max$, and so on. Although this approach has the virtue of preserving more information about the value of u, it seems in some cases to have the disadvantage of excessively churning things up, and we have not used it very much. As a matter both of elegance and of physical verisimilitude, one usually tries to design the rules and their parameters so that no special measures are needed to keep the variables in range.

In order to display our CAs, we pick one of the state variables to be the display variable u. We then use a map called *Band* to map u's range onto the integers up to some largish number *MaxColorIndex* (typically 1000). We have been using simple linear maps for *Band*, although other kinds of maps could be useful, for instance to exaggerate the color changes near some critical value. We use a *Palette* array of *MaxColorIndex* of RGB color values and we display u as *Palette[Band(u)]*.

Our standard design for *Palette* is to randomly choose some "anchor colors" for some of the *Palette* entries, and then to use linear interpolation in RGB space

to ramp the entries whose indices lie between the indices of the anchor colors. This produces a smoothly shaded effect. When generating monochrome images, we simply alternate white and black for the anchor colors, producing a *Palette* which is a shaded series of gray-tone stripes.

We have experimented with a variety of possible boundary conditions for our CAs. The most commonly used is the periodic boundary condition, in which a one-dimensional CA space is treated as a circle and a two-dimensional CA space is treated as a torus. This is the only boundary condition used in the examples discussed in this chapter.

There are various possible ways to seed the continuous-valued CAs. Among them are: a constant starting value at all cells, a two-dimensional sine-wave pattern, a single tentlike spike, multiple spikes that the user can place with the mouse, and a fully randomized initial state.

Finding the best set of parameter values for a given rule is difficult. The search spaces are, after all, very large and perhaps chaotically organized. We have used an evolutionary search strategy that seems to have first been introduced by Richard Dawkins in his classic *Blind Watchmaker* program [2]. Like *Blind Watchmaker, CAPOW* allows the user to view nine rules at once, to select a visually appealing rule by clicking on it, and to thereby have the other eight rules become mutations of the chosen rule. The mutation rate is user-selectable, and there are other kinds of randomization options as well. In other words, many of the rules discussed here have been found by directed search methods. Color plate 13 shows an image of the *CAPOW* window with nine different CA rules active.

In searching for interesting rules, we use a refinement of Stephen Wolfram's familiar classification of one-dimensional CAs into four kinds: (I) those that die, (II) those that repeat, (III) those with nonrepeating persistent structures, and (IV) pseudorandom (see table 4). If we view this classification as a spectrum of increasing complexity, it seems logical to have the pseudorandom rules come last, even though Wolfram chose to list the last two classes in the opposite of the expected order. It is useful to distinguish between spatial and temporal periodicity in type II. In two dimensions we have a special kind of complex rule that, rather than exhibiting discrete gliders, shows the self-organizing scroll patterns identified with the Belousov-Zhabotinsky reactions in chemistry. With this in mind, we distinguish between two cases of type III as well.

4 REACTION-DIFFUSION SYSTEMS

Some of the most interesting patterns in nature come from reaction-diffusion systems in which a chemical reaction is taking place while the components of the reaction are being diffused. Much of the research in this area has focused on reactions which involve two substances: an autocatalyzing activator which also

TABLE 4 The complexity types of two-dimensional CAs.

Complexity type	Wolfram type	Attractor	Behavior
I	1	Point	Dies out
IIa	2	Cycle	Fixed space pattern
IIb	2	Cycle	Periodic cycle
IIIa	4	Strange	Self-organizing scrolls
IIIb	4	Strange	Moving gliders or globs
IV	3	Pseudorandom	Chaotic seething

produces an inhibitor substance. (See the classic Alan Turing paper [8] and the recent nontechnical survey [1].)

Hans Meinhardt [4] formulates several differential equation schemes for reaction-diffusion systems based on activator-inhibitor reactions. This book also includes a disk with the executable and the BASIC source code for Meinhardt's SP program, which displays continuous-valued one-dimensional CAs based on a wide range of activator-inhibitor-diffusion systems. Meinhardt's work was the major inspiration for our development of the *CAPOW* software.

The work in this section is based on Meinhardt's differential equation scheme for an activator-inhibitor diffusion rule with activator saturation. Depending on how the parameters are set, we can get every possible CA complexity type with the exception of IIIb.

We can easily find rules of type I, which rush to take on the maximum or minimum value for all cells, and remain frozen there.

