Universidade Estadual de Maringá
Centro de Ciências Exatas
Departamento de Física

Trabalho de Conclusão de Curso

Estudando autômatos celulares com a linguagem de programação *Go*

Acadêmica: Beatriz de Castro Bittencourt

Orientador: Prof. Dr. Breno Ferraz de Oliveira

Maringá, 24 de janeiro de 2025

Universidade Estadual de Maringá
Centro de Ciências Exatas
Departamento de Física

Trabalho de Conclusão de Curso

Estudando autômatos celulares com a linguagem de programação *Go*

Trabalho de Conclusão de Curso apresentado ao Departamento de Física da Universidade Estadual de Maringá, sob orientação do Prof. Dr. Breno Ferraz de Oliveira, como parte dos requisitos para obtenção do título de Bacharel em Física.

Acadêmica: Beatriz de Castro Bittencourt

Orientador: Prof. Dr. Breno Ferraz de Oliveira

Maringá, 24 de janeiro de 2025

Beatriz de Castro Bittencourt

Estudando autômatos celulares com a linguagem de programação *Go*

Trabalho de Conclusão de Curso apresentado ao Departamento de Física da Universidade Estadual de Maringá, sob orientação do Prof. Dr. Breno Ferraz de Oliveira, como parte dos requisitos para obtenção do título de Bacharel em Física.

Aprovado em __/__/____

Banca Examinadora

Prof. Dr. Breno Ferraz de Oliveira

Prof. Dr. Guilherme Maia Santos

Prof. Dr. Miguel Jorge Bernabé Ferreira

Agradecimentos

Agradeço primeiramente a Deus por ter feito possível esta jornada, me fortalecendo e trazendo perseverança a cada passo. Agradeço à Universidade Estadual de Maringá por proporcionar minha caminhada por este curso, rica em experiências e aprendizados, culminando neste trabalho final. Agradeço ao CNPq e à Fundação Araucária pela oportunidade de iniciação científica, fornecendo apoio por meio do PIBIC e indiretamente a base para este trabalho, que não seria possível sem o conhecimento adquirido por meio do projeto. Agradeço ao professor Dr. Breno Ferraz de Oliveira por todas as orientações recebidas ao longo de minha graduação, desde o citado PIBIC, a experiência no projeto de extensão TRUEM até este trabalho. Estendo meus agradecimentos à banca examinadora.

Agradeço à minha família e a meus amigos pelo apoio recebido durante esta caminhada. Estendo meus agradecimentos a meus colegas de curso, por todas as conversas, sucessos e desafios compartilhados ao longo destes cinco anos. Finalmente, mas não menos importante, agradeço a Luiz Felipe Locatelli Giroldo pelo apoio e presença constantes, como colega de curso e parceiro.

Conteúdo

Resumo

O conceito de autômatos celulares, introduzido em 1940, ganhou destaque apenas em 1970 com a divulgação do Jogo da Vida, desenvolvido por John Horton Conway e popularizado pelo escritor Martin Gardner no contexto de matemática recreativa. Nesse modelo, células em uma rede alternam entre os estados vivo e morto com base em interações locais, gerando uma ampla gama de padrões e comportamentos que despertaram crescente interesse global. A versatilidade apresentada por este autômato levou ao desenvolvimento de modelos mais complexos, incluindo autômatos que transitaram do domínio discreto para o contínuo em busca de novas "formas de vida" e dinâmicas. Dentre eles, destacam-se SmoothLife e Lenia, que buscaram implementar generalizações do Jogo da Vida, resultando na descoberta de espécimes variados e, no caso de Lenia, na classificação de mais de 400 espécies em táxons. Tais descobertas, bem como a similaridade de espécimes de Lenia com organismos vivos, ampliaram a discussão quanto aos atributos que caracterizam a vida e a busca por vida artificial. Neste trabalho, foram realizadas simulações para os três autômatos mencionados, explorando diferentes condições iniciais, dimensões e parâmetros a fim de reproduzir comportamentos documentados e analisar a influência dos métodos na evolução temporal dos sistemas. Os resultados obtidos confirmaram o objetivo proposto e destacaram o potencial de autômatos celulares como ferramentas para a modelagem de sistemas dinâmicos e para futuros estudos na área de vida artificial.

Palavras-Chave: autômatos celulares, vida artificial, simulações estocásticas

Abstract

The concept of cellular automata, introduced in 1940, gained prominence only in 1970 with the publicity of the Game of Life, developed by John Horton Conway and popularized by writer Martin Gardner in the context of recreational mathematics. In this model, cells in a grid alternate between alive and dead states based on local interactions, generating a wide range of patterns and behaviors that sparked growing global interest. The versatility demonstrated by this automaton led to the development of more complex models, including automata that transitioned from the discrete to the continuous domain in the search for new "forms of life" and dynamics. Among these, SmoothLife and Lenia stand out, as they implemented generalizations of the Game of Life, resulting in the discovery of various specimens and, in the case of Lenia, the classification of more than 400 species into taxa. These discoveries, as well as the similarity of Lenia's specimens to living organisms, broadened the discussion on the attributes that characterize life and the quest for artificial life. In this study, simulations were performed for the three aforementioned automata, exploring different initial conditions, dimensions, and parameters to reproduce documented behaviors and analyze the influence of methods on the temporal evolution of the systems. The results confirmed the proposed objective and highlighted the potential of cellular automata as tools for modeling dynamic systems and for future studies in the field of artificial life.

Keywords: cellular automata, artificial life, stochastic simulations

Capítulo 1

Introdução

O conceito de autômato celular foi introduzido por John von Neumann e Stanislaw Ulam em 1940 (SCHIFF, 2011; NEUMANN, 2017) e descreve um modelo de computação, ou seja, como saídas de funções matemáticas são calculadas a partir de uma entrada. Autômatos são, por definição, dispositivos de computação autopropulsionados e abstratos que seguem uma sequência definida de operações. Assim, as saídas são calculadas de forma automática pelo dispositivo, utilizando informações de entradas e regras pré-definidas.

Autômatos celulares, por sua vez, são autômatos restritos a uma rede de células. São formalmente definidos como uma rede regular de células, que têm um número finito de estados possíveis. A mudança ou manutenção destes se deve às interações de cada célula com sua vizinhança, cujo modelo é escolhido arbitrariamente. De acordo com uma distribuição inicial, pseudoaleatória ou intencional, e regras fixas, descritas por equações matemáticas (TOFFOLI; MARGOLUS, 1987), o estado de cada célula é atualizado simultaneamente (SCHIFF, 2011). Com a varredura de toda a rede, tem-se um passo temporal, que define uma geração.

Embora autômatos celulares tenham sido amplamente estudados desde a década de 1940, foi com a divulgação do Jogo da Vida de John Horton Conway em 1970 (GARDNER, 1970) que o interesse pelo tema se expandiu para além do ambiente acadêmico, alcançando fama global em décadas recentes. O Jogo da Vida de Conway é um autômato celular que utiliza uma configuração inicial, definindo arbitrariamente células vivas e mortas, e é utilizado um conjunto de regras específico para determinar a evolução de cada célula. Em suma, uma célula morta se torna viva se possuir exatamente três vizinhos vivos, e uma célula viva permanece viva se possuir dois ou três vizinhos vivos. Em quaisquer outros casos, a célula permanece ou se torna morta. É feita a varredura de toda a rede, linha por linha, e assim são atualizados os estados de todas as células de forma simultânea. Após a atualização completa da rede, passa-se uma geração, e o processo se repete.

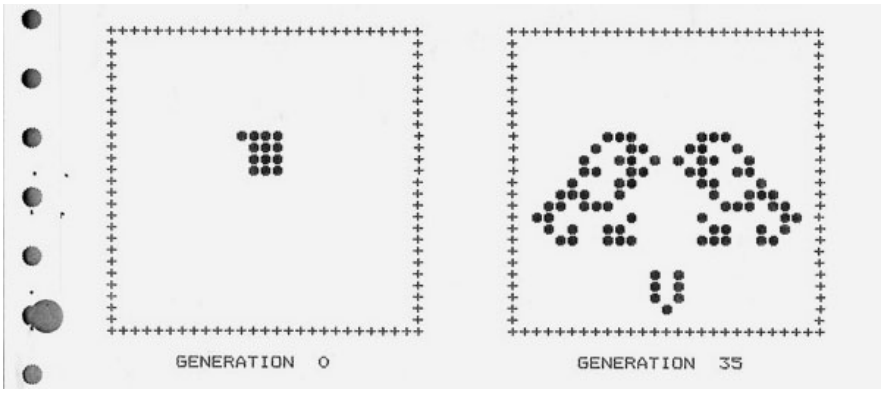


Figura 1.1: Evolução de uma rede para o Jogo da Vida de Conway (ZELDES, 2007).

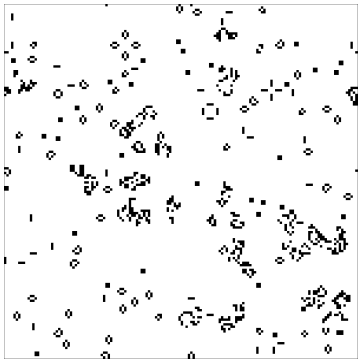
Desta forma, no Jogo da Vida de Conway, tem-se um dos dois estados possíveis a depender dos estados das células vizinhas. Assim, um dos principais atributos que impulsionaram a popularidade deste autômato foi o surgimento de variados padrões na rede conforme a progressão temporal (GARDNER, 1970). Enquanto centenas de padrões celulares foram encontrados e catalogados e inúmeros ainda estão para ser descobertos, passou-se a buscar evoluções diferentes. Ou seja, embora o Jogo da Vida de Conway siga um conjunto definido de normas, inúmeras variantes foram criadas ao alterá-lo, e estas exibem dinâmicas e resultados únicos. Com estas novas versões do Jogo da Vida, as possíveis saídas apresentam situações novas e restritas a certas regras, como formações específicas de células. Assim, graças a seu potencial de versatilidade, o Jogo da Vida se mantém como um tópico de interesse no meio científico.

Outros parâmetros que podem ser alterados incluem a própria rede, o número de estados possíveis e o modelo de vizinhança. A partir de mudanças em parte desses atributos, juntamente com novos conjuntos de regras, surgiram novos autômatos celulares como Larger Than Life (EVANS, 1996) e RealLife (PIVATO, 2007). A partir das discussões trazidas por esses modelos, principalmente quanto aos possíveis resultados obtidos por meio da transição do meio discreto à continuidade, foram criados dois autômatos celulares independentes: SmoothLife (RAFLER, 2011) e Lenia (CHAN, 2019).

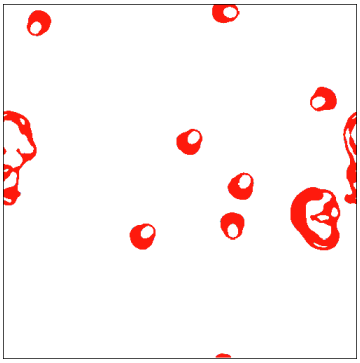
SmoothLife representa uma generalização do Jogo da Vida em um domínio contínuo, considerando a continuidade entre os estados vivo e morto, isto é, estados gradativos. Além disso, a vizinhança é descrita por um anel que a divide em duas regiões ao redor da célula. Para computar as interações entre a célula e a vizinhança, utiliza-se a teoria de campo médio, em que a vizinhança é tratada como um campo com o qual a célula interage (KADANOFF, 2009). É possível tratar o tempo como contínuo, embora o formato padrão deste autômato utilize gerações discretas. Essas características permitem o surgimento de padrões variados na rede, exclusivos ao SmoothLife, e que se movem e interagem das mais diferentes maneiras.

Lenia, por sua vez, utiliza uma vizinhança radial e com pesos estatísticos, a fim de fazer com que uma região possua maior influência no estado da célula. Ademais, utiliza uma distribuição de potencial e uma função de crescimento para determinar a evolução temporal da rede, atualizando-a por meio de iterações. O tempo pode ser discreto ou fracionado, e para o último os padrões na rede se movem e interagem de forma fluida, proporcionando mais espécimes e evidenciando suas semelhanças com organismos vivos. É teorizada uma forma inteiramente contínua deste autômato, em que a vizinhança e o tempo também se tornam contínuos por meio do uso de infinitesimais.

(a) Jogo da Vida



(b) SmoothLife



(c) Lenia

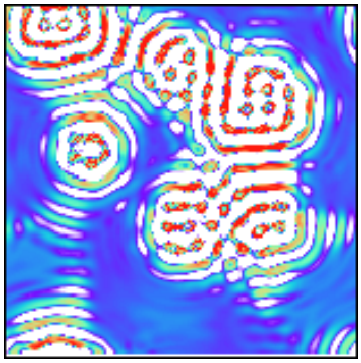


Figura 1.2: Exemplos de evolução para redes dos três autômatos celulares. Autoria própria.

Primeiramente, será discutido com profundidade o Jogo da Vida de Conway, expondo sua história, sua estrutura básica, possíveis padrões encontrados e algumas variações populares, incluindo novos autômatos. Em seguida, será abordado o SmoothLife, seu surgimento, seu fun-

cionamento, o que proporciona e como difere do primeiro. Ademais, será apresentado o Lenia, sua relevância como objeto de estudo, sua distinta estrutura e alguns espécimes catalogados. Então, a partir de uma Metodologia base para cada autômato, serão escritas simulações utilizando a linguagem de programação Go. Então, analisaremos os Resultados obtidos para cada autômato celular. Por fim, Considerações Finais serão feitas.

Capítulo 2

O Jogo da Vida de Conway

2.1 História

O Jogo da Vida é um autômato celular criado por John Horton Conway. Sua primeira divulgação foi por meio da coluna mensal *Mathematical Games* de Martin Gardner, na revista *Scientific American* em outubro de 1970. (GARDNER, 1970)

John Horton Conway (1937-2020) foi um matemático britânico. Nascido e criado em Liverpool, estudou na faculdade Gonville and Caius da Universidade de Cambridge, posteriormente trabalhando como professor na mesma universidade a partir de 1964. Em 1986, decidiu deixar Cambridge para um cargo na Universidade de Princeton, nos Estados Unidos. Conway residiu nos Estados Unidos até sua morte, em 11 de abril de 2020, em decorrência da COVID-19.

Conway era conhecido por sua pesquisa em matemática pura, especialmente nos campos de teoria dos números, grupos finitos, teoria dos nós, teoria de jogos combinatórios e teoria de codificação. Além disso, também se interessava por matemática recreativa, colecionando algumas contribuições nesta área. No final da década de 1950, começou a se corresponder com Martin Gardner, escritor de ciência popular e recém-criado colunista da revista *Scientific American*. A coluna *Mathematical Games* era destinada à discussão de trabalhos em matemática recreacional, e foi mantida por 25 anos até a aposentadoria de Gardner, em 1979.

Neste período, Gardner escreveu sobre o trabalho de Conway em matemática recreativa em inúmeras ocasiões. Como relatado por ele, embora o matemático fosse extremamente prolífico nesta área, raramente publicava suas descobertas - desta forma, a coluna se tornou seu principal meio de divulgação. Assim, Gardner foi o responsável por apresentar o Jogo da Vida ao mundo, bem como torná-lo um autômato de grande popularidade e expandir o tema de autômatos celulares para além do meio acadêmico.

Em seu texto, Gardner o descreve detalhadamente, apresentando as regras e conjecturas elaboradas por Conway e instruindo o leitor quanto à aplicação física do jogo, mesmo sem a utilização de máquinas. Além disso, expôs alguns resultados comuns e listou desafios propostos pelo próprio Conway.

Embora aqueles com acesso a máquinas e *displays* visuais encontrassem mais conveniência para experimentá-lo, devido à possibilidade de salvar passos temporais e à automatização do processo, o autômato poderia ser aplicado utilizando virtualmente quaisquer objetos à disposição, como moedas, papel e lápis ou damas. Graças a este fato, à sua imprevisibilidade e aos desafios propostos por Conway, a coluna da edição de outubro de 1970 se tornou especialmente popular, recebendo incontáveis reações dos leitores. Eventualmente, tornou-se a mais lida em todo o seu período de publicação.

Além do interesse obtido entre leitores casuais, Gardner recebeu inúmeras respostas aos desafios propostos por Conway, muitos dos quais o próprio desconhecia a solução e oferecia prêmios em dinheiro. Entre os leitores que resolveram desafios e descobriram novos padrões,

destacam-se nomes como Bill Gosper, Richard K. Guy e Robert T. Wainwright por suas contribuições.

O Jogo da Vida obteve tamanha popularidade através da divulgação de Gardner que este considerou dedicar-lhe uma segunda coluna mensal. Porém, esta ideia foi descartada quando Wainwright propôs a criação de uma *newsletter* trimestral, chamada *LIFELINE*, dedicada a expor os desenvolvimentos de Conway, Gardner, Wainwright e outras figuras envolvidas, além de receber contribuições dos leitores. Esta *newsletter* foi mantida por Wainwright, como redator e editor, pelo curso de quase três anos, entre março de 1971 e setembro de 1973 (ADAMATZKY, 2010).

Com o passar dos anos, Conway e outros matemáticos escreveram mais obras abordando o Jogo da Vida. Como exemplo, tem-se o livro *Winning Ways for Your Mathematical Plays* (BERLEKAMP; CONWAY; GUY, 2018), escrito por Conway, Guy e Elwyn Berlekamp. De acordo com a popularidade global deste autômato celular e o interesse constante neste por pesquisadores de diversas áreas, ainda são feitas descobertas e criados novos modelos derivados ou inspirados por ele.

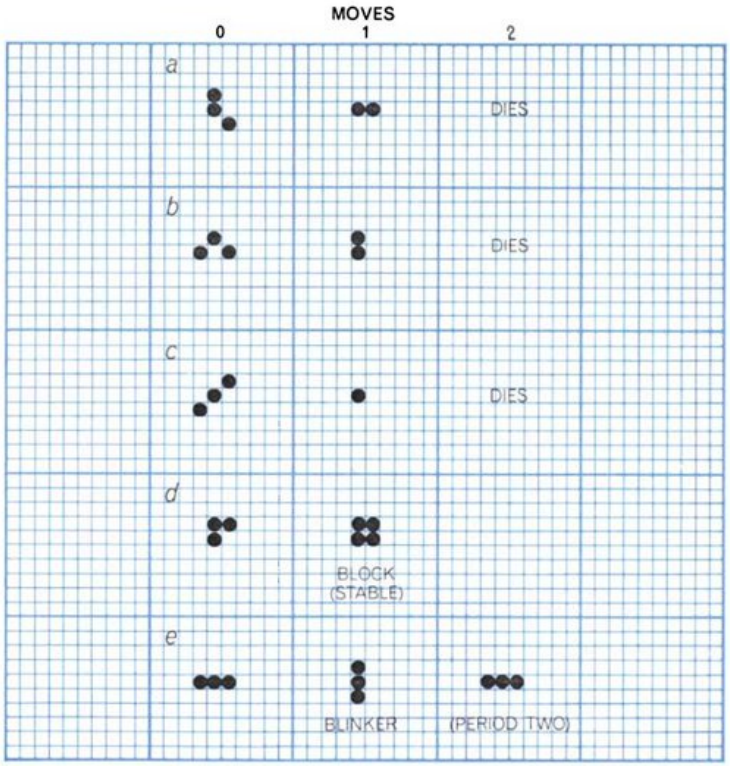
MATHEMATICAL GAMES

The fantastic combinations of John Conway's new solitaire game "life"

by Martin Gardner

Most of the work of John Horton Conway, a mathematician at Gonville and Caius College of the University of Cambridge, has been in pure mathematics. For instance, in 1967 he discovered a new group—some call it "Conway's constellation"—that includes all but two of the then known sporadic groups. (They are called "sporadic" because they fail to fit any classification scheme.) It is a breakthrough that has had exciting repercussions in both group theory and number theory. It ties in closely with an earlier discovery by John Leech of an extremely dense packing of unit spheres in a space of 24 dimensions where each sphere touches 196,560 others. As Conway has remarked, "There is a lot of room up there."

In addition to such serious work Conway also enjoys recreational mathematics. Although he is highly productive in this field, he seldom publishes his discoveries. One exception was his paper on "Mrs. Perkins' Quilt," a dissection problem discussed in "Mathematical Games" for September, 1966. My topic for July, 1967, was sprouts, a topological pencil-and-paper game invented by Conway and M. S. Paterson. Conway has been mentioned here several other times.



The fate of five triplets in "life"

This month we consider Conway's latest brainchild, a fantastic solitaire pastime he calls "life." Because of its analogies with the rise, fall and alterations of a society of living organisms, it belongs to a growing class of what are called "simulation games"—games that resemble real-life processes. To play life you must have a fairly large checkerboard and a plentiful supply of flat counters of two colors. (Small checkers or poker chips do nicely.) An Oriental "go" board can be used if you can find flat counters that are small enough to fit within its cells. (Go stones are unusable because they are not flat.) It is possible to work with pencil and graph paper but it is much easier, particularly for beginners, to use counters and a board.

The basic idea is to start with a simple configuration of counters (organisms), one to a cell, then observe how it changes as you apply Conway's "genetic laws" for births, deaths and survivals. Conway chose his rules carefully, after a long period of experimentation, to meet three desiderata:

1. There should be no initial pattern for which there is a simple proof that the population can grow without limit.
2. There should be initial patterns that *apparently* do grow without limit.
3. There should be simple initial patterns that grow and change for a considerable period of time before coming to an end in three possible ways: fading away completely (from overcrowding or from becoming too sparse), settling into a stable configuration that remains unchanged thereafter, or entering an oscillating phase in which they repeat an endless cycle of two or more periods.

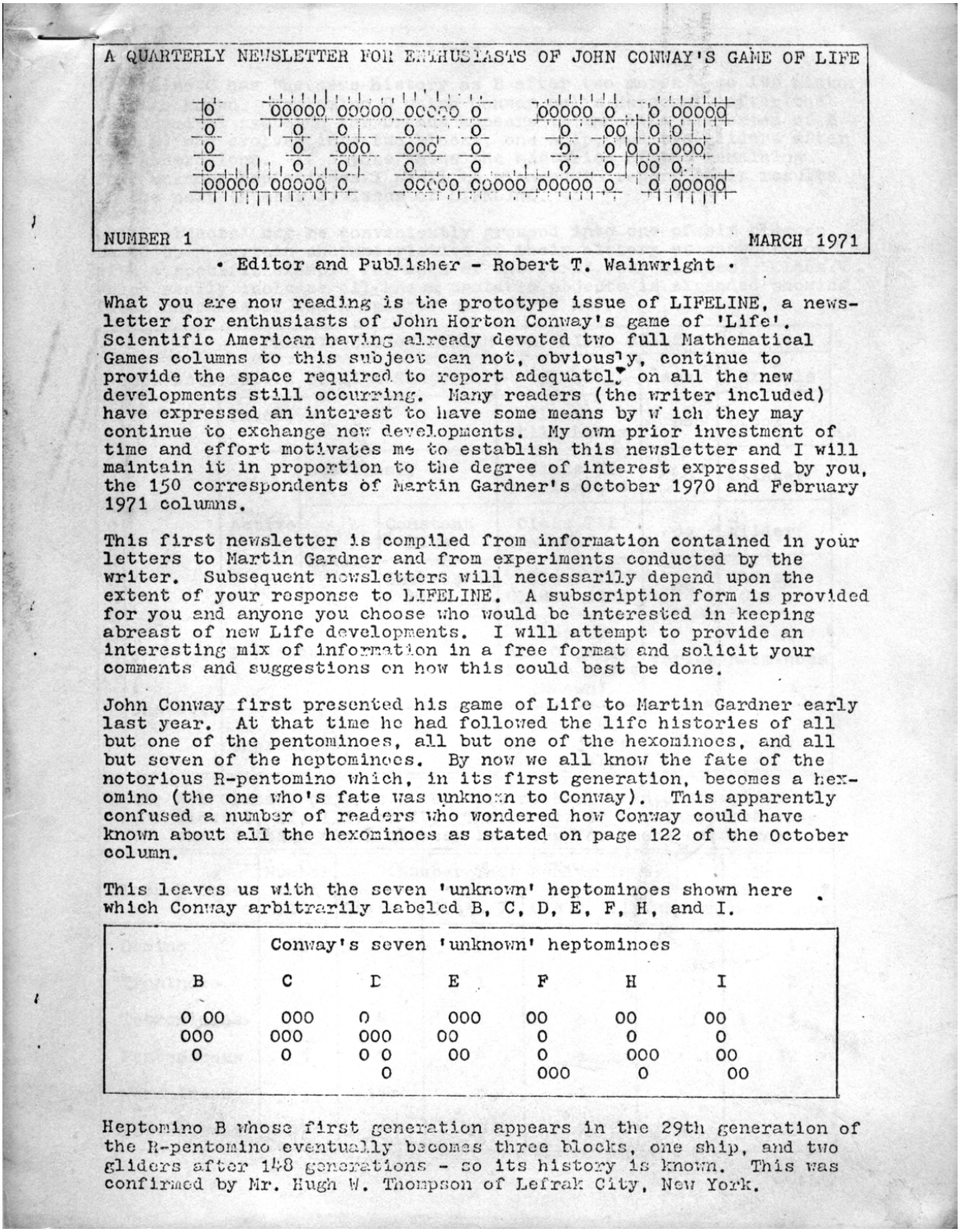
In brief, the rules should be such as to make the behavior of the population unpredictable.

Conway's genetic laws are delightfully simple. First note that each cell of the checkerboard (assumed to be an infinite plane) has eight neighboring cells, four adjacent orthogonally, four adjacent diagonally. The rules are:

1. Survivals. Every counter with two or three neighboring counters survives for the next generation.
2. Deaths. Each counter with four or more neighbors dies (is removed) from overpopulation. Every counter with one neighbor or none dies from isolation.
3. Births. Each empty cell adjacent to exactly three neighbors—no more, no fewer—is a birth cell. A counter is placed on it at the next move.

It is important to understand that all births and deaths occur *simultaneously*. Together they constitute a single genera-

Figura 2.1: Primeira página da coluna *Mathematical Games* de outubro de 1970 (GARDNER, 1970).



A QUARTERLY NEWSLETTER FOR ENTHUSIASTS OF JOHN CONWAY'S GAME OF LIFE

NUMBER 1 MARCH 1971

• Editor and Publisher - Robert T. Wainwright •

What you are now reading is the prototype issue of LIFELINE, a newsletter for enthusiasts of John Horton Conway's game of 'Life'. Scientific American having already devoted two full Mathematical Games columns to this subject can not, obviously, continue to provide the space required to report adequately on all the new developments still occurring. Many readers (the writer included) have expressed an interest to have some means by which they may continue to exchange new developments. My own prior investment of time and effort motivates me to establish this newsletter and I will maintain it in proportion to the degree of interest expressed by you, the 150 correspondents of Martin Gardner's October 1970 and February 1971 columns.

This first newsletter is compiled from information contained in your letters to Martin Gardner and from experiments conducted by the writer. Subsequent newsletters will necessarily depend upon the extent of your response to LIFELINE. A subscription form is provided for you and anyone you choose who would be interested in keeping abreast of new Life developments. I will attempt to provide an interesting mix of information in a free format and solicit your comments and suggestions on how this could best be done.

John Conway first presented his game of Life to Martin Gardner early last year. At that time he had followed the life histories of all but one of the pentominoes, all but one of the hexominoes, and all but seven of the heptominoes. By now we all know the fate of the notorious R-pentomino which, in its first generation, becomes a hexomino (the one who's fate was unknown to Conway). This apparently confused a number of readers who wondered how Conway could have known about all the hexominoes as stated on page 122 of the October column.

This leaves us with the seven 'unknown' heptominoes shown here which Conway arbitrarily labeled B, C, D, E, F, H, and I.

Conway's seven 'unknown' heptominoes

B C D E F H I

Heptomino B whose first generation appears in the 29th generation of the R-pentomino eventually becomes three blocks, one ship, and two gliders after 148 generations - so its history is known. This was confirmed by Mr. Hugh W. Thompson of Lefrak City, New York.

Figura 2.2: Primeira página do volume 1 da *newsletter LIFELINE*, de março de 1971 (LIFEWIKI, 2009).

2.2 Estrutura

Em sua essência, o Jogo da Vida é um autômato celular de duas dimensões. A estrutura básica deste é de uma rede $N_x \times N_y$ com células que assumem um de dois estados, morto ou vivo (0 ou 1). Dada uma configuração inicial, o novo estado, ou a manutenção do estado anterior, é dado pela interação da célula escolhida com sua vizinhança. A vizinhança é determinada pela forma da vizinhança de Moore, composta pelos oito indivíduos em contato direto com a célula em questão (WEISSTEIN, 2003). Considerando as células nos limites da rede, são implementadas condições de contorno. Assim, os resultados dependem de um conjunto de regras específicas. Após a varredura de todas as células da rede, que devem se atualizar simultaneamente, ocorre um passo temporal, chamado no Jogo da Vida de geração.

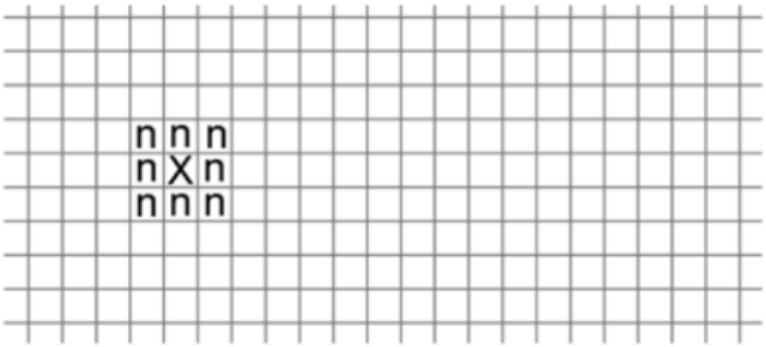


Figura 2.3: Esquema ilustrativo de uma célula, denotada por X, e seus vizinhos, denotados por n (ADAMATZKY, 2010).

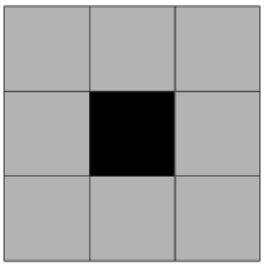


Figura 2.4: Representação da vizinhança de Moore, em cinza, em relação à célula central, em preto (SKONECZNY, 2017).

As normas elaboradas por Conway quanto a configurações iniciais têm sua forma para que seja assegurado o fator de imprevisibilidade para qualquer configuração inicial utilizada (GARDNER, 1970):

1. Não deve haver configuração inicial para a qual há simples provas de que a população crescerá sem limites;

2. Deve haver configurações iniciais pelas quais apenas aparenta-se que a população crescerá sem limites;

3. Deve haver simples configurações iniciais que crescem e mudam por um tempo considerável.

Dados os requisitos para esboçar a configuração inicial, esta pode ser criada diretamente pelo pesquisador ou gerada automaticamente por um computador.

Feita a configuração inicial, tem-se as regras para as mudanças entre estados. O conjunto de regras utilizado pode variar a depender do intuito do pesquisador. A notação comumente utilizada para expressar o conjunto de regras é B$x_1, x_2.../$S$y_1, y_2...$, em que x e y são números naturais; B representa "nascimento" (do inglês, *birth*) e S representa "sobrevivência" (do inglês, *survival*). Assim, os números junto de cada letra retratam as quantidades de vizinhos necessárias para ocorrer nascimento ou sobrevivência na rede. A seguir, tem-se o conjunto de regras proposto por Conway e considerado o padrão (GARDNER, 1970):

1. Toda célula viva com dois ou três vizinhos vivos sobreviverá;
2. Toda célula viva com menos de dois ou mais de três vizinhos vivos morrerá, seja por isolamento ou superpopulação;
3. Toda célula morta com exatamente três vizinhos vivos se tornará viva, ocorrendo um nascimento.

O conjunto acima é, então, representado como B3/S23, e configura o Jogo da Vida de Conway. É importante ressaltar que a adição "de Conway" ao Jogo da Vida refere-se especificamente às regras B3/S23. Note que as regras podem ser escritas como uma matriz de transição, descrevendo as probabilidades das mudanças entre estados. Como foi mencionado anteriormente, essas mudanças devem ser simultâneas, completando uma geração ao varrer toda a rede, enquanto o número de gerações é arbitrário. Devido às regras de Conway, observa-se que a rede evolui de forma imprevisível, porém apenas um de três fins pode ser encontrado (GARDNER, 1970):

- Todas as células vivas desaparecem após um número de gerações;
- A rede atinge uma forma completamente estável independente do número de gerações, composta de "naturezas-mortas";
- A rede atinge formas que oscilam periodicamente, infinitamente, sendo este período de pelo menos duas gerações.

A partir das possíveis configurações iniciais e regras elaboradas por Conway, incontáveis padrões, isto é, aglomerados distintos de células vivas podem ser encontrados. Estes padrões podem ser divididos entre naturezas-mortas, *blinkers*, *gliders*, osciladores, *glider guns*, *puffer trains*, *wickstretchers* e outros tipos. Também foi feita a descoberta de que padrões antissimétricos eventualmente se tornam simétricos, e essa simetria pode ser apenas enriquecida, não perdida.

Na obra mencionada na seção anterior, *Winning Ways for Your Mathematical Plays*, Conway prova que o Jogo da Vida é Turing-completo, ou seja, pode computar qualquer coisa que computadores também consigam. Também provou que o autômato possui um construtor universal, ou seja, um padrão que constrói qualquer outro padrão que envolva síntese de *gliders* (BERLEKAMP; CONWAY; GUY, 2018).

2.3 Padrões

Os padrões encontrados no Jogo da Vida de Conway são fundamentais para entender as interações entre células vivas e mortas e como as mudanças de estado afetam o funcionamento da rede. De fato, as descobertas de formas dinâmicas como osciladores e *gliders* podem ser listadas como as principais causas de interesse popular no Jogo da Vida e, consequentemente, em autômatos celulares.

Como autômatos finitos eventualmente evoluem em ciclos de estados que se repetem indefinidamente (ADAMATZKY, 2010), grande parte das formas estudadas envolve ciclos de diferentes períodos. Além disso, a maioria possui algum nível de simetria em sua estrutura.

Neste trabalho, serão descritos os padrões mais comuns encontrados para o Jogo da Vida de Conway (isto é, B3/S23) e seus funcionamentos, a fim de elucidar a respeito das inúmeras possibilidades de "espécimes" apresentadas e como suas descobertas influenciaram os autômatos celulares posteriores. A fim de simplificar a visualização em tempo real da aplicação das regras B3/S23 e as evoluções temporais dos padrões descritos a seguir, sugere-se executá-los nesta simulação *online* (MARTIN, 2004).

2.3.1 Naturezas-mortas

Naturezas-mortas são os ciclos mais simples encontrados no Jogo da Vida de Conway, de período 1. Assim, são padrões completamente estáticos, ou seja, não evoluem uma vez formados. Isto ocorre porque todas as células vivas envolvidas têm o número de vizinhos necessários para permanecer neste estado. Evidentemente, formações de apenas células inativas não são consideradas naturezas-mortas.

Existem naturezas-mortas compostas por duas figuras, geralmente simétricas, que não são estáveis se separadas, mesmo que não estejam conectadas por uma célula em comum. A razão disto é a estabilidade de uma depender exclusivamente de sua interação com a outra. Dessa forma, se separadas, não continuam naturezas-mortas.

Em contrapartida, há estruturas chamadas pseudo-naturezas-mortas, pois também são estáveis separadamente, cada parte se tornando uma natureza-morta completamente independente da outra.

No Jogo da Vida de Conway, foram descobertas 34 naturezas-mortas, além de 21 pseudo-naturezas-mortas. Naturezas-mortas figuram entre os padrões mais comuns encontradas no Jogo da Vida (ADAMATZKY, 2010).

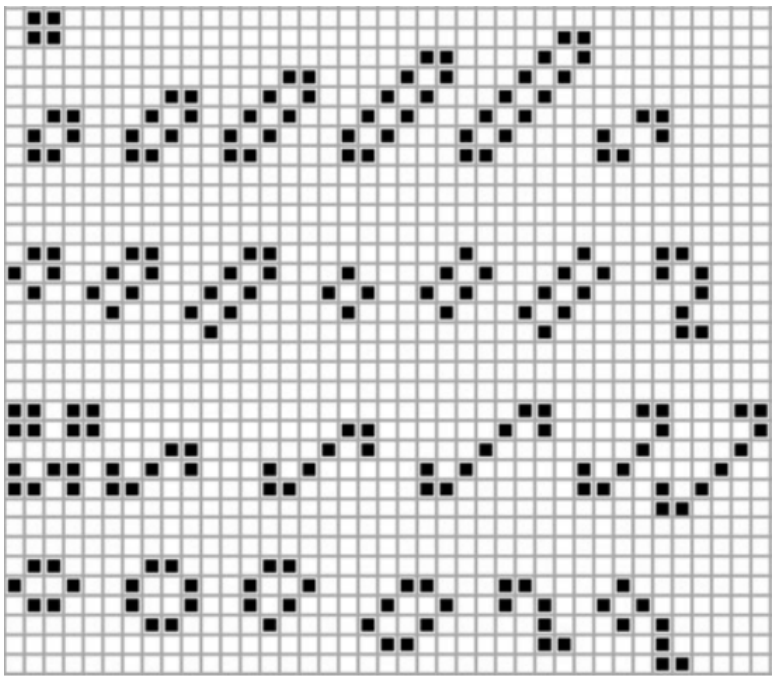


Figura 2.5: Esquema ilustrativo de algumas naturezas-mortas encontradas no Jogo da Vida (ADAMATZKY, 2010).

2.3.2 Alternadores

Alternadores são padrões de período dois, que oscilam indefinidamente, de uma geração a outra. A estrutura chamada *blinker*, que é um conjunto de três células vivas na horizontal ou na vertical, é uma das formas mais comuns que surgem após uma longa evolução, juntamente com a natureza-morta em forma de bloco de quatro células, exposta na seção anterior.

Alternadores podem se formar igualmente por superpopulação ou isolamento, a principal diferença estando nas estruturas formadas em cada caso. Em geral, ocorrem porque a estrutura se encontra presa em um ciclo de criar e destruir células, devido à proximidade ou afastamento de vizinhos vivos de uma determinada célula. No caso de um *blinker*, por exemplo, a célula central se mantém graças aos vizinhos laterais, mas estes desaparecem por possuírem apenas a ela como vizinha. Ao mesmo tempo, o conjunto de três células vivas produz uma nova célula acima e abaixo do grupo.

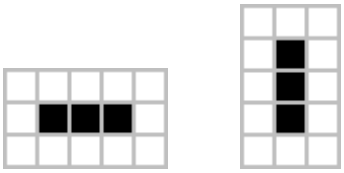


Figura 2.6: Um *blinker*, de uma geração a outra (ADAMATZKY, 2010).

Este tipo de estruturas possui números variados de células, sendo *blinker* o menor espécime possível. Em geral, possuem simetria. Por conta de sua variedade, não há uma contagem exata de alternadores, mas sim uma catalogação de formas mais frequentes.

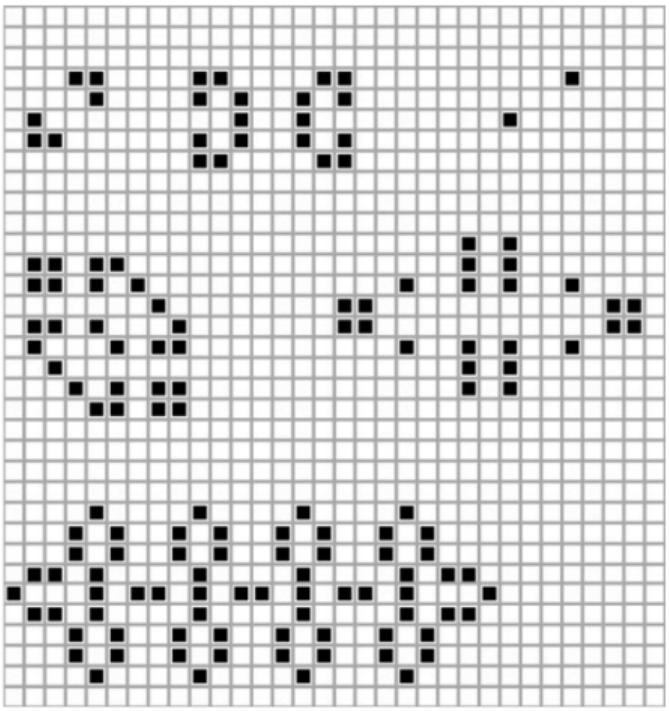


Figura 2.7: Esquema ilustrativo de alguns alternadores encontrados no Jogo da Vida (ADAMATZKY, 2010).

2.3.3 *Gliders*

Gliders são as estruturas mais famosas do Jogo da Vida de Conway (ADAMATZKY, 2010), e sua descoberta impulsionou a popularidade do conceito de autômatos celulares. *Gliders* são formas que parecem atravessar a rede, pois algumas células desaparecem e ressurgem adiante conforme mudam de estado. Além disso, possuem simetria, repetindo sua configuração com simetria de reflexão após duas gerações e retornando à orientação original após quatro gerações.

Apesar deste retorno, sugerindo periodicidade, não a possuem em uma rede infinita devido ao deslocamento que ocorre. Ainda por este motivo, não operam em um ciclo. Dessa forma, *gliders* apenas possuem periodicidade em uma rede finita, definida pelas gerações necessárias para o padrão atravessar a rede e voltar à sua posição original.

De certa forma, é possível pensar em *gliders* como mecanismos que transportam informações no Jogo da Vida. Ademais, se deslocam apenas diagonalmente. Em geral, são formados por apenas cinco células vivas. Por conta deste tamanho, *gliders* são facilmente encontrados após algumas gerações do jogo. Este fato apoia a afirmação de que estruturas estáveis são os resíduos mais comuns de evolução a longo prazo.

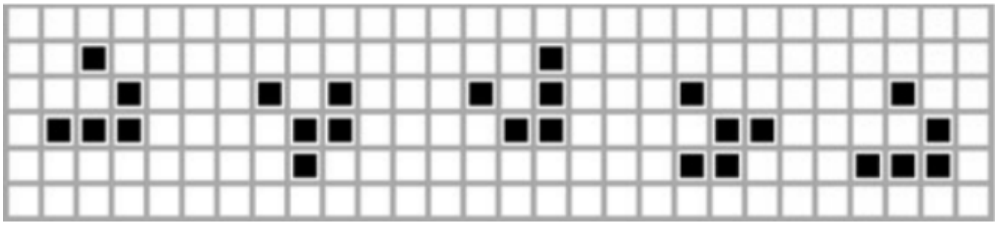


Figura 2.8: Ciclo de um *glider* e seu deslocamento (ADAMATZKY, 2010).