The rules of type IIa are of particular interest. These rules converge to static-appearing patterns resembling the coats of animals such as leopards and zebras. These kinds of reaction-diffusion patterns are often called *Turing patterns*, as Turing's motivation for considering these rules was indeed to find ways to generate stable patterns which emerge in morphogenesis. The rules of type IIb show a uniform oscillation up and down. If these rules oscillate wildly enough to hit the maximum values and experience "clamping," then recurrent dot patterns are introduced by the clamping process.

The rules of type IIIa are those in which certain wavelike structures form and move about. Among these *traveling wave* patterns; of particular significance are those in which scrolls self-organize. The scroll-forming patterns are instances of the ubiquitous two-dimensional CA rules sometimes called *Zhabotinsky rules*. None of the rules of this kind investigated so far seem to show stable moving patterns that characterize complexity type IIIb.

And finally it is always easy to find a settings that produce type IV patterns which seethe wildly.

Meinhardt formulates these rules in terms of two real number variables a and b which represent the intensity of, respectively, the activator and the inhibitor. In our simulations we have typically let a and b range from 0 to 4, focusing on rules in which the a and b values never actually approach the maximum value of 4 closely enough to require clamping. We use a helper variable *bSafe* to prevent division by zero, along with a number of parameters that are named in table 5.

Meinhardt's activator-inhibitor equations have this form.
(Activator)

$$a_t = aDiffuse \times \nabla^2 a + \frac{sDensity a^2}{(b \times (1 + aSaturation \times a^2))} + sDensity \times aBase - aDecay \times a\,.$$

(Inhibitor)

$$b_t = bDiffuse \times \nabla^2 b + sDensity \times a^2 + bBase - bDecay \times b\,.$$

To model these as CA schemes, we treat the time step as 1 and use updates of the form $aNew = a + a_t$. The full activator-inhibitor CA rule takes the following form.

(Avoid division by 0) IF $(b > bMin)$ THEN $bSafe = b$ ELSE $bSafe = bMin$.
(Activator)

$$aNew = a + aDiffuse \times (aNabeAvg - a) + \frac{sDensity \times a \times a}{(bSafe \times (1 + aSaturation \times a \times a))} + sDensity \times aBase - aDecay \times a\,.$$

(Inhibitor)

$$bNew = b + bDiffuse \times (bNabeAvg - b) + sDensity \times a \times a - bDecay \times b\,.$$

(Clamp) Clamp both a and b to be in the range $[0, abMax]$.

Figure 2 shows an example of one of the self-organizing scroll CAs of type IIIa called Zhabo Worms, while color plates 2 and 3 show examples of stable Turing pattern CAs of type IIa called Turing Leopard and Turing Stripes. The parameter values used for these three rules appear in table 6.

5 WAVE EQUATIONS

The two-dimensional wave equation rules give patterns very similar to the surface of pan of water, although our use of periodic boundary conditions means that

TABLE 5 The variables and parameters for the activator-inhibitor-diffusion rule.

Symbol	Meaning	Comment
Variables:		
a	Concentration of the activator.	Typical range 0 to 4.
b	Concentration of the inhibitor.	Typical range 0 to 4.
Safety Variables:		
bSafe	Concentrartion of the inhibitor, corrected to be above *bMin*.	Typical range 0.001 to 4. Use to avoid division by 0.
Equation Parameters:		
sdensity	Source density, akin to a reaction rate.	Range 0 to 1. Small 0.01 Turing, medium 0.5 for for Zhabo.
aDiffuse	Diffusion rate of the activator.	Range 0 to 1. Needs to be close to *bDiffuse* for Zhabo.
bDiffuse	Diffusion rate of the inhibitor.	Range 0 to 1. Needs to be much larger than *aDiffuse* for Turing.
aBase	Basic activator production rate.	Small 0.01 for Turing, medium 0.3 for Zhabo.
bBase	Basic inhibitor production rate.	Small, about 0.004.
aDecay	Activator removal rate.	Large 0.5 for Zhabo, small 0.01 for Turing.
bDecay	Inhibitor removal rate.	Large 0.3 for Zhabo, small 0.01 for Turing.
aSaturation	Slows down the rate of activator production as a increases.	Need this to get good Turing patterns. Low values like 0.04 give dots, high values like 0.2 give stripes.

TABLE 5 Continued.