Formas maiores que se movimentam na rede não são consideradas *gliders*. Neste caso, logo foram encontrados padrões que se movem horizontal e verticalmente, mas não diagonalmente. Por estas diferenças, foi utilizada outra denominação, *spaceships*. No Jogo da Vida de Conway, existem apenas três *spaceships* distintas, apesar de ser possível encontrá-las em outros tamanhos e orientações, de forma simétrica.

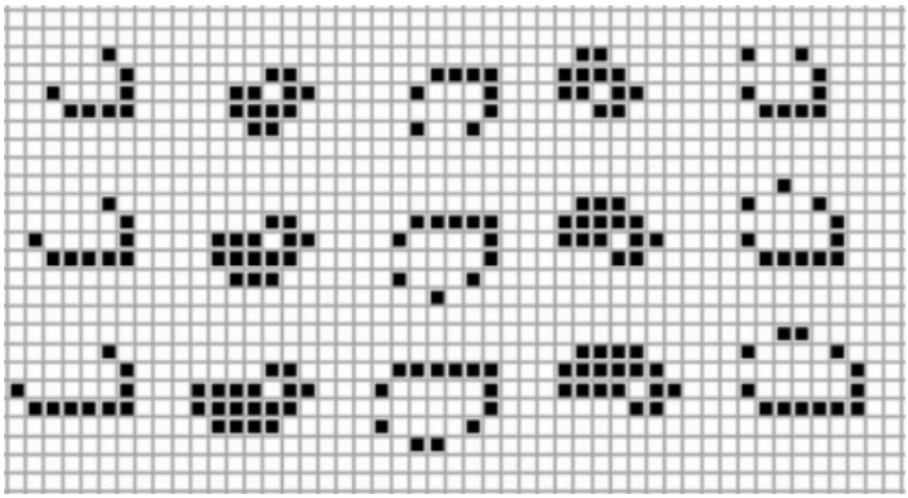


Figura 2.9: As três possíveis *spaceships* e suas variantes simétricas (ADAMATZKY, 2010).

Outra característica das *spaceships* é que estas deixam um rastro de células vivas por onde vão, que rapidamente desaparecem. Porém, *spaceships* mais largas podem produzir rastros duradouros. Combinações de *spaceships* de variados tamanhos podem produzir estruturas chamadas *puffer trains*, que serão discutidas mais adiante.

Gliders e *spaceships* permanecem no Jogo da Vida de Conway como as únicas estruturas primordiais capazes de autopropulsão, isto é, deslocamento autônomo.

2.3.4 Osciladores

Osciladores de períodos maiores que dois são raros no Jogo da Vida de Conway, e não são facilmente encontrados durante a evolução de uma rede ou como resíduos ao final desta. Desta forma, é usualmente empregada a construção deliberada destas estruturas (ADAMATZKY, 2010).

Estas estruturas são muito diversas em tamanho e período, especialmente por surgirem em situações específicas e geralmente de forma intencional. Algumas configurações são naturalmente expansivas, vertical e horizontalmente, e estas costumam resultar em ciclos de compressão e dilatação e períodos longos.

Existem muitas formas de criar osciladores em uma rede, que consistem principalmente em posicionar outras estruturas estrategicamente. Descobertas interessantes advindas destas experimentações incluem fileiras estáticas que se transformam em osciladores simétricos e eventualmente retornam a si.

Enquanto osciladores costumam ser extensos, comumente empregando dezenas de células vivas, seus períodos podem ser curtos ou longos, com alguns espécimes ultrapassando centenas de gerações.

Um tipo especial de oscilador se chama *wick*, que é constituído por uma região central que oscila indefinidamente, enquanto as extremidades podem ou não ser também osciladoras. Estes osciladores são relevantes por levarem ao surgimento de estruturas chamadas *wickstretchers*, a serem descritas.

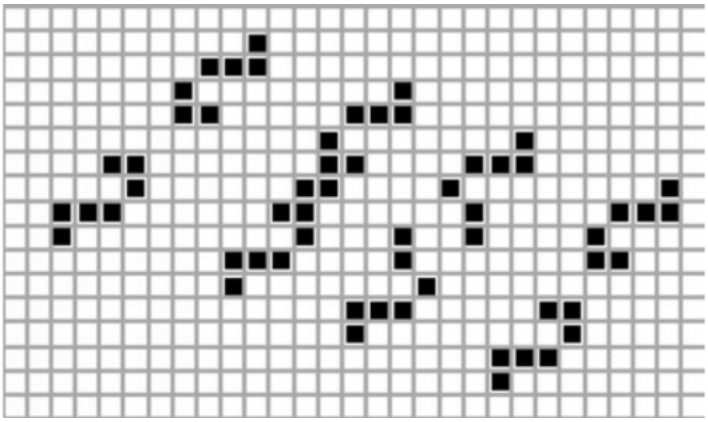


Figura 2.10: Ciclo de vida de um oscilador específico (ADAMATZKY, 2010).

2.3.5 *Glider Guns*

Glider guns, de forma simples, são estruturas capazes de lançar novos *gliders* à rede, como uma arma que os cria indefinidamente. Foram encontradas por meio de experimentações com colisões e quase colisões de *gliders* e *spaceships*. De fato, várias colisões e aproximações foram consideradas e testadas até que um conjunto específico de parâmetros levou à criação de uma estrutura que produz *gliders* sempre que houvesse a interação (ADAMATZKY, 2010).

Até a escrita deste trabalho, foram encontrados apenas dois *glider guns*, sendo um destes de grande importância para o Jogo da Vida de Conway por facilitar construções mais avançadas na rede e possibilitar novas descobertas.

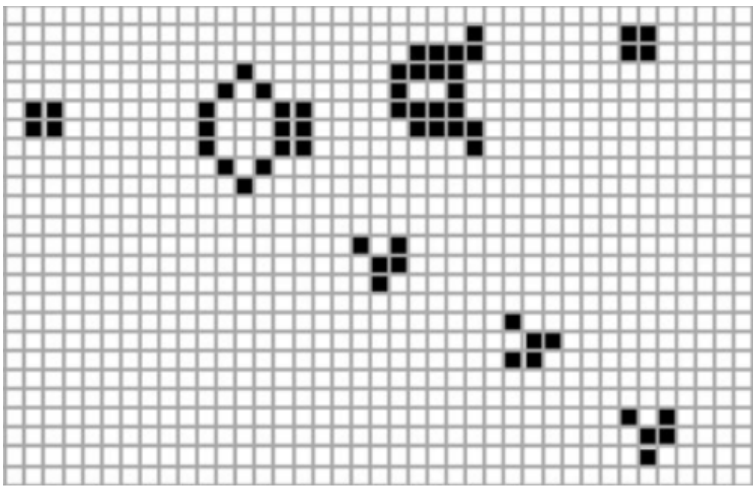


Figura 2.11: Um *glider gun* produzindo *gliders* (ADAMATZKY, 2010).

2.3.6 *Puffer Trains*

Puffer trains são definidos como padrões que, enquanto se locomovem pela rede, deixam um rastro duradouro por trás, como a fumaça de um trem. Embora inicialmente os resíduos se mostrem sem controle e altamente expansivos, é possível controlá-los utilizando outras estruturas, como *spaceships* e espécimes capazes de "consumir"as células restantes.

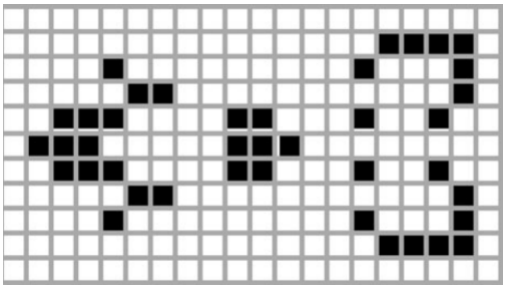


Figura 2.12: Um *puffer train* e seu rastro de células vivas (ADAMATZKY, 2010).

Este controle é essencial para o estudo destas formas, pois permite a sobrevivência de um *puffer train* e torna a nuvem de células esparsa. Um exemplo deste fato é a descoberta de *gliders* que habitam a fumaça do *puffer train*. Em geral, *puffer trains* são utilizados em conjunto para forçar interações entre gliders e possibilitar a criação de novas estruturas.

2.3.7 *Wickstretchers*

Wickstretchers são estruturas em que uma parte se estende indefinidamente, em autorreplicação, enquanto ao menos uma "ponta" se mantém estática. Em vários casos, uma extremidade da formação aparenta se deslocar desenfreadamente, estirando a estrutura que a conecta à outra extremidade, e esta não sofre alteração. Em alguns casos, um padrão repetido de nascimento e desaparecimento expulsa como "resíduos" algumas células vivas, que contribuem diretamente para a expansão da estrutura conectora.

Como ocorre para *puffer trains*, embora um *wickstretcher* aparente não ter controle, é possível controlá-lo por meio da interação com outras estruturas, que podem desintegrá-lo ou interromper sua expansão.

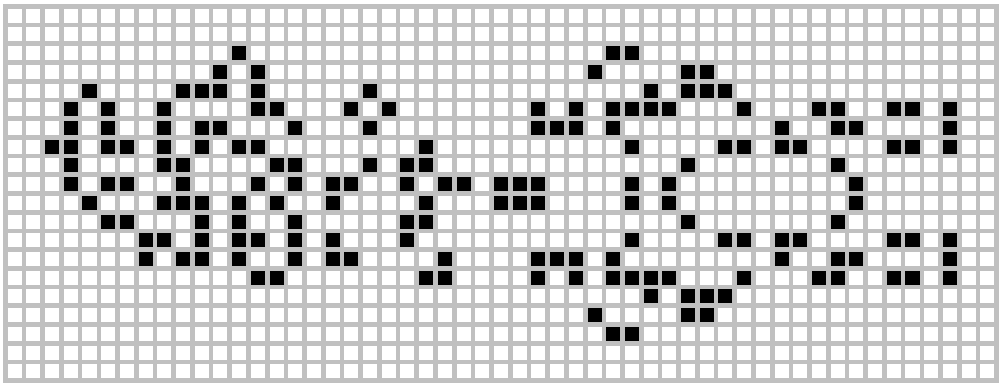


Figura 2.13: Um *wickstretcher* (ADAMATZKY, 2010).

Ao emparelhar um *wickstretcher* com um *wickeater*, um padrão que consome *wicks* ao gerar uma reação em cadeia, é possível construir *spaceships* de tamanho arbitrário, incluindo *spaceships* que possuem o comportamento de extensão observado em *wickstretchers*.

2.4 Variações e Autômatos Derivados

A alteração de parâmetros essenciais ao Jogo da Vida de Conway, como as regras impostas pelo matemático, pode trazer à tona novos resultados. Isto geralmente é feito levando em consideração o comportamento real de uma população, com a finalidade de realizar aproximações e previsões da progressão temporal desta. Outra razão comum para as mudanças, porém, pode ser resumida como curiosidade, a fim de explorar que novos padrões surgem e como certas regras e parâmetros afetam a rede.

Com este princípio, existem inúmeras variações do Jogo da Vida originadas da alteração arbitrária das regras que controlam a rede, B3/S23. Apesar das possibilidades aparentemente infinitas, é evidente que alguns conjuntos de regras adquirem mais popularidade do que outros, graças aos seus resultados.

Ademais, buscando objetivos específicos além do Jogo da Vida, outros autômatos celulares foram criados. Algumas problemáticas apresentadas nesses modelos são consideradas a inspiração de outros autômatos celulares como SmoothLife (RAFLER, 2011) e Lenia (CHAN, 2019), portanto serão discutidos de maneira mais extensa.

Larger Than Life

Como citado anteriormente, no Jogo da Vida, um *glider* é estável e se move diagonalmente pela rede. Então, começou-se uma busca por *gliders* que se movem em outras direções, incluindo a possível existência de algum que se movesse em todas as direções. Estes *gliders* com outros deslocamentos são ausentes no Jogo da Vida de Conway. Dessa forma, tornou-se necessário teorizar algumas alterações cujos parâmetros tornassem a rede propícia a formá-los. Chegou-se à conclusão de que estes parâmetros incluem uma vizinhança maior e descrita pelo raio de uma esfera. O primeiro modelo baseado no Jogo da Vida de Conway a fazer esta consideração foi Larger Than Life, por Kellie Michele Evans em 1996. (EVANS, 1996)

Em Larger Than Life, ou LtL, uma célula arbitrária tem sua vizinhança com um de vários possíveis formatos, desde vizinhança de Von Neumann ou vizinhança de Moore, circular, cruciforme ou outras possibilidades. No caso específico de uma vizinhança circular, esta seria descrita por um raio r, e o número de vizinhos dado por $(2r-1)\times 2-1$. O modelo também permite vizinhanças mais extensas, que vão além das células adjacentes à principal.

Além disso, os estados vivo e morto são dados por intervalos, ditos intervalos de nascimento e sobrevivência. Estes intervalos contêm dois números cada, ditos preenchimentos. Assim, uma célula nasce ou sobrevive a depender se a vizinhança ativa está ou não contida nestes intervalos. Por exemplo, em LtL, o Jogo da Vida de Conway teria o intervalo de nascimento $[3, 3]$, já que 3 é o único número de vizinhos vivos permitido para um nascimento, e o intervalo de sobrevivência seria $[3, 4]$, as únicas quantidades de vizinhos vivos que permitem a sobrevivência de uma célula.

Ademais, LtL permite o uso de regras do modelo Generations, uma versão do Jogo da Vida em que as células "envelhecem" antes de desaparecer, com estados para decair ou ascender e gravando a memória de células mais antigas (WOJTOWICZ, 2001).

RealLife

Uma extensão do modelo de Evans é fazer com que o raio da vizinhança tenda a infinito. Evans conjecturou que, neste caso, o autômato celular LtL convergiria para o "limite do contínuo". Assim, a célula principal se tornaria um ponto infinitesimal. Este caso particular foi investigado matematicamente por Marcus Pivato em 2007, que compilou suas descobertas em um novo autômato celular - RealLife. (PIVATO, 2007)

Apesar de sua criação, não foi investigado experimentalmente por seu criador, mas sim surgiu como um modelo teórico. Ou seja, não foram realizadas simulações, mas cálculos a fim de prever que resultados seriam obtidos. Assim, o autor demonstrou matematicamente a existência de naturezas-mortas em RealLife e que nele surgiriam alguns padrões presentes em Larger Than Life.

Além disso, Pivato provou matematicamente que RealLife é um autômato euclidiano, ou seja, muda de um estado a outro a depender da sequência de entradas fornecida e da condição inicial. A memória deste tipo de autômato é restrita pelo número de estados que possui.

Conforme o próprio pesquisador, pouco se sabe sobre a evolução temporal de seu modelo. Dessa forma, múltiplas questões foram deixadas em aberto para que fossem exploradas pela comunidade acadêmica. Por exemplo, foi possível provar matematicamente a existência de naturezas-mortas em RealLife, mas não de osciladores ou estruturas autopropulsionadas como *gliders*. O autor destaca que, enquanto há evidência experimental suficiente destes padrões em Larger Than Life, há poucos teoremas que os comprovem, e seu autômato possui sequer evidência teórica ou empírica destas estruturas.

Outro questionamento destacado por Pivato é a condição propícia para a formação dos padrões persistentes citados anteriormente, dado que a periodicidade não é necessária para seu surgimento em autômatos euclidianos. Por fim, o autor nota que enquanto autômatos celulares utilizando o modelo Larger Than Life podem exibir computação universal, tornando-os Turing-completo, este atributo e autorreplicação não são conhecidos em RealLife, devido ao fato de ser um autômato euclidiano.

Capítulo 3

SmoothLife

3.1 Estrutura

SmoothLife é um autômato celular criado por Stephan Rafler em 2011 (RAFLER, 2011). Baseado no Jogo da Vida de Conway, é descrito como uma generalização do mesmo para o domínio contínuo. Sua origem foi a exploração da problemática do Jogo da Vida com células infinitesimais e vizinhança como o raio de uma esfera que tende ao infinito.

Rafler decidiu abordar o caso descrito pelo autômato RealLife (PIVATO, 2007) de forma diferente. Em vez da célula principal ser um ponto infinitesimal, conjecturou que houvesse tamanho finito e forma circular, enquanto a vizinhança geral assume forma de anel. Este formato de anel divide a região em duas partes, externa e interna, esta contendo a célula central. Além disso, têm-se múltiplos estados, sendo estes números decimais entre 0 e 1, sem as categorias "vivo" e "morto" utilizadas no Jogo da Vida.

Buscando simplificar as interações da célula com a vizinhança, considerando a complexidade que resultaria de contabilizar cada uma individualmente, utiliza-se a teoria de campo médio. A teoria de campo médio consiste em tratar efeitos individuais como um único efeito, reduzindo um problema de múltiplos corpos a um problema de um só corpo e tornando possível a obtenção de uma solução aproximada (KADANOFF, 2009). Assim, não é necessário denotar cada célula da vizinhança, mas sim utilizar preenchimentos, como em Larger Than Life (EVANS, 1996).

Para o Jogo da Vida, seria possível supor uma função qualquer $f(x,t)$ para determinar o estado da célula. Em SmoothLife, porém, o estado não seria dado pelo valor da função em um dado ponto, mas sim pelos preenchimentos, ou seja, as áreas das regiões que dividem o círculo neste ponto. Estes preenchimentos são ditos m (área interna) e n (área externa), podem ser escritos como funções e utilizam-se fatores de normalização (N e M) para que estejam entre 0 e 1, visando estados neste intervalo.

Assim, as equações para os preenchimentos n e m são dadas por:

$$n = \frac{1}{N} \int_{r_i<|\vec{u}|<r_a} d\vec{u} f(\vec{x}+\vec{u}, t) \tag{3.1}$$

$$m = \frac{1}{M} \int_{|\vec{u}|<r_i} d\vec{u} f(\vec{x}+\vec{u}, t) \tag{3.2}$$

A distância entre a célula central e um ponto arbitrário, em SmoothLife, corresponde à distância euclidiana, isto é, a menor distância entre os dois pontos. Por um padrão imposto por Rafler, o raio da vizinhança externa, denotado por r_a, é três vezes maior que o raio da vizinhança interna, r_i.

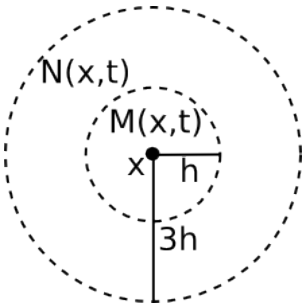


Figura 3.1: Vizinhança no autômato SmoothLife, com sua divisão de áreas e diâmetro. Os preenchimentos aqui são funções denotadas por $N(x,t)$ e $M(x,t)$, enquanto os raios das áreas são dados em termos de h (LYSENKO, 2012).

Assim, em vez de haver uma matriz de transição (SIMON; BLUME, 1994) como no Jogo da Vida de Conway, em que a atualização do estado da célula depende de seu estado e da soma dos estados dos vizinhos, há uma função de transição $s(n,m)$ regida pelos preenchimentos. Como em Larger Than Life (EVANS, 1996), SmoothLife ainda depende de quatro parâmetros, que são os limites de dois intervalos, de nascimento e desaparecimento. Estes quatro números reais são representados comumente como b_1, b_2 (nascimento), d_1 e d_2 (desaparecimento). Os intervalos são relacionados a partir de funções sigmoides, σ_1 e σ_2, para enfatizar o aspecto suave do autômato celular. As funções sigmoides propostas por Rafler são dadas por:

$$\sigma_1(x,a) = \frac{1}{1+\exp\left(-(-x-a)4/\alpha\right)} \tag{3.3}$$

$$\sigma_2(x,a,b) = \sigma_1(x,a)(1-\sigma_1(x,b)) \tag{3.4}$$

$$\sigma_m(x,y,m) = x(1-\sigma_1(m,0,5)) + y(\sigma_1(m,0,5)) \tag{3.5}$$

Finalmente, a função de transição $s(n,m)$ é definida pela seguinte equação, que depende das funções anteriores:

$$s(n,m) = \sigma_2(n, \sigma_m(b_1,d_1,m), \sigma_m(b_2,d_2,m)) \tag{3.6}$$

A largura do passo temporal é dada por α, presente na equação 1.3. Por serem necessárias duas versões de σ_m, tem-se então α_n e α_m, outros dois valores reais utilizados para gerenciar a suavidade dos passos temporais. Enquanto estas funções sigmoides são necessárias para o pleno funcionamento do autômato celular, a suavidade destas não é. Em essência, este ainda depende dos intervalos de nascimento e desaparecimento.

Finalmente, a progressão temporal neste modelo pode ser discreta ou contínua ao utilizar decimais ou infinitesimais, estes chamados dt. Tem-se que $dt = 1/T$, sendo T o número de passos temporais por geração. A seguir, a equação que rege a progressão temporal de SmoothLife:

$$f(\vec{x}, t+dt) = f(\vec{x},t) + dtS[s(n,m)]f(\vec{x},t) \tag{3.7}$$

O uso de valores altos para T dá origem a SmoothLifeL, a versão contínua de SmoothLife. De forma simplificada, se SmoothLife possui função de transição $s_d(m,n)$, SmoothLifeL possuirá função de transição $s_c(m,n) = 2s_d(m,n) - 1$. De acordo com as predições matemáticas e observações de Rafler, as versões discreta e contínua possuem padrões exclusivos a elas. Realizando

o objetivo que movimentou modelos anteriores como Larger Than Life e RealLife, Rafler encontrou um *glider* que se move em direções arbitrárias em SmoothLife discreto, nomeado *smooth glider*.

Figura 3.2: Um *smooth glider* obtido para $r_a = 21$, $b_1 = 0,278$, $b_2 = 0,365$, $d_1 = 0,267$, $d_2 = 0,445$, $\alpha_n = 0,028$ e $\alpha_m = 0,147$ se deslocando para o canto superior direito (RAFLER, 2011).

SmoothLife é um autômato que pode ser reproduzido para qualquer número de dimensões. A versão padrão, proposta por Rafler, é bidimensional.

3.2 Padrões

3.2.1 SmoothLife Discreto

Em geral, padrões encontrados em SmoothLife discreto são instáveis e rapidamente desaparecem ou se transformam. A figura principal desta versão é o *smooth glider*, exclusivo a ela, que é estável para $dt = 1$ mas desaparece após algumas gerações para valores menores. Este fato representa um contraste ao Jogo da Vida de Conway, pois indica que SmoothLife em sua forma discreta não possui a mesma variedade de figuras.

Smooth gliders são, de fato, as figuras "unitárias" desta versão, pois outras formas surgem a partir da interação destes. As interações ainda podem possuir caráter destrutivo, aniquilando as partes. No caso oposto, as figuras resultantes são compostas de múltiplos estados rapidamente relacionando-se e que parecem contrair e expandir em direções arbitrárias. Costumam seguir a forma curva das figuras de origem, porém assimétricas e altamente instáveis. Estas regiões encontram dois possíveis fins: criam novos *smooth gliders* ou desaparecem completamente.

Por vezes, *smooth gliders* podem expandir sua largura e se dividir para dar origem a novos *smooth gliders*, como em um processo mitótico. Assim, para $dt = 1$, a única forma estável é o *smooth glider*, e a existência de figuras como naturezas-mortas e alternadores não é conhecida.

3.2.2 SmoothLifeL

Para valores menores de dt, como $dt = 0,5$ e adiante, considera-se SmoothLifeL. Nestes casos, a saída encontrada na simulação tem maior dependência na condição inicial da rede e nos parâmetros utilizados. Em geral, os padrões encontrados nesta versão se caracterizam por sua estabilidade.

Fins comuns incluem o desaparecimento completo de células ativas da rede, o surgimento de naturezas-mortas em formato de anel ou de uma figura massiva que eventualmente domina toda a rede. Esta figura é chamada "massa expansiva" por esta propriedade, e é caracterizada por numerosos sulcos de células ativas e espaços vazios entre eles. Estes sulcos variam proporcionalmente em espessura e profundidade ao raio da vizinhança e ocorrem mudanças de estado periódicas entre eles, abrindo e fechando lacunas sucessivamente.

SmoothLifeL possui outros *gliders* que assumem o papel do *smooth glider*. Porém, estes *gliders* não coexistem, mas sim são exclusivos a parâmetros específicos acerca da rede e da

função de transição desta. Dessa forma, variam em estrutura e comportamento, embora mantenham a característica básica de deslocamento. Os dois espécimes exclusivos a SmoothLifeL são chamados *pulsating glider*, aquele que parece pulsar, e *wobbly glider*, aquele que parece cambalear.

Figura 3.3: Três *gliders*: *smooth glider*, *pulsating glider* e *wobbly glider*, respectivamente. As setas indicam que são omnidirecionais (CHAN, 2019).

Em casos de uma condição inicial desproporcional, aleatoriamente distribuindo regiões de células ativas na rede, numerosas figuras distintas e exclusivas são encontradas. Pode-se citar várias naturezas-mortas, osciladores e estruturas estáveis. Figuras notáveis deste modelo incluem *gliders* longos, duplas de *gliders* que rotacionam em conjunto, *wickstretchers* e ameboides vinculados por meio de "cordas". Estes padrões, diferentemente dos citados anteriormente, são dinâmicos, rapidamente se desfazendo e refazendo conforme interações com outros padrões e a progressão temporal da rede.

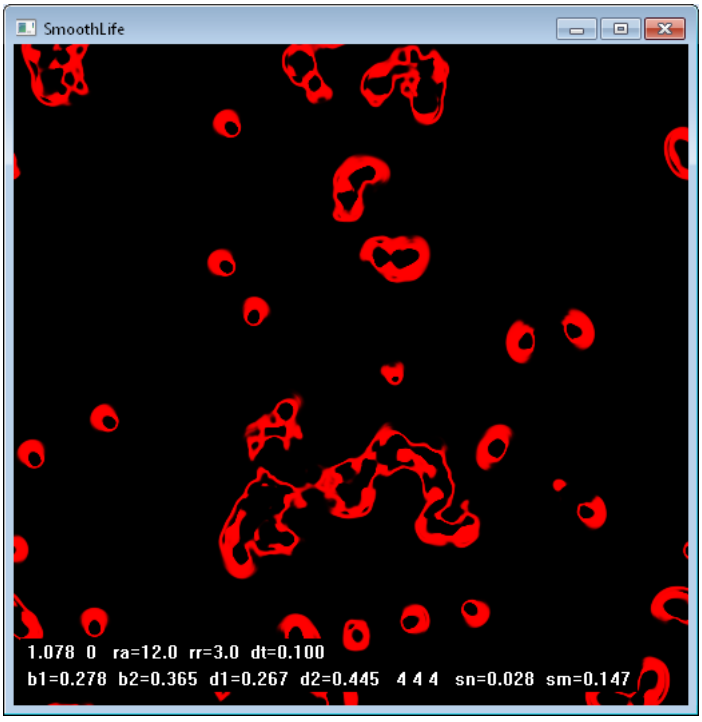


Figura 3.4: Uma simulação de SmoothLife discreto, com células ativas em vermelho. Para esta simulação, fez-se: $r_a = 12$, $dt = 1$, $b_1 = 0,278$, $b_2 = 0,365$, $d_1 = 0,267$, $d_2 = 0,445$, $\alpha_n = 0,028$ e $\alpha_m = 0,147$ (RAFLER, 2012).

Figura 3.5: Uma simulação de SmoothLifeL, com células ativas em verde e azul. Os parâmetros utilizados são $r_a = 31,8$, $dt = 0,157$, $b_1 = 0,092$, $b_2 = 0,098$, $d_1 = 0,256$, $d_2 = 0,607$, $\alpha_n = 0,015$ e $\alpha_m = 0,340$ (RAFLER, 2012).

Capítulo 4

Lenia

4.1 Primordia e os primórdios de Lenia

Lenia (do latim *lenis*, suave) é um autômato celular criado por Bert Kwang-chak Chan, publicado em 2019 (CHAN, 2019). Este modelo baseou-se principalmente no Jogo da Vida de Conway (GARDNER, 1970), buscando sua generalização máxima. Por consequência, também tem inspiração nos modelos SmoothLife (RAFLER, 2011) e Larger Than Life (EVANS, 1996).

Em sua pesquisa nas áreas de autômatos celulares e vida artificial, Chan concentrou-se em duas das quatorze problemáticas abertas sobre tal tema: tornar possível simular vida biológica existente e criar novas formas de vida utilizando sistemas artificiais (BEDAU et al., 2000). Estes dois tópicos se tornaram objetivos a longo prazo para o engenheiro de software de pesquisa. Buscando aproximar-se de suas realizações, Chan criou primeiramente o autômato celular Primordia em 2015 (CHAN, 2015a). Por suas propriedades e resultados, é considerado o precursor de Lenia.

Em Primordia, utiliza-se regras similares ao modelo Generations (WOJTOWICZ, 2001), citado anteriormente em Larger Than Life. A gama de estados que uma célula pode assumir é extensa, e as regras de crescimento dependem de parâmetros específicos, sem valores padrão. Em suma, uma célula passa para um estado de maior valor se a soma dos estados de sua vizinhança está contida em uma faixa de crescimento dada pelos parâmetros mencionados. A mesma célula sobrevive, ou mantém seu estado, se o valor está contido em outra faixa, dita faixa de sobrevivência. Enfim, passa para um estado reduzido ou desaparece em quaisquer outros casos.

A importância deste autômato para a gênese de Lenia se encontra nas descobertas de Chan, que estabeleceram as fundações para a criação do último. Utilizando os parâmetros certos e as regras impostas, encontrou-se um padrão único, chamado *Orbium*. Este fato, juntamente com a evidência de outros vários espécimes exclusivos a Primordia, alavancou os esforços para o desenvolvimento de um autômato celular mais sofisticado.

Figura 4.1: Ciclo de vida de um *Orbium* em Primordia (CHAN, 2015a).

4.2 Estrutura

Então, para a criação de um autômato celular com maior grau de complexidade e semelhança com vida biológica, Chan considerou os atributos necessários para vida artificial de forma a fielmente reproduzi-la, como morfogênese, mobilidade, crescimento, homeostase, evolucionabilidade e autorreplicação (KOSHLAND, 2002; MCKAY, 2004; SCHRöDINGER, 1974).

Inspirado por sua criação anterior, o pesquisador notou que uma maneira de se aproximar da reprodução destes atributos em sistemas artificiais provém de estudar a aproximação para a continuidade, como feito em SmoothLife. Assim, utilizando a generalização da maior quantidade de aspectos possível do Jogo da Vida de Conway, nasce o autômato celular Lenia, em 2017.

Primeiramente, os estados vivo e morto se estendem em uma faixa fracionada, que pode ser dividida em infinitesimais. Assim, como em SmoothLife e Primordia, há múltiplos estados. Porém, em Lenia, cada estado é uma relação matemática envolvendo o anterior, e as atualizações das células ocorrem iterativamente, ou seja, de forma repetitiva. Além disso, a vizinhança de uma célula é descrita por uma esfera de raio R, centrada na célula. A equação que rege a esfera engloba os vetores de posição que representam as células vizinhas, isto é, suas posições relativas.

Como em todos os autômatos celulares discutidos, para atualizar o estado de uma dada célula, considera-se os estados das células vizinhas. As interações com a vizinhança ocorrem por meio de uma soma ponderada, e cada interação possui peso a depender do vetor de posição com respeito à célula principal. Para contabilizar as interações entre a célula e a vizinhança, como em SmoothLife, utiliza-se a teoria de campo médio.

Assim, ao utilizar a teoria de campo médio, calcula-se uma função que representa toda a vizinhança, chamada *kernel* ("núcleo"), denotada por $K(n)$. Esta decompõe-se em outras duas funções, ditas *kernel core* $K_C(r)$ (centro) e *kernel skeleton* (esqueleto), $K_S(r, \beta)$.

A função K_C é definida radialmente, e seu argumento é a distância polar, que depende do raio R definido pelo pesquisador. Porém, pode ser alterada para outros formatos de vizinhança, e segue condições de contorno (ao menos $K_C(0) = K_C(1) = 0$). A equação a seguir é a forma padrão utilizada para Lenia, com $\alpha = 4$:

$$K_C(r) = \exp\left(\alpha - \frac{\alpha}{4r(1-r)}\right) \tag{4.1}$$

A função K_S depende de K_C, pois a expande e transforma o núcleo em anéis concêntricos. Os anéis evidenciam regiões com picos de peso estatístico, cujas alturas são dadas por β, que é um vetor dado por $(\beta_1, \beta_2, ..., \beta_B)$, em que B é a ordem utilizada. Além disso, todos os valores de β devem estar entre 0 e 1. A equação padrão para a função K_S é como segue:

$$K_S(r; \beta) = \beta_{\lfloor Br \rfloor} K_C(Br \mod 1) \tag{4.2}$$

Então, obtém-se $K(n)$ com K_S e os vetores de posição, e para manter-se entre 0 e 1, ela é normalizada:

$$K(n) = \frac{K_S\left(||n||_2\right)}{|K_S|} \tag{4.3}$$

Após obter a vizinhança e seus pesos por meio de $K(n)$, é feita uma convolução desta função com o estado atual da célula, $A^t(x)$, para obter uma distribuição de potencial, $U^t(x)$:

$$U^t(x) = K * A^t(x) \tag{4.4}$$

Este potencial é necessário para obter o valor de uma função dita *growth mapping* (mapeamento de crescimento), $G(u;\mu,\sigma)$, que depende de três parâmetros: u (o valor do potencial no dado ponto), μ (constante, centro de crescimento) e σ (constante, largura de crescimento). Os parâmetros μ e σ, então, controlam a curva desta função. Além disso, é necessário que $G(\mu) = 1$. A forma padrão utilizada em Lenia é dada pela seguinte equação:

$$G(u;\mu,\sigma) = 2\exp\left(-\frac{(u-\mu)^2}{2\sigma^2}\right) - 1 \tag{4.5}$$

Finalmente, este mapeamento de crescimento é utilizado para obter a distribuição de crescimento em si, dada por:

$$G^t(x) = G(U^t(x)) \tag{4.6}$$

Então, de acordo com a distribuição, atualiza-se o estado da célula adicionando uma pequena fração de tempo Δt (ou infinitesimal dt) ao tempo atual. Matematicamente, tem-se:

$$A^{t+\Delta t}(x) = [A^t(x) + \Delta t G^t(x)]_0^1 \tag{4.7}$$

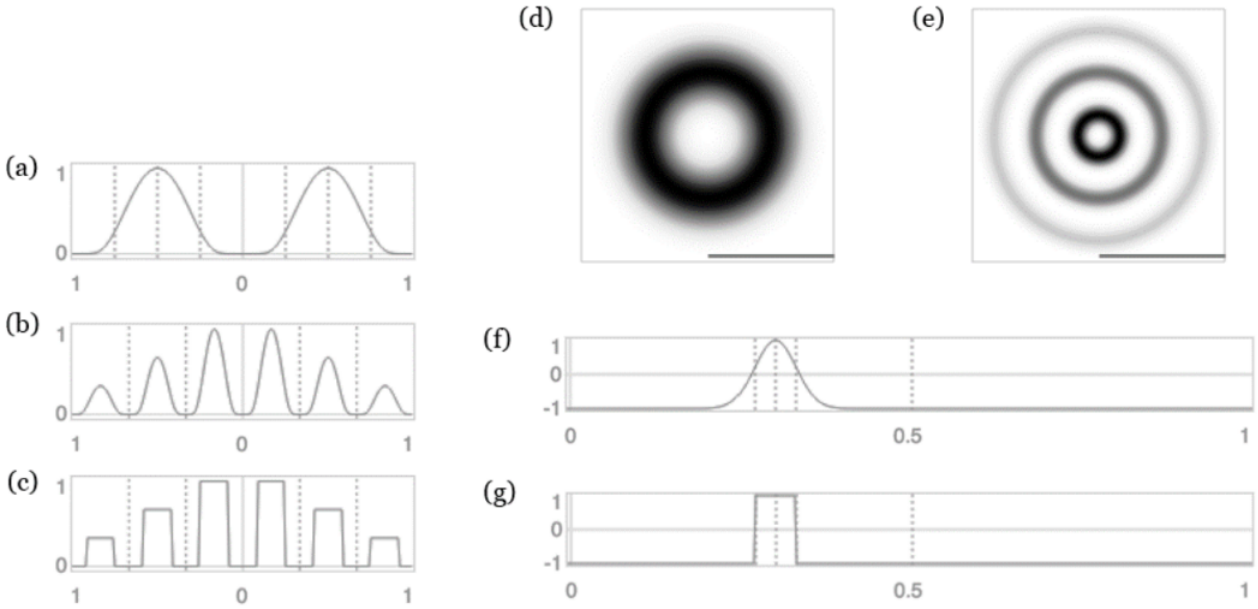


Figura 4.2: Gráficos acerca da vizinhança em Lenia. As figuras de **(a)** a **(c)** representam cortes transversais da vizinhança, dividida em centro (K_C) **(a)** e esqueleto (K_S) (**(b)** e **(c)**), e seus pesos de convolução retratados pelos picos relacionados a β. As imagens **(d)** e **(e)** retratam a vizinhança graficamente, em que as áreas mais escuras indicam maior peso estatístico. Os gráficos **(f)** e **(g)** indicam a função mapeamento de crescimento $G(u;\mu,\sigma)$, com $\mu = 0,3$ e $\sigma = 0,03$, usando função exponencial **(f)** ou retangular **(g)** (CHAN, 2019).

Assim, o funcionamento de Lenia se resume a:

1. Os parâmetros referentes às dimensões da simulação: o raio R, o número de passos por geração T e o número de possíveis estados P, bem como as frações Δx, Δt e Δp ou infinitesimais dx, dt e dp;

2. Os parâmetros associados às funções: μ, σ, β;

3. As funções K_C e G, pois delas dependem as outras.

Apesar das equações citadas serem consideradas as equações padrão para Lenia, é possível modificá-las para obter outros resultados. De fato, isto foi feito para inúmeras espécies encontradas por Chan e outros colaboradores, a serem discutidas posteriormente. Outros parâmetros numéricos como R, μ e σ também podem sofrer alterações.

Entre as funções que podem ser alteradas, destacam-se K_C e G, as funções *kernel core* e mapeamento de crescimento. Estas podem assumir formatos exponenciais, polinomiais, retangulares e outros, a depender do intuito do pesquisador. Estas mudanças acarretam a transformação completa da rede, permitindo espécimes de comportamentos e dinâmicas variados, incluindo a reprodução de outros autômatos celulares em Lenia, como o Jogo da Vida de Conway e o SmoothLife.

Uma série específica de alterações em Lenia visa aproximar ainda mais o autômato à continuidade, com alguns parâmetros divididos em infinitesimais ou tendendo a infinito. Esta busca por um autômato verdadeiramente contínuo leva Lenia a se fragmentar em duas vertentes, Lenia discreto e Lenia contínuo. Lenia discreto utiliza puramente medidas fracionadas ou inteiras e é, de fato, o autômato celular Lenia padrão, enquanto Lenia contínuo é uma aproximação teorizada que utiliza a aproximação ao infinito. Assim, a vertente que é enfoque neste trabalho é a que permite simulações e experimentação, ou seja, Lenia discreto.

4.2.1 Lenia Discreto

A versão discreta de Lenia generaliza o Jogo da Vida de Conway ao estender e normalizar as dimensões de espaço, tempo e estado. O conjunto de estados é configurado para $\{0, 1, 2, 3, ..., P\}$, e a vizinhança é uma esfera discreta com raio R. Além disso, há a quantidade de passos dados por geração, T. Os parâmetros R, T e P são valores inteiros.

Para normalizá-los, são introduzidas as variáveis Δx, Δt e ΔP, chamadas respectivamente de distância local, passo temporal e precisão de estado. Matematicamente, $\Delta x = 1/R$, $\Delta t = 1/T$ e $\Delta P = 1/P$. Estes deltas são utilizados no cálculo da distribuição de potencial e, consequentemente, na determinação da distribuição de crescimento e do estado a ser assumido pela célula em questão.

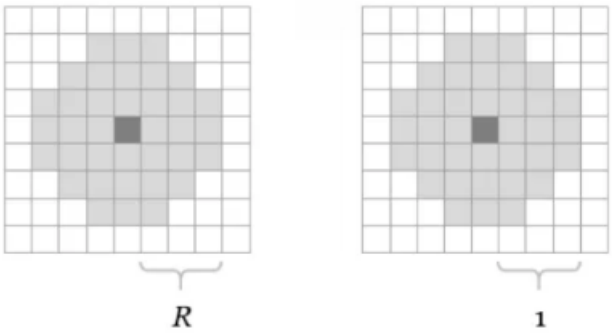


Figura 4.3: A vizinhança usada em Lenia discreto, uma esfera discreta de raio R **(b)** e sua normalização **(c)** (CHAN, 2019).

4.2.2 Lenia Contínuo

A versão contínua de Lenia é uma hipótese ao modificar as dimensões de espaço, tempo e estado de Lenia discreto. Em Lenia contínuo, R, T e P tendem a infinito, o que causa

a transformação das frações Δx, Δt e Δp em infinitesimais. Como discutido anteriormente, os parâmetros citados são fundamentais na determinação do potencial, do crescimento e dos possíveis estados. Este fato implica a alteração indireta de todos os outros parâmetros e funções associadas. Visto que esta vertente é teorizada, não possui espécimes descobertos ou método experimental.

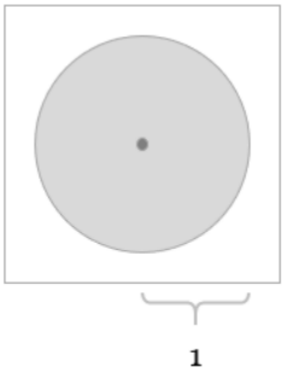


Figura 4.4: A vizinhança usada em Lenia contínuo, uma esfera suave normalizada (CHAN, 2019).

4.3 Padrões

4.3.1 Métodos

A simulação e o estudo de Lenia revelaram que este autômato é provido de tal profundidade e complexidade que é possível traçar incontáveis paralelos com sua vida artificial e a vida biológica na Terra. Assim, Chan utilizou vários conceitos e terminologias da biologia para explicar e categorizar suas formas de vida, como taxonomia, nomenclatura, ecologia, morfologia, comportamento, fisiologia e alometria. Aqui, serão discutidos os métodos utilizados por Chan e outros pesquisadores para encontrar a diversidade de Lenia.