Symbol	Meaning	Comment
Simulation Parameters:		
Neighborhood	Dimensionality and neighborhood.	In two dimensions we get the best-looking, smoothest results with a Moore neighborhood with edges weighted slightly more than corners.
bMin	Minimum inhibitor in rule.	Small 0.001 for Turing, medium 0.5 for Zhabo.
abMax	Maximum value of activator or inhibitor in rule.	4 works well.

the waves do not reflect off the edges. Figure 3 shows an image of a randomly perturbed two-dimensional wave schema rule.

One might be tempted to say that the wave-based rules have complexity type IIIb, in that the individual wave patterns behave somewhat like gliders that move around. On the other hand, since the wave equation is linear, the wave crests cross each other without interacting. And the interaction of gliders is really the essence of what we think of as complexity type IIIb, for one expects a complexity type IIIb rule to appear as if it may be capable of simulating a universal computer.

In order to have wavelike rules in which the individual wave-patterns interact, we need a nonlinear wave equation along the lines mentioned in section 2. We have worked with three nonlinear two-dimensional wave schemes. Two are fairly straightforward: a quadratic and a cubic nonlinear wave. The third is more complicated, it is a cubic nonlinear wave that has a "homeostatic" tweak designed to prevent it from becoming unstable.

The first quadratic and cubic wave rules are based on a von Neumann neighborhood. The homeostatic cubic wave rule uses the von Neumann neighborhood for the "wave mode" of its updates and uses the Moore neighborhood when it enters an "averaging mode" to smooth out instabilities. Recalling that we use uE, uN, uW, and uS to stand for the intensity values of the update cell's four von Neumann neighbors, we can write the quadratic and cubic wave schemes as follows:

FIGURE 2 Zhabo Worms. This is an activator-inhibitor diffusion rule. It slowly self-organizes scrolls from a random start.

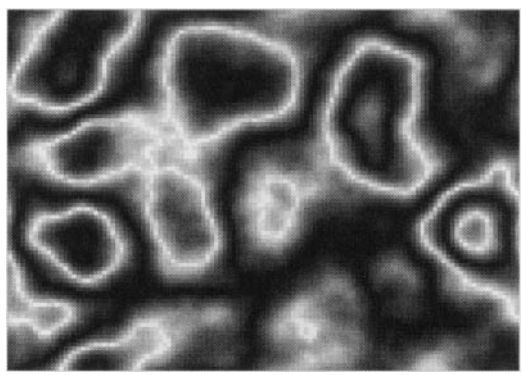

FIGURE 3 A two-dimensional wave schema being run with a *Wave* constant of 0.694 and a *uMax* of 3.0. The pattern started as a two-dimensional sine wave and was repeatedly perturbed with random conical "dings." It will continue sloshing around like this indefinitely.

(Two-Dimensional Quadratic Wave Scheme)

$$uNew = (2 \times u - uPast) + 2 \times Wave \times (uNabeAvg - u + Nonlinearity \times ((uE - u)^2 - (u - uW)^2 + (uN - u)^2 - (u - uS)^2))$$

(Two-Dimensional Cubic Wave Scheme)

$$uNew = (2 \times u - uPast) + 2 \times Wave \times (uNabeAvg - u + Nonlinearity \times ((uE - u)^3 - (u - uW)^3 + (uN - u)^3 - (u - uS)^3))$$

TABLE 6 The parameters for the three activator-inhibitor-diffusion rules of figure 2, color plate 14, and color plate 15.

	Zhabo Worms	**Turing Leopard**	**Turing Stripes**
Neighbors	2D Moore	2D Moore	2D Moore
Complexity	IIIa	IIa	IIa
sDensity	0.52	0.011	0.015
aDiffuse	0.0975	0.0399	0.049
bDiffuse	0.04375	0.99995	0.99995
aBase	0.256	0.01	0.01
bBase	0.004	0.0055	0.0055
aDecay	0.52	0.015	0.01
bDecay	0.3	0.01	0.015
aSaturation	0.0	0.04	0.2
bMin	0.52	0.001	0.001
abMax	4.0	4.0	4.0

(Clamp) Clamp u to be in the range $[-uMax, uMax]$.

The third nonlinear wave is a "homeostatic cubic wave" scheme which we will discuss below. In figure 4 we show our four kinds of waves side by side.