Um padrão autônomo e que evolui automaticamente em Lenia é chamado de forma de vida, e formas de vida similares são parte de uma espécie. Até o presente momento, mais de 400 espécies foram encontradas por Chan e outros pesquisadores em Lenia. Estas espécies espalham-se em 18 famílias, que agregam suas características visuais e comportamentais. Estas famílias então dividem-se entre seis ordens, e estas em três classes.

A alta riqueza de Lenia, porém, não é encontrada apenas realizando simulações automáticas. De fato, como para o Jogo da Vida, inúmeros padrões apenas surgem com interferência humana, ou seja, manipulação deliberada da rede ou de parâmetros para encontrar formas de vida específicas. Desta forma, são elencados alguns métodos utilizados para encontrar as mais de 400 espécies de Lenia:

- Simulações com condições iniciais pseudoaleatórias, das quais espécies surgem automaticamente;
- Manipulação de parâmetros, no qual uma rede com uma forma de vida pré-existente é submetida a uma simulação com parâmetros distintos daqueles que a deram origem;
- Exploração automática, em que uma forma de vida conhecida é submetida a inúmeras simulações com diferentes parâmetros simultaneamente;

- Mutação manual, onde uma forma de vida conhecida é manualmente visualmente alterada e submetida a novos parâmetros para que estabilize. Uma forma comum de alteração é a manipulação da simetria, invertendo eixos e rotacionando seções do corpo.

Cada método produz espécimes com atributos únicos, muitas vezes exclusivos a valores específicos dos parâmetros μ e σ ou circunstâncias particulares, logo dando surgimento às 18 famílias. Enquanto todos são efetivos para encontrar novas espécies, os mais prolíficos neste objetivo são simulações automáticas e a manipulação de parâmetros. Ademais, os métodos podem ser complementares, como no caso frequente em que padrões não estabilizados em uma simulação acabam por estabilizar-se em outra.

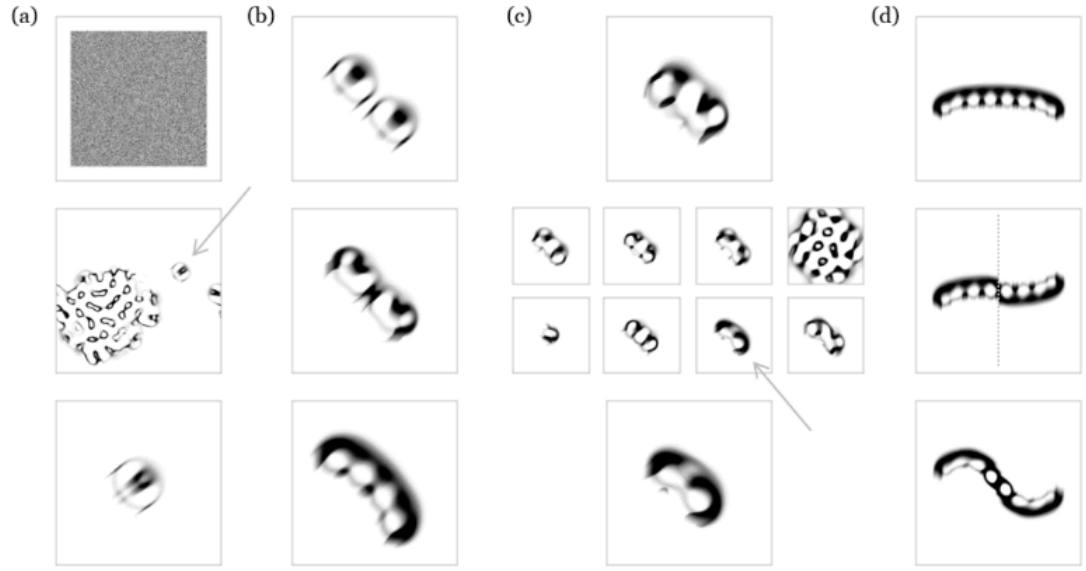


Figura 4.5: Os diferentes métodos para a obtenção de espécies em Lenia. As figuras da coluna **(a)** representam o método da simulação automática. Uma espécie (destacada pela flecha) eventualmente nasce da sopa primordial. A coluna **(b)** representa o método de manipulação de parâmetros. A coluna **(c)** expõe a exploração automática e as espécies distintas que surgem para cada simulação, sendo o espécime inédito destacado pela flecha. A coluna **(d)** representa o método da mutação manual (CHAN, 2019).

Os mecanismos exemplificados acima não apenas contribuem para o descobrimento de novos espécimes em Lenia, como expõem como espécies previamente descobertas adaptam-se a diferentes circunstâncias. Desta forma, reitera-se a dependência das formas de vida com o ambiente em que estão inseridas. Um exemplo gráfico do efeito das dimensões da rede e seus parâmetros sobre uma espécie está exposto na figura a seguir, em que estudou-se as características físicas e comportamentais de um *Orbium* em diferentes simulações.

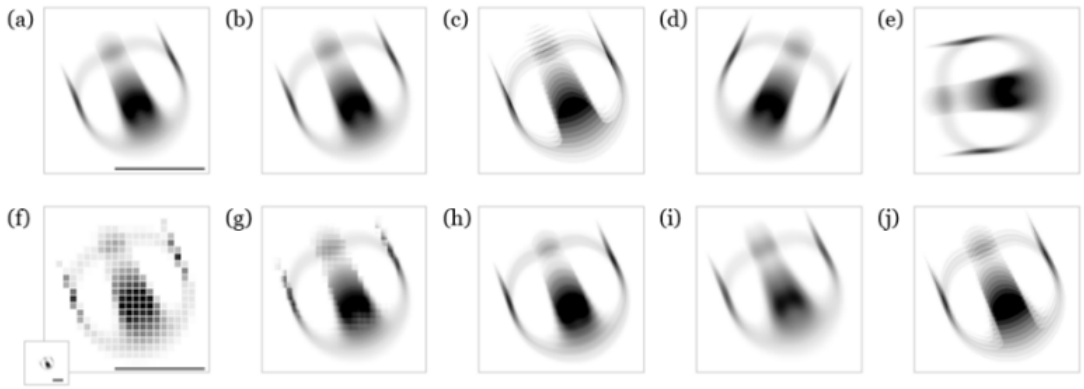


Figura 4.6: Um *Orbium* ($\mu = 0,15$ e $\sigma = 0,016$) submetido a diferentes configurações. A barra de escala equivale ao raio da vizinhança. A figura **(a)** expõe o *Orbium* original ($R = 185, T = 10, P > 10^{15}$) com funções exponenciais. As figuras **(b)** e **(c)** retratam este com funções polinomiais; não há diferença visível em **(b)**, mas há em **(c)**. As figuras **(d)** e **(e)** mostram *Orbium* invertido **(d)**) e rotacionado **(e)**), sem mudanças visíveis. De **(f)** a **(g)**, tem-se o espécime drasticamente reduzido para $R = 15$ **(f)**) (contém *zoom*; tamanho real é a pequena imagem à esquerda) e depois redimensionado para $R = 185$ **(g)**). Em **(h)**, o tempo é comprimido para $T = 5$, e em **(i)**, o tempo é dilatado para $T = 320$. Na figura **(j)**, o número de estados é comprimido para $P = 10$ (CHAN, 2019).

Em geral, como retratado na figura 4.6, há uma relação de proporcionalidade, pois funções com curvas mais suaves tendem a produzir espécimes mais suaves (figura 4.6 **(a)**), enquanto funções geométricas produzem espécimes de acordo (figura 4.6 **(c)**). Relacionando Lenia com SmoothLife, Chan sugere que este possui formas de vida semelhantes às encontradas no primeiro, especificamente quando são utilizadas funções retangulares.

4.3.2 Taxonomia

Nesta seção, será discutida a taxonomia de Lenia, destacando as famílias de espécimes encontradas e quais aspectos as formam.

Ao reunir os espécimes obtidos por meio dos métodos discutidos na seção anterior, alguns atributos em comum são percebidos, especialmente no que tange a características visuais e comportamentais. As possíveis razões para estas similaridades envolvem modificações parecidas nas dimensões da rede, nas funções utilizadas e/ou nos parâmetros deste autômato celular. Com este aspecto em consideração, Chan passou a categorizar seus espécimes. Tendo em vista as comparações entre as formas de vida de Lenia e a vida na Terra, o pesquisador optou por utilizar táxons de forma similar à taxonomia de espécies biológicas.

Uma característica específica que levou a inúmeras espécies semelhantes foi a alteração sistemática dos pesos de convolução nas diferentes regiões da vizinhança, ou seja, a alteração da função *kernel* $K(n)$ e seus picos β. Assim, a primeira divisão realizada na categorização dos padrões de Lenia teve como centro o peso das regiões da vizinhança: surgiram *Exokernel* (anéis mais externos com maior peso), *Mesokernel* (todos os anéis com mesmo peso) e *Endokernel* (anéis interiores com maior peso).

Entretanto, logo notou-se que esta separação não é suficiente para agrupar apenas espécimes visual e dinamicamente similares. Desta forma, estas categorias são tidas como as primordiais de Lenia, comparáveis às classes em biologia, e mostraram-se necessários mais níveis de organização. Logo, surgiram as ordens; a classe *Exokernel* possui duas, *Orbiformes*, que abriga espécimes similares ao *Orbium* e *Scutiformes*, literalmente "em forma de escudo". A classe *Mesokernel* abriga uma ordem, *Echiniformes*, com famílias de características bastante diversas.

Por fim, *Endokernel* possui três ordens, *Kroniformes*, *Radiiformes* e *Amoebiformes*.

Finalmente, foram criadas as 18 famílias de Lenia, e a maior parte contém subfamílias e gêneros. Estas últimas partições visam reunir características anatômicas e fisiológicas fundamentais das espécies. Embora Lenia seja bastante diverso, vários espécimes possuem comportamentos similares aos encontrados nos outros dois autômatos estudados, como naturezas-mortas, *gliders* e alternadores.

A árvore da vida artificial de Lenia está disposta a seguir; as famílias possuem siglas para facilitar a identificação dessas em figuras posteriores.

- Filo Lenia
  - Classe *Exokernel*
    - Ordem *Orbiformes*
      - Família *Orbidae* (O)
    - Ordem *Scutiformes*
      - Família *Scutidae* (S)
      - Família *Pterifera* (P)
      - Família *Helicidae* (H)
      - Família *Circidae* (C)
  - Classe *Mesokernel*
    - Ordem *Echiniformes*
      - Família *Echinidae* (E)
      - Família *Geminidae* (G)
      - Família *Ctenidae* (Ct)
      - Família *Uridae* (U)
  - Classe *Endokernel*
    - Ordem *Kroniformes*
      - Família *Kronidae* (K)
      - Família *Quadridae* (Q)
      - Família *Volvidae* (V)
    - Ordem *Radiiformes*
      - Família *Dentidae* (D)
      - Família *Radiidae* (R)
      - Família *Bullidae* (B)
      - Família *Lapillidae* (L)
      - Família *Folidae* (F)
    - Ordem *Amoebiformes*
      - Família *Amoebidae* (A)

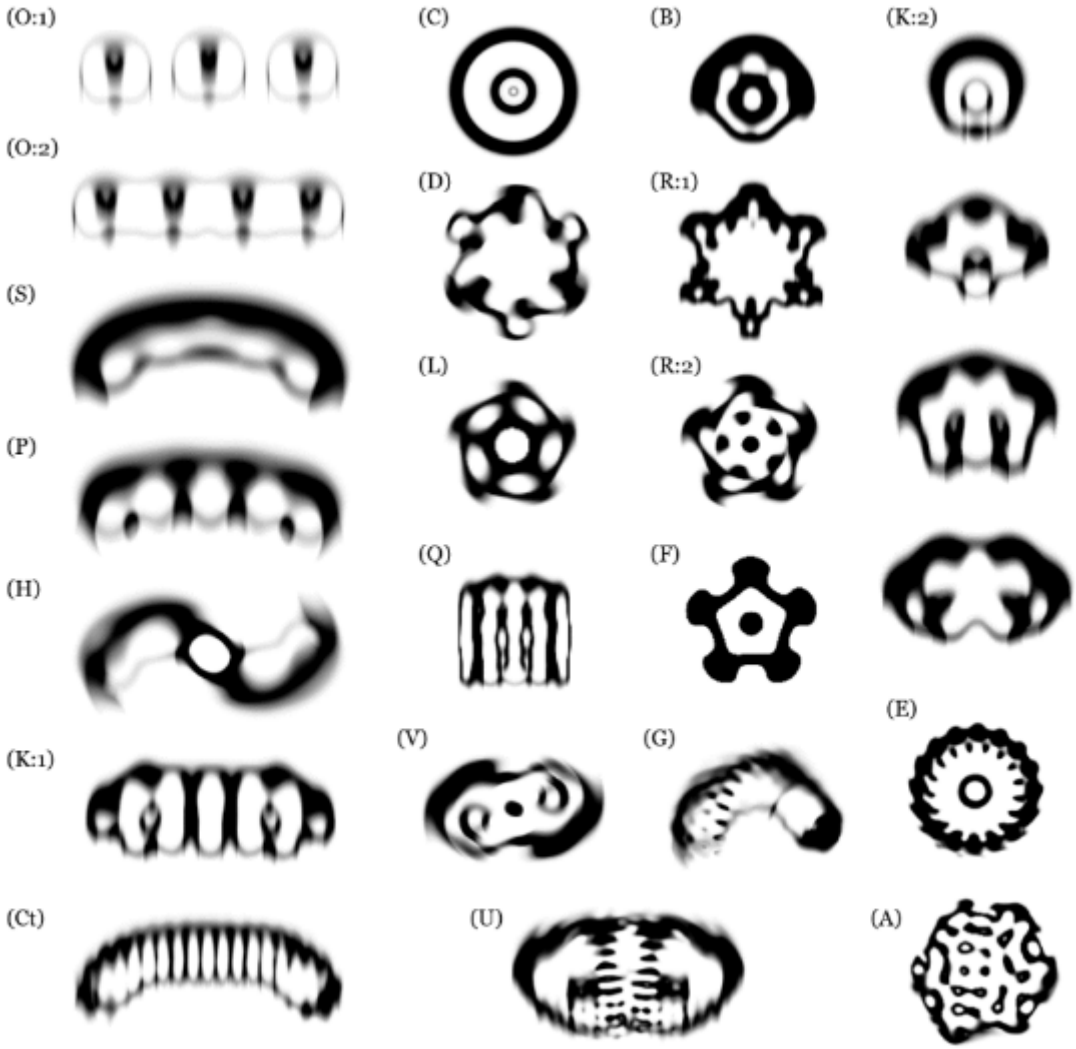


Figura 4.7: Biodiversidade em Lenia, com exemplos de espécies das 18 famílias de Lenia (fora de escala). **Coluna 1**: *Orbidae*, *Scutidae*, *Pterifera*, *Helicidae*, *Kronidae* e *Ctenidae*. **Coluna 2**: *Circidae*, *Dentidae*, *Lapillidae*, *Quadridae* e *Volvidae*. **Coluna 3**: *Bullidae*, *Radiidae*, *Folidae*, *Geminidae* e *Uridae*. **Coluna 4**: *Kronidae*, *Echinidae* e *Amoebidae* (CHAN, 2019).

4.3.3 Lenia e Vida Biológica

Em seu trabalho, Chan evidenciou as similaridades que Lenia e a vida biológica na Terra compartilham. Entre elas, tem-se primeiramente a simetria estrutural, que, em Lenia, se apresenta como simetria radial e simetria bilateral. Além disso, também compartilham relações parecidas entre simetria e locomoção, pois em Lenia espécimes com simetria radial "flutuam" ou se movem lentamente, enquanto aqueles com simetria bilateral se deslocam rapidamente e de forma linear.

Ademais, considerando o conjunto de parâmetros de Lenia como um ambiente e a sobrevivência de espécimes na rede como adaptação a ele, foi encontrado que inúmeras espécies em Lenia são capazes de adaptação e evolução, quando mudam sua forma e se estabilizam, tal como ocorre na Terra.

Por fim, se táxons são considerados altamente adaptativos quando possuem alta biodiversidade, distribuição ecológica ou complexidade, algumas famílias em Lenia podem ser consideradas altamente adaptativas, pois possuem estas características. Chan menciona, em especial, a família *Pterifera*.

Na figura a seguir, Chan elenca alguns espécimes de Lenia e suas semelhanças com estruturas conhecidas na Terra, desde organelas celulares a algas e animais extintos.

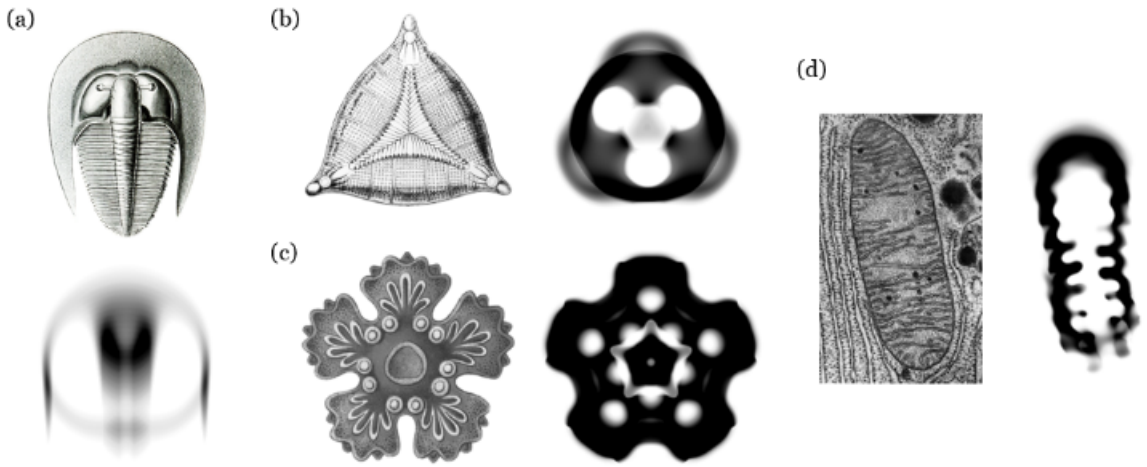


Figura 4.8: Comparações entre espécies encontradas em Lenia e vida biológica. Em **(a)**, o trilobita bilateral *Bohemoharpes ungula* e um *Orbium unicaudatus* Leniano; em **(b)**, diatomácea trímera *Triceratium moronense* e um *Trilapillium inversus* Leniano; em **(c)**, larva pentâmera da estrela do mar de espécies do gênero *Asterias* e um *Asterium inversus* Leniano; e em **(d)**, uma mitocôndria fracamente bilateral e um *Hydrogeminium natans* Leniano (CHAN, 2019).

Capítulo 5

Metodologia

Neste trabalho, os autômatos estudados por meio de simulações foram o Jogo da Vida, SmoothLife e Lenia. Todas as simulações foram escritas utilizando a linguagem de programação Go. Houve o mesmo algoritmo base para todos esses autômatos celulares, que será descrito a seguir.

1. É escrita uma função que cria um novo arquivo a cada geração, ou seja, sempre que ocorre um número definido de passos temporais. Este arquivo contém a posição exata de cada célula da rede e seu estado na geração em questão.

2. Em seguida, é definida outra função contendo a condição inicial, que possui fundamentação pseudoaleatória. A condição inicial é aplicada automaticamente ao executar as simulações.

3. A depender do funcionamento do autômato celular em questão, algumas funções utilizadas na evolução da rede são definidas externas à função principal.

4. É criada a função principal. Aqui, não apenas é determinada a evolução da rede, como é criado outro arquivo que grava a densidade populacional, isto é, as porcentagens de células em cada estado a cada geração. Então, cada célula é tratada como principal, em ordem de linha e coluna, e sua vizinhança é identificada. Assim, as regras são aplicadas e o estado da célula é atualizado. Quando todas as células da rede são verificadas, ocorre um passo temporal. Após certo número de passos temporais (definidos pelo pesquisador), passa-se uma geração, e a densidade populacional é zerada para acomodar os dados da próxima.

Apesar de seu fator de aleatoriedade, a condição inicial pode ser manipulada para atender à razão de distribuição desejada pelo pesquisador. Nas primeiras simulações de cada autômato celular estudado, fez-se com que os estados iniciais fossem equiprováveis (razão 50/50), a fim de replicar um ambiente inicialmente neutro e verificar suas ramificações. Depois, foram estudadas outras razões de distribuição e outros métodos de condição inicial, com o intuito de comparar as evoluções de cada simulação e determinar como a distribuição inicial afeta o progresso temporal da rede.

Ademais, considerando as células limítrofes, foram elaboradas condições de contorno em todas as simulações para que todas as células possuam vizinhança completa, independente de sua posição.

O primeiro arquivo, ou "arquivo zero", expõe a distribuição inicial, isto é, aplica a condição inicial à rede. Os arquivos seguintes a demonstram reagindo linearmente ao conjunto de regras do autômato. As dimensões da rede, bem como o número de gerações, variam com o modelo estudado e o propósito da simulação.