The nonlinear wave schemes easily go unstable, especially the cubic one. In these waves, instability will mean that the intensity values grow without bound. Thanks to the clamping step, the values then get stuck at a maximum or a minimum value. In the case of the two-dimensional quadratic wave, instability can lead to a certain kind of interesting structures. But in the cubic case, an instability is typically a checkerboard of alternating maximum and minimum-valued cells which grows to fill the simulation space.

Our "homeostatic cubic" rule has an ad hoc technique for taming the cubic instabilities. The idea is to run an unstable cubic wave, and to let the nonlinearity of the wave be determined locally. This gives an interesting effect showing Zhabotinsky patterns moving about in a wave medium.

For the homeostatic cubic wave, each cell holds an intensity u and a nonlinearity multiplier called *LocalNonlinearity*. In addition the rules use a helper variable *TooBig* and some additional parameters as shown in table 7.

The update process for this rule has two parts: the computation of the *uNew* value and the computation of the *LocalNonlinearityNew* value.

At the first stage of the update, we compute the *uNew* value in one of two possible ways, depending on the size of the *LocalNonlinearity* variable. If *LocalNonlinearity* is small, that is an indication that we are in a zone that was

TABLE 7 The variables and parameters for the homeostatic cubic wave scheme, along with the values used for the Homeostatic Cubic Zhabo in color plate 16.

Symbol	Comments	Homeostatic Cubic Zhabo
Variables:		
u	Intensity. Ranges from 0 to $uMax$.	
Localnonlinearity	Each cell has its own value for this; it starts at $MinNonlinearity$ and drifts up toward $MaxNonlinearity$.	
TooBig	Boolean helper variable to signal when the rule has become unstable at a given cell.	
Parameters:		
wave	Normally ranges from 0 to 1, although could be larger as we expect this rule to become unstable anyway.	0.5
$MaxNonlinearity$	The $LocalNonlinearity$ values in the individual cells move toward this. Can be any positive value.	100.0
$GrowFactor$	The multiplier which moves the cells' $LocalNonlinearity$ value toward $MaxNonlinearity$. Should be slightly larger than 1.	1.05
$MinNonlinearity$	This needs to be larger than 0 because we increase the $LocalNonlinearity$ by repeated multiplications by $GrowFactor$.	0.01
$WaveThreshold$	This should be bigger than $MinNonlinearity$. When the $LocalNonlinearity$ lies between $MinNonlinearity$ and $WaveThreshold$ we update u with an averaging rule, otherwise we use a cubic wave rule.	0.015
$uMax$	We clamp u to be in the symmetric range $(-uMax, uMax)$.	1.0

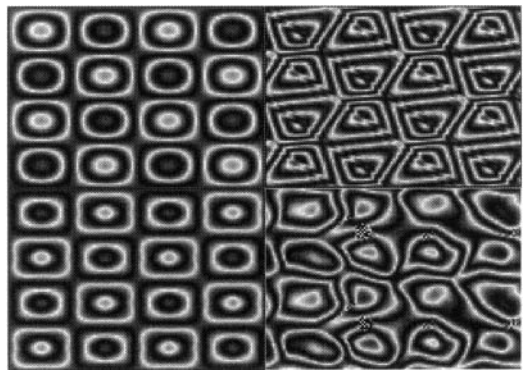

FIGURE 4 A view of four kinds of two-dimensional waves. From the left, the top row has a linear wave and a quadratic wave, and the bottom row has a cubic wave and a homeostatic cubic wave. Each rule was seeded with a four full cycles of a sine-wave pattern and was run for about 500 updates. In the linear wave this pattern simple oscillates forever, making "sushi" patterns that are displayed by showing the intensities in different shades of black and white. In the quadratic wave, the peaks become asymmetric, and in the cubic wave the peaks become more angular. The flaws on the cubic homeostasis wave are locations where the wave has become unstable and has intensity values that are being clamped to the maximum or minimum allowable value. All the rules are being run with a *Wave* constant of 0.694 and a *uMax* of 3.0. The *Nonlinearity* values of the quadratic and cubic, waves are, respectively 0.5, 3. The *Nonlinearity* in the homeostatic cubic wave varies from cell to cell, ranging from 0.001 to 1000.

recently unstable, and we update *uNew* by a simple averaging scheme. If *LocalNonlinearity* is larger, we update *uNew* by a cubic wave scheme, and we use the *LocalNonlinearity* as the cubic wave's *Nonlinearity* parameter. Finally, after updating *uNew*, we clamp *uNew* to lie in the range $(-uMax, uMax)$. If *uNew* was indeed out of range, we set my *TooBig* helper variable to *true*, otherwise we set *TooBig* to false.

At the second stage of the update, we compute the *LocalNonlinearityNew* value in one of two possible ways, depending on whether *TooBig* is *true* or *false*. In the *TooBig* case, we let *LocalNonlinearityNew* collapse to *MinNonlinearity*. This has the dual effect of damping the unstable cubic rule and of signaling the cell to use an averaging rule for its next update. When *TooBig* is false, we multiply *LocalNonlinearityNew* by *GrowFactor* and clamp *LocalNonlinearityNew* to lie in the range (*MinNonlinearity*, *MaxNonlinearity*).

The *uNew* update:
(Cubic Wave Option)

$$\text{IF} \quad (LocalNonlinearity \geq WaveThreshold) \text{ THEN } uNew$$

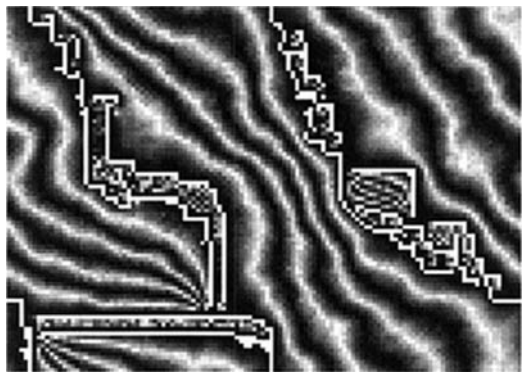

FIGURE 5 A quadratic wave scheme with *Wave* of 0.25, *Nonlinearity* of 0.15, and *uMax* of 3.0. The pattern was seeded with all u values of 1.5 with a conical bump in one location. The cone tip produced an instability which propagated along a closed "fault line." (Recall that this is a toroidal space.) The pattern is now stable and will remain like this indefinitely. Note that small structures are able to move along within the "wave guide" pieces of the fault.

$$
\begin{aligned}
&= 2 \times u - uPast + Wave \\
&\quad \times (uNabeAvg - u + LocalNonlinearity \\
&\quad \times ((uE - u)3 - (u - uW)3 + (uN - u)3 + (u - uS)3));
\end{aligned}
$$

(Averaging Option)

$$\text{IF}(LocalNonlinearity < WaveThreshold) \text{ THEN } uNew = uNabeAvg;$$

(Clamp and set *TooBig*)

Clamp *uNew* to lie in the range $(-uMax, uMax)$. If *uNew* was outside the range set *TooBig* to true, otherwise set *TooBig* to *false*.

The *LocalNonlinearityNew* update:

(Collapse option)

$$\text{IF } (TooBig) \text{ THEN } LocalNonlinearityNew = MinNonlinearity$$

(Growth option)

$$
\begin{aligned}
&\text{IF } (NOTTooBig) \text{ THEN } LocalNonlinearityNew \\
&\quad = GrowFactor \times LocalNonlinearity
\end{aligned}
$$

(Clamp)

Clamp *LocalNonlinearityNew* to be in the range *(MinNonlinearity, MaxNonlinearity)*.

This rule readily falls into a Zhabotinsky-style pattern of complexity type IIIa. The Zhabotinsky spirals are driven by the behavior of the *LocalNonlinearity* parameter, which grows to a maximum value and then drops abruptly.

6 REACTION WAVE SYSTEMS

We have also done some preliminary work in trying to put a wave term in place of the diffusion term in one or both of the two equations in our activator-inhibitor systems. We have tried various ways of doing this, but none has been an outstanding success in terms of producing really interesting behaviors. A typical scheme we have tried has the following form. In this scheme we do not use an *aSaturation* term.
(Avoid division by 0)

$$\text{IF } (-bMin < b < bMin) \text{ THEN } b = Sign(b) \times bMin \text{ ELSE } bSafe = b\,.$$

(Activator)

$$aNew = 2 \times a - aPast + Wave \times (aNabeAvg - a) + \frac{sDensity \times a \times a}{bSafe} + sDensity \times aBase - aDecay \times a\,.$$

(Inhibitor)

$$bNew = 2 \times b - bPast + Wave \times (bNabeAvg - b) + sDensity \times a \times a - bDecay \times b.$$

(Clamp) Clamp both a and b to be in the range $[-abMax, abMax]$.