Com o arquivo de densidade populacional para cada simulação, é possível acompanhar as quantidades de células vivas e mortas conforme o tempo progride, e observar se há ou não estabilização numérica da população após determinado número de gerações. Estes dados também expõem quão drasticamente a rede pode se alterar da condição inicial à primeira geração.

Para criar e executar as simulações deste trabalho, foram utilizados os seguintes ambientes:

- *Hardware*: LG A410 (Windows 7);
- *Software*: Go versão 1.20, Visual Studio Code, Notepad++, Prompt de Comando e Gnuplot.

Os códigos escritos, bem como as redes expostas nos Resultados (em formato de vídeos), estão disponíveis na plataforma Github (BITTENCOURT, 2024). Além disso, para acesso simplificado, os códigos também foram adicionados ao Apêndice (A.1, A.2 e A.3) deste trabalho.

5.1 Jogo da Vida

5.1.1 Jogo da Vida de Conway (B3/S23)

Para a simulação principal do Jogo da Vida de Conway, isto é, com o conjunto de regras descrito no capítulo 1, utilizou-se uma rede de 150×150 *pixels*, totalizando 22.500 células, e 1.000 gerações. Além disso, como explicado anteriormente, sua condição inicial foi elaborada para a equiprobabilidade dos estados vivo e morto.

Para as outras distribuições iniciais exploradas, utilizou-se uma rede menor com o intuito de minimizar os tempos de execução das simulações, considerando que a rede reduzida seria suficiente para observação. As dimensões destas versões, portanto, são de 100×100 *pixels*, totalizando 10.000 células, e as 1.000 gerações se mantêm.

Ademais, quanto às distribuições em si, utilizou-se o mesmo sistema pseudoaleatório, agora sistematicamente manipulado para favorecer um dos dois estados. Também foi usada a notação de porcentagens separadas por /, sendo a porcentagem à esquerda correspondente ao estado vivo e à direita ao estado morto. As condições iniciais exploradas para o Jogo da Vida, portanto, foram 50/50 (distribuição equiprovável), 10/90, 25/75, 75/25 e 90/10.

5.1.2 Outros Conjuntos de Regras

A fim de explorar o impacto das regras utilizadas na evolução temporal do Jogo da Vida, também foram executadas simulações utilizando outros conjuntos de regras:

- B3/S245, onde células só se tornam vivas com exatamente três vizinhos vivos, e só sobrevivem com dois, quatro ou cinco;
- B1/S12, em que células só se tornam vivas com um vizinho vivo, e só sobrevivem com um ou dois;
- B0123478/S01234678, chamado AntiLife, cujas regras causam a inversão dos estados vivo e morto para os padrões encontrados no Jogo da Vida de Conway.

Para estes conjuntos de regras, foram novamente usadas redes de dimensões 150×150 *pixels*, mas 500 gerações para cada simulação. Para o caso B1/S12, foram utilizadas duas condições iniciais distintas para obter um resultado específico.

5.2 SmoothLife

Para o autômato SmoothLife, a rede teve a forma de 500×500 *pixels*, ou seja, 250.000 células, e 500 gerações devido ao maior tempo de execução afetado pelo tamanho da rede. Desta forma, foi de maior interesse observar diferentes evoluções da rede trazidas pela modificação arbitrária da condição inicial e dos parâmetros de SmoothLife.

A condição inicial similar ao Jogo da Vida, com a geração pseudoaleatória de estados em toda a rede, ocorreu em apenas uma simulação. No restante, fez-se o surgimento de regiões não nulas pela rede, ou seja, às células dentro de suas áreas foram atribuídos estados decimais e pseudoaleatórios entre 0 e 1. Células fora destas áreas permaneceram com estado 0. Estas áreas ocorreram em posições aleatórias, porém com dimensões de quadrados. Para as simulações preservando os parâmetros propostos por Rafler (exceto dt), foram gerados 30 quadrados com área de 50×50 *pixels*, ou 2500 células. Esta condição inicial foi inspirada por uma simulação existente criada por Chan (CHAN, 2015b).

Os parâmetros específicos de SmoothLife foram definidos diretamente no algoritmo do autômato. A primeira simulação, executada duas vezes em razão das duas condições iniciais descritas, preservou todos os parâmetros expostos por Rafler para a obtenção de um *smooth glider*: $r_a = 21$ (raio da vizinhança externa), $b_1 = 0,278$, $b_2 = 0,365$ (intervalos de nascimento), $d_1 = 0,267$, $d_2 = 0,445$ (intervalos de desaparecimento), $\alpha_n = 0,028$, $\alpha_m = 0,147$ (larguras dos passos temporais) e $dt = 1$. No entanto, para estudar o impacto do número de passos por geração (dt) no comportamento do autômato, foram testados vários outros valores de dt para o mesmo conjunto de parâmetros. Os outros valores utilizados foram: $dt = 0,01$, $dt = 0,1$, $dt = 10$ e $dt = 100$. Assim, estes foram analisados em duplas e posteriormente juntos à primeira simulação com o propósito de melhor averiguar o efeito de diferentes dt sobre uma mesma rede.

Então, a fim de descobrir o impacto de parâmetros distintos sobre a evolução temporal do SmoothLife, outros valores específicos também foram utilizados. Logo, foram executadas três simulações adicionais, cada uma com seu respectivo conjunto de parâmetros, e estas também foram comparadas entre si em uma tentativa de determinar a contribuição de cada parâmetro nos padrões encontrados.

- Simulação 1: $r_a = 31,8$, $b_1 = 0,092$, $b_2 = 0,098$, $d_1 = 0,256$, $d_2 = 0,607$, $\alpha_n = 0,015$, $\alpha_m = 0,340$ e $dt = 0,157$;
- Simulação 2: $r_a = 12$, $b_1 = 0,299$, $b_2 = 0,438$, $d_1 = 0,198$, $d_2 = 0,438$, $\alpha_n = 0,028$, $\alpha_m = 0,147$ e $dt = 0,1$;
- Simulação 3: $r_a = 12$, $b_1 = 0,362$, $b_2 = 0,470$, $d_1 = 0,221$, $d_2 = 0,540$, $\alpha_n = 0,028$, $\alpha_m = 0,147$ e $dt = 0,1$.

Para a coleta de dados acerca da densidade populacional, isto é, buscando sintetizar todos os estados possíveis de SmoothLife em duas curvas, considerou-se que ainda pertenceriam a duas categorias, "vivo" e "morto". Desta forma, estados com valor maior que $0,5$ foram atribuídos a "vivo", e estados menores que $0,5$ a "morto", e o algoritmo base de gravação das densidades populacionais se manteve. Esta categorização ocorreu com a finalidade exclusiva de auxiliar a análise das populações da rede.

5.3 Lenia

Para o autômato Lenia, utilizou-se redes de dimensões 150×150 *pixels*, totalizando 22.500 células.

Foram feitas simulações com diferentes parâmetros e funções K_C e G, a fim de obter alguns espécimes. Estas alterações foram realizadas com o intuito de reproduzir indivíduos de diferentes famílias. Logo, o único método para obtenção de espécimes utilizado foi o de simulações automáticas.

A seguir, tem-se as famílias e gêneros aos quais pertencem e as funções e os parâmetros utilizados para reproduzi-los. Para estas simulações específicas, usou-se 200 gerações e a condição inicial de regiões não nulas, gerando dez quadrados de dimensões 18×18 *pixels* e todos os estados equiprováveis. Além disso, foi utilizado $dt = 0,1$ para todas as simulações.

- *Orbidae* (*Orbium unicaudatus*):
  - $R = 13$, $\mu = 0,15$, $\sigma = 0,017$ e $\alpha = 4$;
  - $K_C(r) = \exp\left(\alpha - \frac{\alpha}{4r(1-r)}\right)$;
  - $G(u;\mu,\sigma) = 2\exp\left(-\frac{(u-\mu)^2}{2\sigma^2}\right) - 1$;
- *Floridae* (*Pentafolium lithos*):
  - $R = 18$, $\mu = 0,26$, $\sigma = 0,038$, $\alpha = 4$, $\beta_1 = 1$ e $\beta_2 = 1$;
  - $K_C(r) = \begin{cases} \beta_1 \times (4 \times 2R(1-2R))^\alpha & \text{se } R \leq 0,5 \\ \beta_2 \times [4 \times (2R-1)(1-(2R-1))]^\alpha & \text{se } 0,5 < R \leq 1 \end{cases}$;
  - $G(u;\mu,\sigma) = \begin{cases} 2\times \left(1 - \frac{(u-\mu)^2}{(3\sigma)^2}\right)^\alpha - 1 & \text{se } u - \mu \leq 3\sigma \\ -1 & \text{outros casos} \end{cases}$;
- *Circidae* (*Tricircium inversus*):
  - $R = 18$, $\mu = 0,25$, $\sigma = 0,003$, $\alpha = 4$, $\beta_1 = 1$ e $\beta_2 = 0,33$;
  - $K_C(r) = \begin{cases} \beta_1 \times (4 \times 2R(1-2R))^\alpha & \text{se } R \leq 0,5 \\ \beta_2 \times [4 \times (2R-1)(1-(2R-1))]^\alpha & \text{se } 0,5 < R \leq 1 \end{cases}$;
  - $G(u;\mu,\sigma) = \begin{cases} 2\times \left(1 - \frac{(u-\mu)^2}{(3\sigma)^2}\right)^\alpha - 1 & \text{se } u - \mu \leq 3\sigma \\ -1 & \text{outros casos} \end{cases}$;
- *Volvidae* (*Tetravolvium*):
  - $R = 18$, $\mu = 0,24$, $\sigma = 0,0281$, $\alpha = 4$, $\beta_1 = 1$ e $\beta_2 = 0,33$;
  - $K_C(r) = \begin{cases} \beta_1 \times (4 \times 2R(1-2R))^\alpha & \text{se } R \leq 0,5 \\ \beta_2 \times [4 \times (2R-1)(1-(2R-1))]^\alpha & \text{se } 0,5 < R \leq 1 \end{cases}$;
  - $G(u;\mu,\sigma) = \begin{cases} 2\times \left(1 - \frac{(u-\mu)^2}{(3\sigma)^2}\right)^\alpha - 1 & \text{se } u - \mu \leq 3\sigma \\ -1 & \text{outros casos} \end{cases}$.

Os táxons, as funções e os parâmetros acima descritos foram extraídos diretamente da simulação de Chan (CHAN, 2015b).

Em seguida, buscando o surgimento espontâneo de um *Orbium unicaudatus*, foi feita mais uma simulação para a família *Orbidae*, mas com uma condição inicial menos intensa. Assim, a geração de dez quadrados se manteve, porém com dimensões de 15×15 *pixels* e a distribuição de estados modificada para favorecer estados menores que $0,5$. Além disso, passaram-se 50 gerações. Como o objetivo principal desta simulação é obter um *Orbium*, a densidade populacional da rede não foi analisada.

Para as densidades populacionais nas demais simulações, procedeu-se da mesma forma que para SmoothLife; considerou-se estados maiores que $0,5$ o estado vivo e, menores, o estado morto. Novamente, a função única da separação em duas categorias é sintetizar todos os estados em duas curvas e facilitar a análise da população.

Capítulo 6

Resultados

6.1 Jogo da Vida de Conway

6.1.1 Distribuição Equiprovável (50/50)

Como mencionado previamente, neste trabalho, esta simulação é considerada a principal para o Jogo da Vida. O primeiro arquivo gerado pela simulação expõe a distribuição inicial da rede. Uma vez inserida a condição inicial, esta foi sujeita à evolução temporal moldada pelas regras originais. A partir da geração inicial e desta progressão, foram obtidos os *snapshots* seguintes, obtidos por meio dos arquivos de dados das gerações.

(a) Condição inicial (b) 100 gerações (c) 500 gerações (d) 1000 gerações

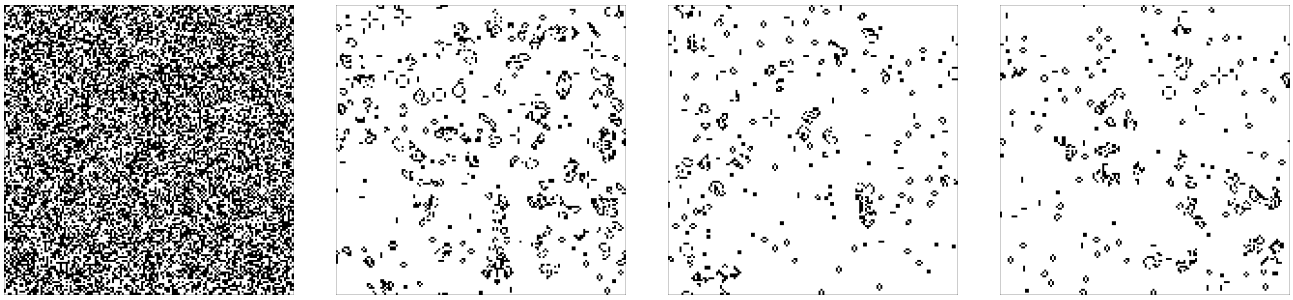


Figura 6.1: A evolução do Jogo da Vida de Conway com distribuição inicial equiprovável. Autoria própria.

Como esperado, observa-se que a população ativa rapidamente reduziu-se devido ao desaparecimento instantâneo de células vivas isoladas ou em regiões superpopuladas. Por meio do conjunto de regras de Conway, foram encontrados vários dos padrões ditos essenciais discutidos no capítulo 1, como naturezas-mortas, osciladores, alternadores e *gliders*. Apesar disto, várias formas específicas não foram encontradas por conta das circunstâncias necessárias para o surgimento destas, que raramente ocorrem em distribuições aleatórias.

Ao aproximar-se do marco de mil gerações, o sistema não alcançou estabilidade completa, embora algumas regiões se mostrem estáticas ou periódicas. Nas áreas em que ainda ocorrem transformações, nota-se principalmente alternadores e células instáveis, desaparecendo ou criando novas células vivas nestes processos. Como ilustrado acima, as extremidades são majoritariamente constituídas de naturezas-mortas e *blinkers*, embora esses padrões possam ser perturbados graças às células ativas próximas.

A evolução tendendo à estabilidade é um fim comum encontrado no Jogo da Vida, como explicado no capítulo 1, e especialmente no algoritmo utilizado devido à distribuição pseudoaleatória e à evolução automática. Considerando as regras de Conway e a metodologia em uso, as

interações nesta são frequentemente destrutivas por causa da larga ocorrência de isolamento e superpopulação. De fato, interações construtivas geralmente requerem posicionamentos específicos, que se tornam improváveis em uma simulação automática e inicialmente aleatória. Neste sentido, os resultados obtidos para este autômato celular são semelhantes àqueles previstos em teoria.

Por fim, com base nos dados da densidade populacional da rede conforme a progressão temporal, foi confeccionado um gráfico ilustrando as porcentagens referentes aos dois estados e sua variação com o passar das gerações.

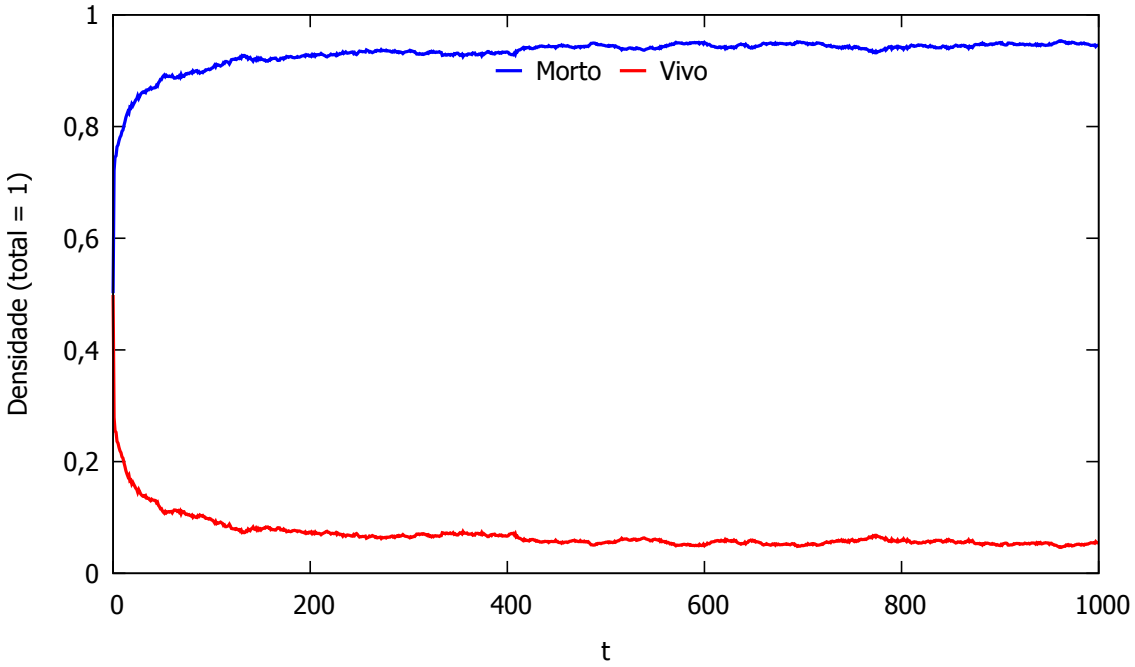


Figura 6.2: A evolução temporal da densidade populacional para o Jogo da Vida de Conway com razão de probabilidade 50/50. Autoria própria.

Ao analisar as gerações iniciais, observa-se um declínio na população ativa, que causa um aumento na densidade populacional de células mortas. Este fato demonstra numericamente o desaparecimento em massa observado na rede, referente a células vivas isoladas ou em locais superpovoados. Ademais, revisando os dados brutos e as curvas esboçadas, a disparidade numérica entre os estados vivo e morto é crescente. Embora a simulação inicie-se aproximadamente equiprovável, sendo 49,8% da rede composta de células vivas e 50,2% de células mortas, ao final, a porcentagem de células ativas torna-se 5,5%, enquanto as inativas ocupam 94,5% da rede.

Evidencia-se ainda que, apesar do desequilíbrio populacional progressivo, ocorre uma tendência à estabilização numérica após a primeira metade das gerações da simulação. Isto é, após certo número de gerações, tornam-se raras e mínimas as alterações populacionais na rede. Este fato é visto visualmente por meio da figura 6.1.

Logo, apesar da condição inicial equiprovável, o equilíbrio numérico entre os estados vivo e morto rapidamente desaparece, e células ativas se tornam significativamente mais raras conforme o Jogo da Vida progride. Por fim, constatou-se a quase estabilização numérica da população da rede em suas gerações finais, mencionada anteriormente e exposta visualmente.

6.1.2 Outras Condições Iniciais

As evoluções temporais de cada uma das condições iniciais citadas no capítulo 4 estão dispostas a seguir.

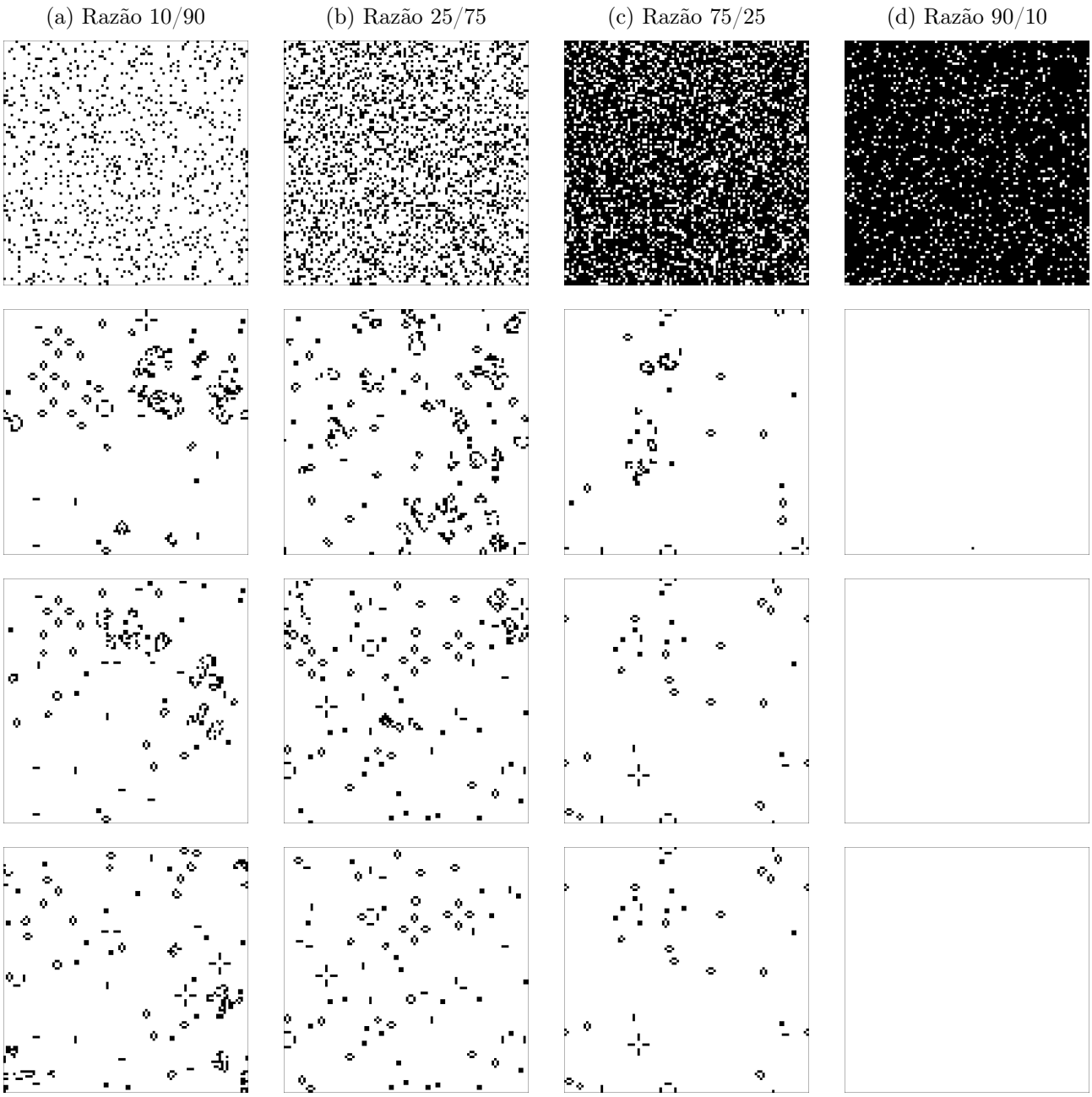


Figura 6.3: A progressão de 1.000 gerações do Jogo da Vida de Conway com outras distribuições iniciais. Primeira linha: condições iniciais. Segunda linha: gerações 100 (exceto razão 90/10, geração 1). Terceira linha: gerações 500. Última linha: gerações 1.000. Autoria própria.