This rule produces patterns resembling clouds that continually form and dissolve (see fig. 6 and table 8).

7 SUGGESTIONS FOR FURTHER WORK

Meinhardt [4] investigates other, more complicated, kinds of one-dimensional activator-inhibitor systems, and it would be interesting to cast more of these into two-dimensional form. It would be of particular interest to see Turing patterns which move about; that is, it would be nice to see crawling dots and writhing stripes. One result of this might be that our CAs could begin to model the motions of extended objects.

Another, related, goal is to find some continuous-valued CAs with more purely type IIIb behaviors. That is, one would like to see gliderlike patterns moving about and interacting.

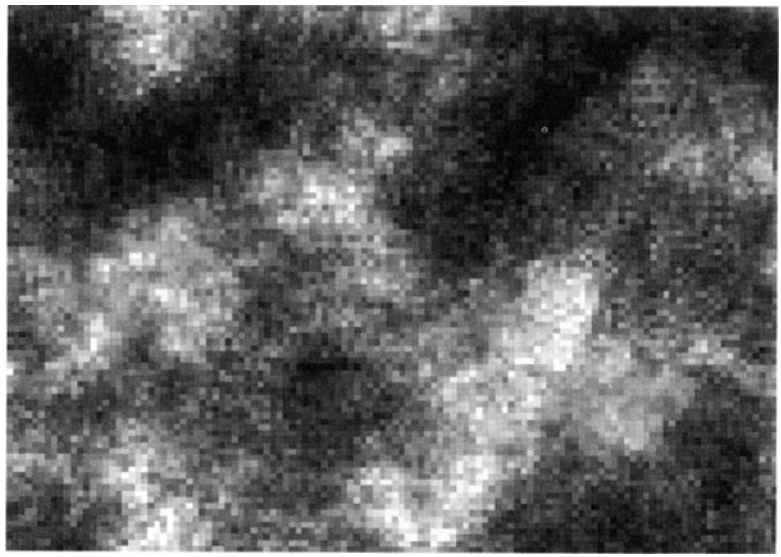

FIGURE 6 A cloudlike pattern formed by an activator inhibitor rule with wave terms in place of the customary diffusion terms. This rule converges quickly to this behavior from a random start. The rule is shown using only one band of color, that is, black in the minimum intensity and white is the maximum. See table 8 for the parameter values used.

TABLE 8 The parameter values used for the Cloud rule in figure 6.

Name	Clouds
Neighbors	2-D Von Neumann
Complexity	IV
Wave	0.25
aDensity	0.01
aBase	0.01
bBase	0.0055
aDecay	0.01
bDecay	0.015
bMin	0.001
abMax	32.0

It might also be useful to base some rules on third-order differential equations; presumably using something like a $uSecondNabeAvg - 2 \times uNabeAvg + u$ term, where the *uSecondNabeAvg* would be computed from neighbors two cells away. Perhaps some of these rules could exhibit solitons that might play the role of information-bearing gliders.

Finally, there is still the open frontier of three-dimensional CAs. Certainly it would be nice to generalize the simple activator inhibitor schemes to three-dimensions so as to produce three-dimensional Turing patterns.

REFERENCES

[1] Ball, Philip. *The Self-Made Tapestry: Pattern Formation in Nature.* Oxford, NY: Oxford University Press, 1999.

[2] Dawkins, Richard. "The Evolution of Evolvability." In *Artificial Life*, edited by C. Langton, 201. Reading, MA: Addison-Wesley, 1989.

[3] Fermi, E., J. Pasta, and S. Ulam. "Studies of Nonlinear Problems." Los Alamos Report LA-1940, Los Alamos, NM, 1955.

[4] Meinhardt, Hans. *The Algorithmic Beauty of Shells.* New York: Springer-Verlag, 1995.

[5] Ostrov, Daniel, and Rudy Rucker. "Continuous-Valued Cellular Automata for Nonlinear Wave Equations." *Complex Systems* **10** (1996): 91–119.

[6] Rucker, Rudy. "CAPOW! Version 6.3," software available for free download at ⟨http://www.mathcs.sjsu.edu/capow⟩, 1999. This product was funded by a contract with EPRI (The Electric Power Resesarch Institute) of Palo Alto, CA.

[7] Toffoli, Tommaso, and Norman Margolus. *Cellular Automata Machines.* Cambridge, MA: MIT Press, 1987.

[8] Turing, Alan. "The Chemical Basis of Morphogenesis." *Phil. Trans. Roy. Soc. Lond. B* **237** (1952): 37.

[9] Ulam. S. *Sets, Numbers and Universes*, 491–501. Cambridge, MA: MIT Press, 1974.

[10] Weissert, Thomas. *The Genesis of Simulation in Dynamics: Pursuing the Fermi-Pasta-Ulam Problem.* New York: Springer-Verlag 1997.

This page intentionally left blank

Phase Transition via Cellular Automata

Gadi Moran

The dynamics of unit-charged graphs under iterated local majority rule observed in Moran [2] strongly suggested to me a phase-transition phenomenon. In a correspondence with D. Ruelle on this matter in late 1993, he expressed his feelings that the connection was too vague and that temperature was absent in it. This note is a reproduction of my 1993 response, where I try to force my suggestive feelings into a bit more formal frame.

A recent work of Yuval Ginosar and Ron Holzman [1], which extends Moran [2], allows us to replace the definition of a solid, given in section 4, by a sharper one, namely that of a "puppet" in their terminology. This means that in section 4 we may define a $G \in Y$ to be a solid if *every* initial charge upon it decays under these dynamics—possibly in infinite time—into a time-periodic charging of a time period not longer than two.

New Constructions in Cellular Automata,
edited by David Griffeath and Cristopher Moore, Oxford University Press.

1 INTRODUCTION

This note suggests an approach to the phenomenon of phase transition based on the behaviour of some cellular automata on infinitely countable nets, as noted recently in Moran [2]. Specifically, we use a majority automaton operating simultaneously on a countably infinite graph as a test device determining its "phase." Results in Moran [2] suggest some sharp partition of a configuration space made up of the totality of such graphs into "solids," where the only periods allowed for the automaton are 1 or 2, versus the others. Results in Moran [2] allow also the introduction of a "temperature" functional—a numerical parameter defined for each configuration, with the property that a configuration is "solid" whenever its "temperature" is negative.

We first describe a possible physical interpretation of such a model, taking the nodes of a graph to be "particles" (stars, electrons, ions, atoms, molecules, radicals—as the case may be) in some Riemannian manifold. Our interpretation is obviously open to a wide diversity of modifications. It is hoped that in spite of its admittedly speculative nature, it may invoke a novel approach to the theoretical treatment of phase transition.

2 THE MODEL

Let V denote a countable infinite collection of "particles" situated in some Riemannian manifold. Assume for simplicity that the particles are balls, whose radii range between ρ_1 and ρ_2, $0 < \rho_1 < \rho_2$. The particles may be static, may oscillate around some fixed position or may move freely around.

Fix a positive number τ. At a given time t_0 we consider our system of particles V and follow the individual motion of each $v \in V$. Let $\rho \leq r$ be a fixed positive number, and let B_v denote a ball of radius r around the center of the ball particle v. Over the period of time $t_0 \leq t \leq t_0 + \tau$, our particles may move around, with the B_v's moving with them. We associate a graph $G_{t_0} = (V, N_{t_0})$ with our system (at time t_0) by making v and v' joined by an edge if and only if B_v and $B_{v'}$ intersect while moving during the period $[t_0, t_0 + \tau]$. Formally we put:

1. $N(v, v') = N(v', v) = 1$ if and only if: $v \neq v'$ and for some t, $t_0 \leq t \leq t_0 + \tau$, $B_v^t \cap B_{v'}^t \neq \phi$ (where obviously, B_v^t is the space occupied by B_v at time t);
2. $N(v, v') = N(v', v) = 0$, otherwise.

REMARKS:

1. Notice that G_{t_0} is independent of t_0 for a system made of static particles, or when the intersections $B_v \cap B_{v'}$ are "large" relative to the motions of the

particles around fixed positions. It is time dependent for a general system of freely moving particles.

2. Our condition that all particles are balls at least as big as a ball of radius $\rho_1 > 0$ combined with a universal bound S_M on the speed of particles (e.g., the speed of light) imply that the graph G_{t_0} is locally finite, and often that there is a uniform finite bound on the degrees in G_{t_0} (namely, the maximal number of disjoint balls of radius ρ_1 that can enter in the region covered by a ball of radius $2r$ moving at speed $2S_M$ over a period τ).

3. When the particles are static and far apart (more than $2r$ from each other) G_{t_0} is discrete graph (this should correspond to O°K).