A partir dos *snapshots* obtidos para estas condições iniciais, é possível inferir uma correlação entre a estabilidade visual e numérica da rede e as razões de cada distribuição. Para redes cujas simulações iniciam-se com mais células inativas do que ativas, observa-se grande quantidade de padrões estáveis como naturezas-mortas e *blinkers*, mas também interações ativas. Inversamente, as simulações com alta quantidade de células vivas possuem menos padrões formados na rede, sendo que a rede com maior razão de células ativas evolui para o desaparecimento completo e praticamente instantâneo destas.

Desta forma, redes com distribuições menores de células ativas, embora numericamente escassas, possibilitam o nascimento e a manutenção de células vivas mais facilmente. Este fato decorre das regras originais de Conway, que favorecem vizinhanças moderadas. Enquanto raramente encontra-se superpopulação em simulações com tais condições iniciais, células vivas isoladas são relativamente comuns e, como esperado, desaparecem imediatamente. Assim, estas distribuições esparsas favorecem a evolução para redes com três tipos de regiões: vazias, completamente estáveis e com interações ativas.

Por outro lado, observa-se que a quantidade maior de células ativas não necessariamente resulta em uma rede com mais transformações ou mais padrões. De fato, a partir da figura 6.3, vê-se a situação oposta. Esta ocorrência é consequência direta das regras de Conway, especialmente no que tange à quantidade de vizinhos ativos necessária para o nascimento ou a manutenção do estado vivo. Logicamente, a superpopulação da rede resulta no desaparecimento maciço de grande parte das células vivas. Assim, a estabilidade numérica é rapidamente atingida, seja por meio de uma rede vazia ou composta apenas de padrões estáveis. Na razão inicial 90/10, por exemplo, apenas uma célula sobrevive à primeira geração, e assim a estabilidade é atingida a partir da segunda.

Finalmente, foi encontrado que três das quatro condições iniciais evoluíram para redes inteiramente estáveis, seja por meio de uma rede completamente inativa (razão 90/10) ou repleta de naturezas-mortas e alternadores (razões 25/75 e 75/25). Embora a rede com razão inicial 10/90 não tenha alcançado estabilidade ao final das gerações, nota-se visualmente uma aproximação para esta. Logo, não é possível descartar a possibilidade de todas as distribuições iniciais estudadas convergirem para a estabilidade populacional.

Em seguida, foi confeccionado um gráfico reunindo as densidades populacionais para as condições iniciais aqui discutidas. Portanto, tem-se que cada cor representa uma razão inicial, enquanto os diferentes traços de curva representam os dois estados. As curvas tracejadas referem-se ao estado morto na rede, enquanto as curvas preenchidas representam o estado vivo.

O gráfico acima confirma a tendência à estabilidade numérica exposta visualmente por meio da figura 6.3 e encontrada para a simulação de razão inicial 50/50. Analisando as curvas obtidas, nota-se que as simulações com razões iniciais menores de células ativas mantêm as maiores razões durante a simulação, enquanto aquelas com razões iniciais maiores possuem a menor porcentagem de células vivas com o passar do tempo. Isto ocorre, como explicado anteriormente, devido ao desaparecimento em massa encontrado em grandes populações de células vivas. Assim, como observado, redes inicialmente menos povoadas são favorecidas na manutenção de suas células ativas, bem como no surgimento de novas células vivas.

Além disso, observando as formas das curvas esboçadas para cada razão inicial, observa-se que razões mais extremas possuem também curvas mais retangulares. Para a distribuição 10/90, por exemplo, sua fase inicial altamente desequilibrada transforma-se em uma evolução quase linear, com as porcentagens populacionais pouco alterando com o passar do tempo. Isto se deve principalmente às regiões estáveis obtidas em poucas gerações, enquanto as oscilações pouco significativas representam as breves interações. Por outro lado, para a distribuição 90/10, a sobrevivência de apenas uma célula e o consequente desaparecimento desta transformam as curvas de densidade populacional em retas posicionadas nos valores mínimo e máximo do gráfico, representando o estado morto preenchendo a rede por completo.

Finalmente, a convergência para estabilidade observada em três das quatro simulações é confirmada numericamente. As densidades populacionais para as distribuições 25/75, 75/25 e 90/10 transformam-se em retas em algum ponto das gerações. Logicamente, com as rápidas transformações ocorridas em redes mais povoadas, a simulação de razão 90/10 é a que atinge estabilidade mais rapidamente, seguida pela razão 75/25.

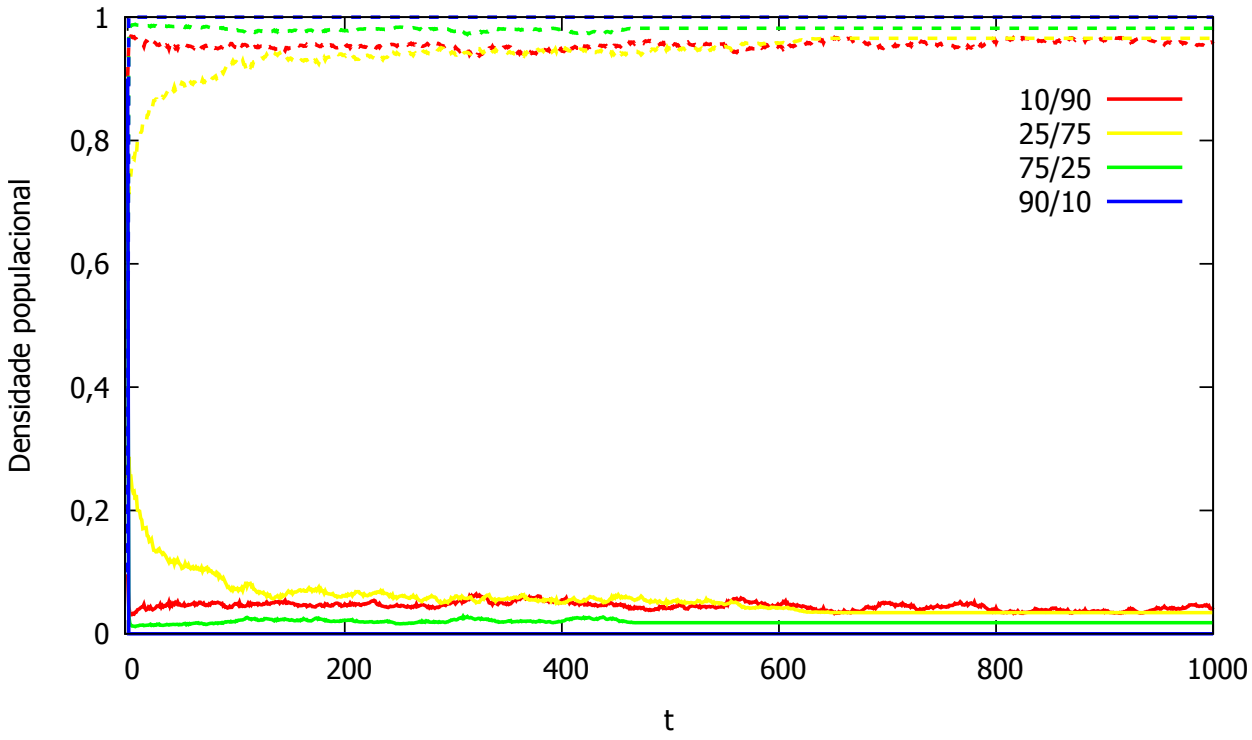


Figura 6.4: A evolução temporal da densidade populacional para o Jogo da Vida de Conway para as diferentes razões iniciais. Autoria própria.

6.1.3 Outros Conjuntos de Regras

B3/S245

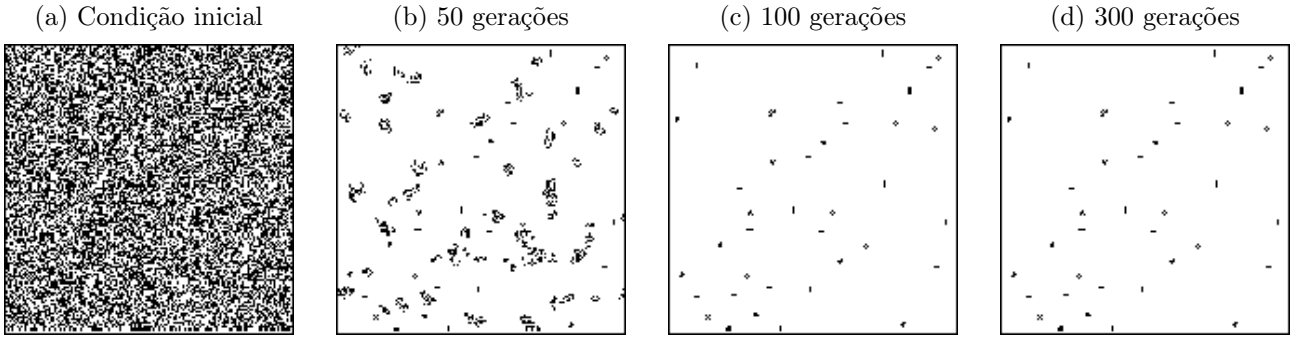


Figura 6.5: A evolução do Jogo da Vida de Conway para as regras B3/S245. Autoria própria.

Para este conjunto de regras, nota-se que a rede rapidamente se esvazia, mais drasticamente que para o Jogo da Vida de Conway. Isto provavelmente se deve ao fato da maioria das células vivas possuir menos de quatro ou mais de cinco vizinhos vivos. Rapidamente, restaram apenas algumas regiões com interações ativas entre células, com grande parte da rede estabilizando como naturezas-mortas ou alternadores.

Então, a partir de ao menos 100 gerações, a rede se estabilizou completamente, sem áreas ativas de transformação ou formação de novos padrões. Os padrões restantes se resumem a apenas naturezas-mortas ou alternadores, estes de período dois. Por conta deste último fato, é plausível que seus movimentos apenas possam ser observados em gerações ímpares para esta simulação, fazendo com que pareçam estáticos nos *snapshots* acima.

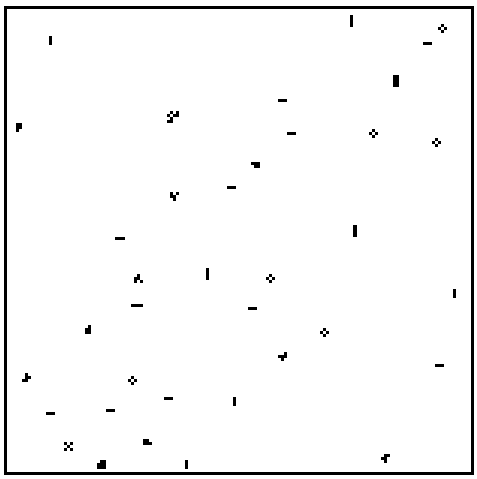


Figura 6.6: A geração 500 para as regras B3/S245. Autoria própria.

Finalmente, nota-se que os padrões formados de naturezas-mortas e alternadores são, em geral, bastante distintos dos encontrados no Jogo da Vida de Conway. Dois destes padrões na rede, por exemplo, têm a forma de um *glider* tradicional, mas não se deslocam. Além disso, como esperado, nota-se que grande parte dos padrões restantes têm algum nível de simetria.

O gráfico de densidade populacional para esta variação do Jogo da Vida está disposto a seguir.

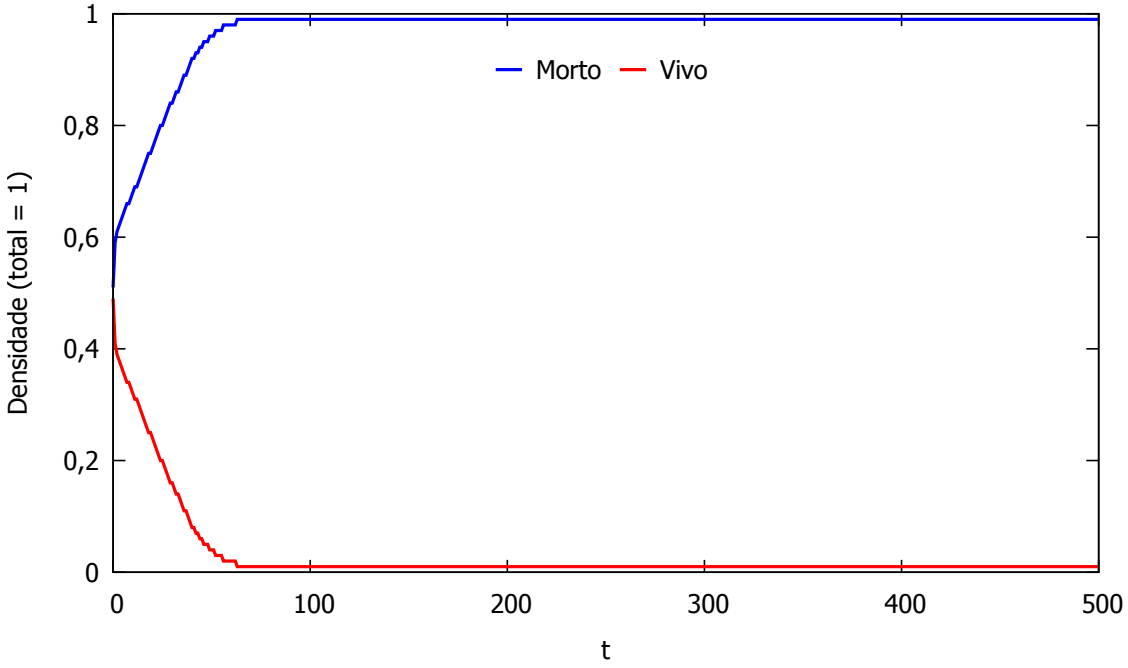


Figura 6.7: A evolução temporal da densidade populacional para o Jogo da Vida com regras B3/S245. Autoria própria.

Analisando a figura acima, nota-se ainda que as curvas geradas para este conjunto de regras não diferem significativamente das curvas encontradas para o Jogo da Vida de Conway com distribuição equiprovável. Um contraste, porém, está no quão rapidamente as curvas se transformam em retas: para as regras B3/S23 não apenas não houve estabilização completa da rede, como a variação da densidade populacional de cada estado ocorreu de forma mais gradual.

AntiLife (B0123478/S01234678)

(a) Condição inicial (b) 50 gerações (c) 100 gerações (d) 300 gerações

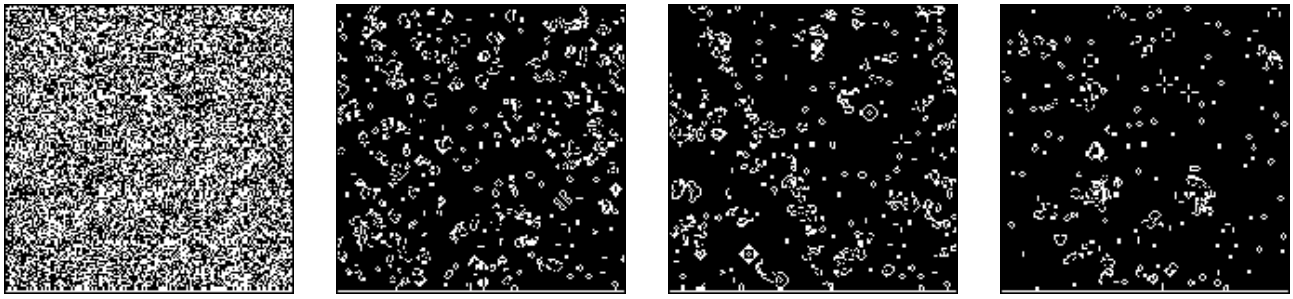


Figura 6.8: A evolução do Jogo da Vida para as regras B0123478/S01234678. Autoria própria.

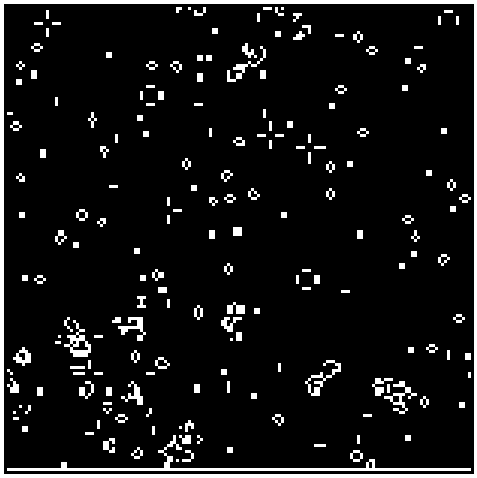


Figura 6.9: A geração 500 para o Jogo da Vida com regras B0123478/S01234678. Autoria própria.

Para este conjunto de regras, observou-se que, de fato, ocorreu a inversão de estados para o Jogo da Vida de Conway: as células mortas passaram a seguir as regras de Conway. Assim, células vivas preencheram a maior parte da rede, sendo as células mortas as reagentes para esta simulação.

Os padrões encontrados são idênticos aos originais; tem-se naturezas-mortas e alternadores com a mesma formação das regras B3/S23, porém substituindo células vivas por mortas. Assim, esta simulação cumpriu a inversão completa do Jogo da Vida de Conway.

Em seguida, foi confeccionado o gráfico da densidade populacional para este conjunto de regras:

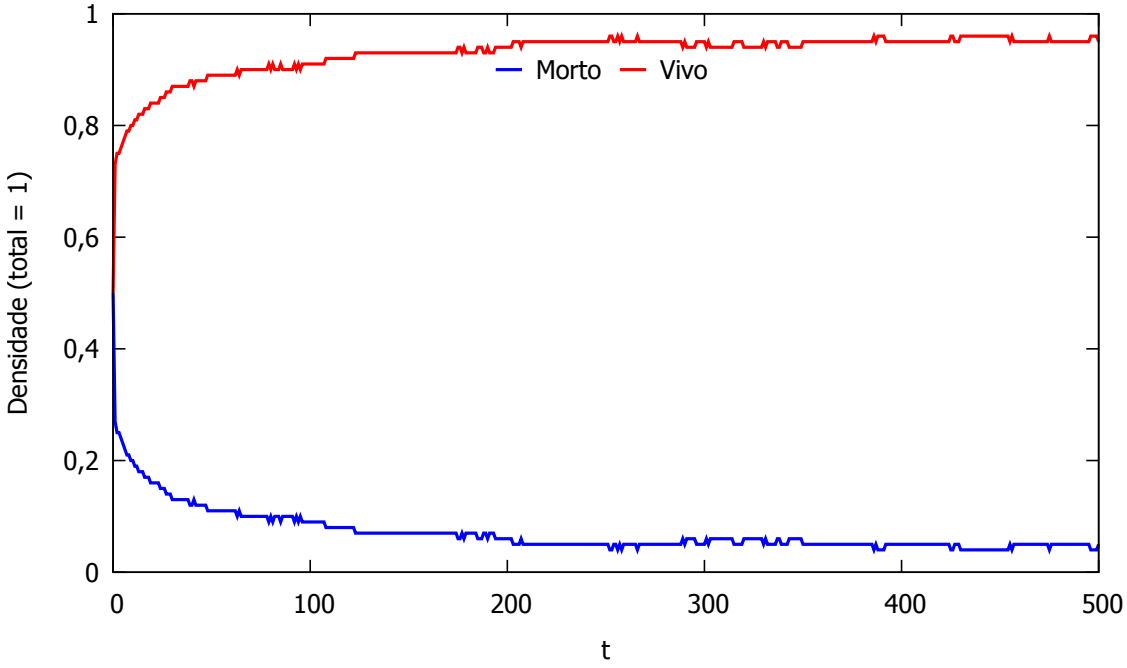


Figura 6.10: A evolução temporal da densidade populacional para o Jogo da Vida com regras B0123478/S01234678. Autoria própria.