For a system of particles that are moving or close together, G_{t_0} will consist of connected components with possibly infinitely many particles in some. By choosing r and t large enough, we can justify the assumption made below (needed for applying Moran [2]) that G_{t_0} is a connected graph.

3 THE CONFIGURATION SPACE (PHASE SPACE)

We take for a configuration space Y the totality of all *connected* graphs on a fixed set of vertices V possessing *a uniform finite bound* on the degrees (i.e., the number of neighbours a vertex $v \in V$ may have). Some justification for the two restrictions—connectedness and uniform-degree-boundedness—is suggested in remarks 2 and 3 of section 2.

4 SOLIDS VERSUS FLUIDS

We test a graph $G \in Y$ for solidity by its response to "shocks." Solids are expected to absorb shocks. Fluids let them through. A "shock" is any initial $\{+,-\}$-"charge" of the particles v in V. By saying that "a shock is applied to G" we mean that a *unit* charge was placed at each vertex v of G. Applying a shock to G we obtain a *charged graph* G°, which is subjected to iterations of a transformation we shall call *local majorization*, applicable to an arbitrary charged graph of finite degrees, thus yielding a sequence of charged graphs $G^0, G^1, G^2, \ldots$. The transition from G^k to G^{k+1} is made by recharging $v \in V$ by the charge of the majority of its neighbours in G^k, leaving v's charge unchanged in case of a tie.

A shock may or may not lead to an ultimately periodic sequence $(G^k)_{k=0,1,2,\ldots}$. We are interested in the possible periods that may occur in the first case. If any periodic sequence that occurs as a result of a shock has a period of length ≤ 2 , we say that G has the *Period Two Property (p2p)*. We choose the *p2p* as our test for solidity.

Definition $G \in Y$ is *solid* if and only if G has the $p2p$. Otherwise we call G *fluid*.

5 THEOREM

We restate a result in Moran [2] pertaining to our subject in the present glossary:

Theorem Let $G \in Y$ and let $D(G)$ denote the maximum degree of some $v \in V$. Let d denote the maximal even integer strictly smaller than $D(G)$. Also, let $g(G)$ denote the growth of G. *Then*:

$$g(G) < 1 + \frac{2}{d} \Rightarrow G \text{ is solid} \tag{1}$$

(Moran [2], Theorem 2).

Two remarks are in order:

1. $d = 0$ is possible. Then $2/d$ is to be interpreted as ∞. (This is a trivial case—the graph G is a chain. It is always a solid.)
2. The growth $g(G)$ mentioned in the Theorem is a real number satisfying: $1 \leq g(G) \leq D(G) - 1$. It is defined as follows. Choose $v_0 \in V$, and for $n = 1, 2, 3, 2$ let b_n denote the number of vertices $v \in V$ whose G-distance from v_0 is at most n (i.e., who can be joined to v_0 by a path in G of length $\leq n$). *Then*: $g(G) := \limsup b_n^{1/n}$.

Example 2 in Moran [2] presents a fluid $G_d \in Y$ with $g(G_d) = 1 + (2/d)$, $D(G_d) = d + 1$, for each even number $d > 0$.

6 TEMPERATURE

Let $G \in Y$. Define

$$\theta(G) := \ln\left(\frac{g(G)}{1 + \frac{2}{d}}\right)$$

$$T(G) := e^{\theta(G)} = \frac{g(G)}{1 + \frac{2}{d}}$$

where $g(G)$, d are given in section 5. (Again, $d = 0$ requires special care. We put then $\theta = -\infty$, $T = 0$). By section 5 we have (Moran [2] Theorem 2, Example 2):

1. $q(G) < 0 \Rightarrow G$ is solid.
2. For every positive even integer d there is a $G_d \in Y$ such that G_d is fluid, $\theta(G_d) = 0$ and $D(G_d) = d + 1$.

We refer to $\theta(G)$ as the celsius temperature of G and to $T(G)$ as the kelvin temperature of G.

REFERENCES

[1] Ginosar, Y., and R. Holzman. "The Majority Action on Infinite Graphs: Strings and Puppets." *Discrete Math.* (1999): To appear.

[2] Moran, G. "On the Period-Two–Property of the Majority Operator in Infinite Graphs." *Trans. A.M.S.* **347** (1995): 1649–1667.

This page intentionally left blank

Index

Z