O gráfico acima confirma a completa inversão do Jogo da Vida de Conway, invertendo também as curvas. Como observado por meio dos *snapshots* da rede, confirma-se que a população de células vivas cresceu enquanto a de células mortas reduziu-se, um resultado completamente oposto aos vistos em outros Jogos da Vida. Além disso, vê-se que o conjunto de regras B0123478/S01234678 teve vários períodos de população constante, representados pelos trechos retos em cada curva. Porém, estes foram breves e interrompidos por picos e vales. Estes períodos podem retratar, por exemplo, o deslocamento de *gliders* pela rede, causando variações na população ao colidir com outros espécimes.

Finalmente, como no Jogo da Vida de Conway com condição inicial equiprovável, a rede não se estabilizou nas gerações corridas.

B1/S12 (Razão 50/50)

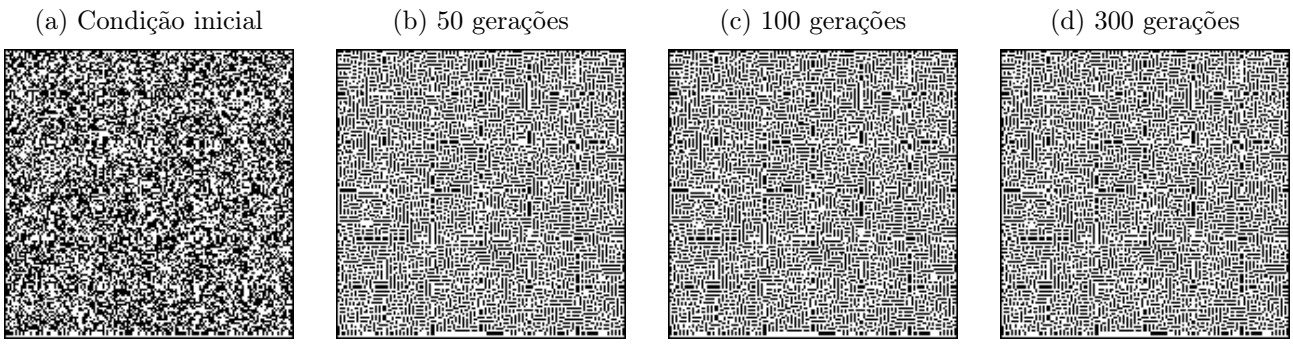


Figura 6.11: A evolução do Jogo da Vida com regras B1/S12 e distribuição inicial equiprovável. Autoria própria.

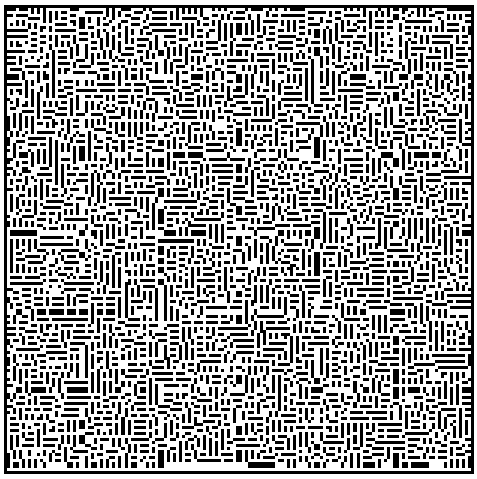


Figura 6.12: A geração 500 para o Jogo da Vida com as regras B1/S12 e distribuição inicial equiprovável. Autoria própria.

Este conjunto de regras é conhecido por gerar quatro aproximações do triângulo de Sierpinski em condição inicial com apenas uma célula viva. Porém, com condição inicial de distribuição de estados equiprovável para todas as células da rede, vê-se um comportamento bastante diferente, ilustrado nos *snapshots* acima.

Percebe-se, primeiramente, que a simulação se estabilizou rapidamente, antes mesmo de passar 50 gerações. Além disso, a forma final é bastante distinta se comparada ao Jogo da Vida de Conway, formando principalmente retas, quadrados e figuras retangulares. Em algumas regiões, aglomerados retangulares de apenas células vivas sobrevivem, enquanto em outras há pequenos vácuos que não são preenchidos. A maior parte dos padrões é composta de poucas células, e em geral se mantém uma distância entre esses padrões, impedindo interações significativas entre eles.

Apesar desse fato, porém, foram encontradas oscilações periódicas na rede, na forma de padrões vivos pequenos surgindo em vazios e prontamente desaparecendo. Analisando gerações seguidas, percebe-se que estes osciladores possuem período três.

A seguir, tem-se o gráfico de densidade populacional para esta simulação.

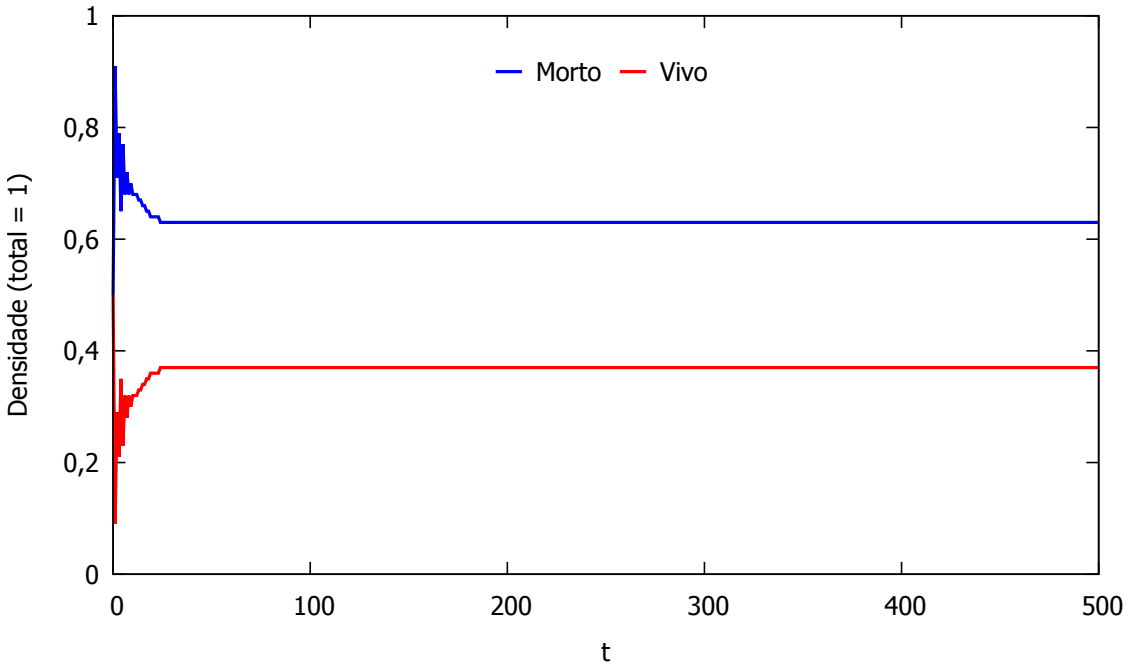


Figura 6.13: A evolução temporal da densidade populacional para o Jogo da Vida com regra B1/S12 e razão de probabilidade 50/50. Autoria própria.

Para as primeiras gerações, observa-se flutuações de densidade populacional consideráveis, com picos e quedas bastante próximos. No entanto, estas oscilações seguem uma tendência de aumento da população viva, até que ocorre a completa estabilização da rede e as populações se tornam constantes. Estes dados corroboram com as representações visuais da rede, em que o equilíbrio foi atingido em poucas gerações, mesmo com a ação de osciladores de período três.

B1/S12 (Célula Central)

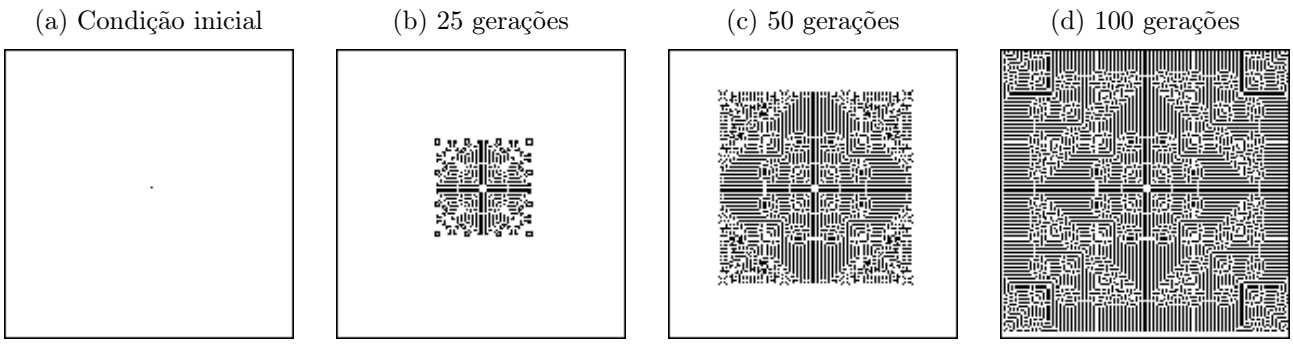


Figura 6.14: A evolução do Jogo da Vida com regras B1/S12 e condição inicial de apenas uma célula viva. Autoria própria.

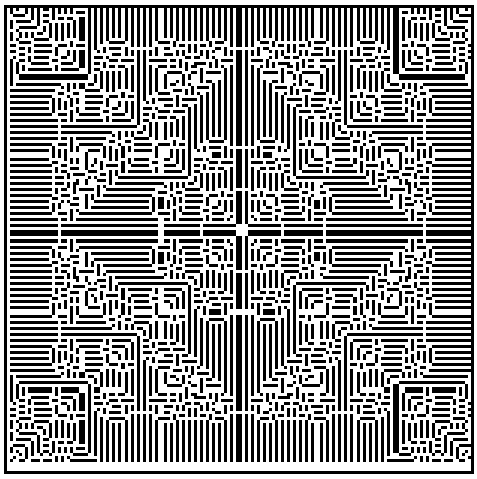


Figura 6.15: A geração 500 para o Jogo da Vida com as regras B1/S12 e condição inicial de apenas uma célula viva. Autoria própria.

Para visualizar a evolução da rede com o possível surgimento de aproximações do triângulo de Sierpinski, utilizou-se então a condição inicial específica de que apenas a célula central se encontra viva. Com apenas esta célula, nota-se o crescimento ilimitado de um padrão muito maior, iniciando ao redor da célula viva e eventualmente tomando toda a rede.

Esta construção, embora pareça crescer indefinidamente, atinge equilíbrio entre células vivas e mortas rapidamente, pois estas surgem e desaparecem em uma taxa constante através de uma alternância observada para o padrão de triângulos, que ocorre enquanto este se expande. Após ocupar todo o espaço disponível da rede, esta estrutura mantém a alternância de algumas células, sendo composta inteiramente de naturezas-mortas e alternadores.

Portanto, para este conjunto de regras com condição inicial de uma célula viva e 500 gerações, encontram-se quatro aproximações do triângulo de Sierpinski, como sugerido por outros pesquisadores.

Em seguida, foi esboçado o gráfico de densidades populacionais para esta simulação.

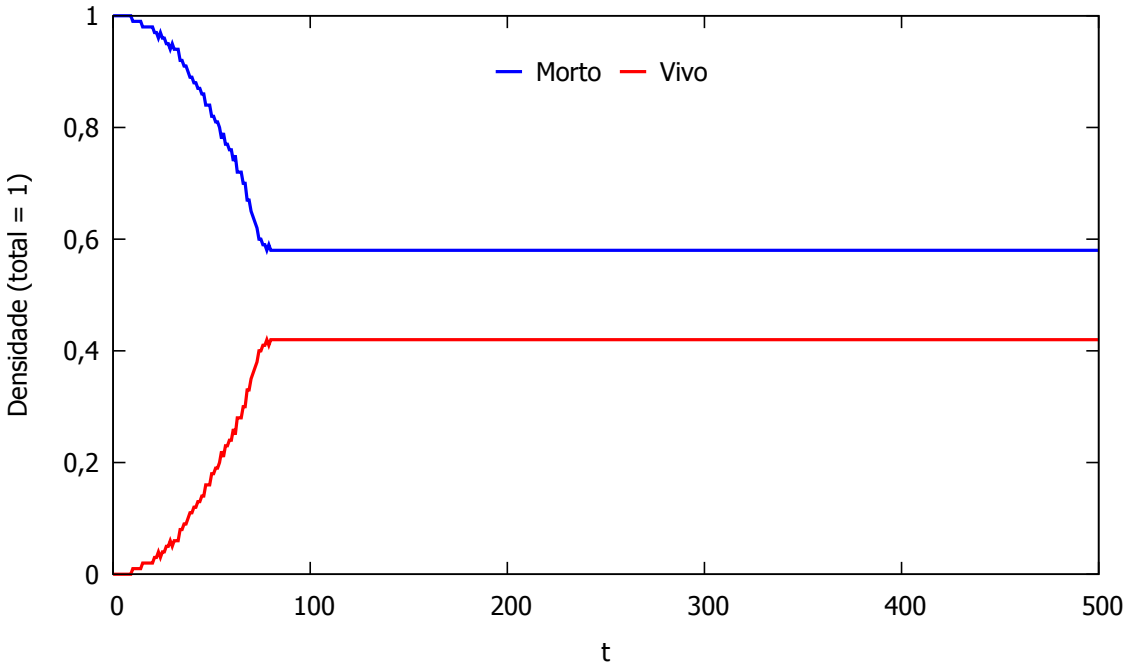


Figura 6.16: A evolução temporal da densidade populacional para o Jogo da Vida de Conway com razão de probabilidade 50/50. Autoria própria.

Como discutido anteriormente, o gráfico evidencia o aumento exponencial da população viva na rede, causada pela expansão da única célula viva no padrão observado. Porém, este aumento é limitado pelo próprio padrão, e a rede alcança densidades populacionais constantes antes de 100 gerações. A partir deste número, embora o padrão de triângulos ainda cresça, não ocorre mudança na porcentagem de população viva na rede. O mesmo fenômeno é observado quando o padrão ocupa todo o espaço da rede e apenas alternadores promovem mudança visual na rede.

6.2 SmoothLife

6.2.1 Condições Iniciais

As duas condições iniciais estão dispostas a seguir. As cores representam o estado de cada célula e seguem um gradiente de > 0 (violeta) a 1 (vermelho), com células nulas brancas, como representado na figura 6.17.

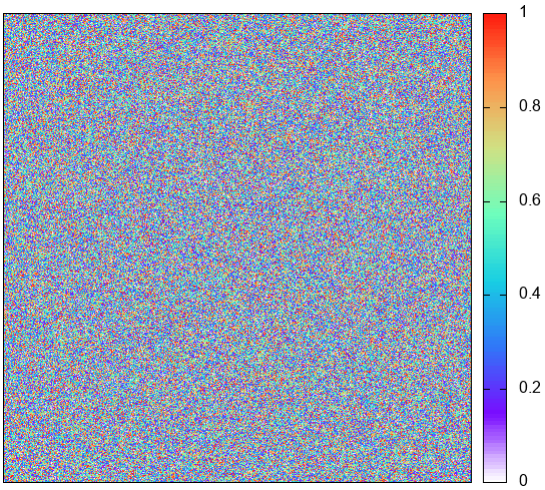


Figura 6.17: A condição inicial de rede inteira pseudoaleatória. Autoria própria.

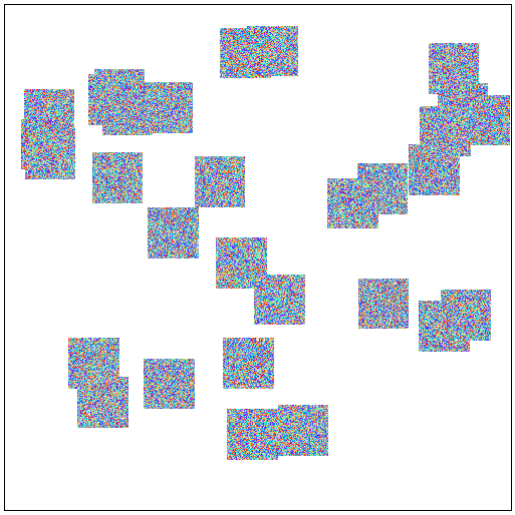


Figura 6.18: A condição inicial de quadrados não nulos. Autoria própria.

6.2.2 Parâmetros Originais

Discreto (dt = 1)

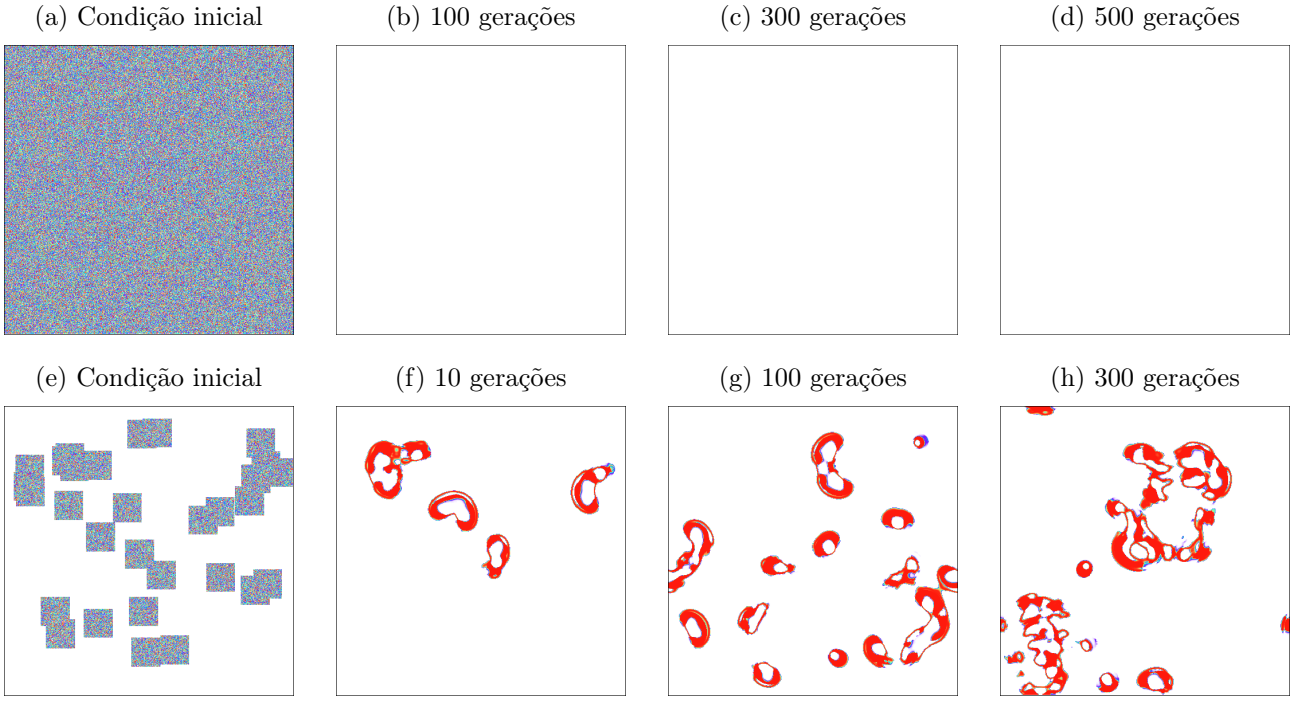


Figura 6.19: A evolução temporal das diferentes condições iniciais para $dt = 1$. Autoria própria.

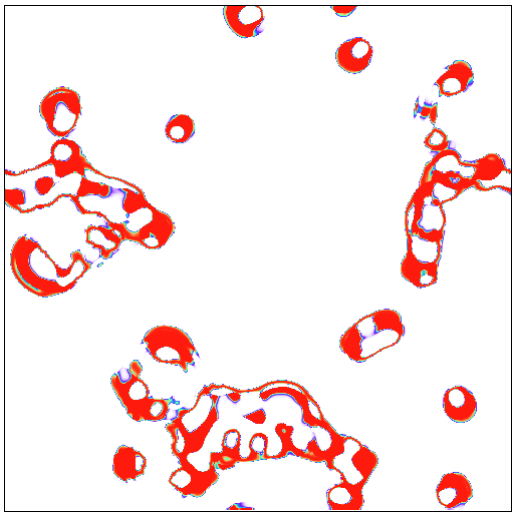


Figura 6.20: A geração final para a condição inicial de regiões não nulas e $dt = 1$. Autoria própria.

Para as duas simulações desta seção, nota-se resultados discrepantes para os mesmos parâmetros. Assim, é evidenciada empiricamente a relevância da condição inicial na progressão temporal da rede.

Para a condição inicial de rede inteiramente pseudoaleatória, observou-se o gradual desaparecimento de todas as células ativas da rede. Este fato provavelmente se deve às funções propostas por Rafler e seus funcionamentos em uma rede repleta de células ativas.

De outro modo, para a condição inicial de quadrados não-nulos, nota-se o surgimento pontual de "espécimes" a partir destas regiões. e pouco se deslocam com o passar do tempo. Assim, em poucas gerações foi obtido o *smooth glider* observado por Rafler, que se desloca em quaisquer direções. As interações entre eles dão surgimento a formas mais extensas e voláteis, que frequentemente criam mais *gliders* e desaparecem. Além disso, como se pode ver nas figuras 6.2.2 e 6.20, o *smooth glider* para este valor de dt tem vida útil curta, deslocando-se e reduzindo-se a uma pequena esfera até desaparecer.

Desta forma, foi cumprido o objetivo principal de encontrar o *smooth glider* por meio das simulações de SmoothLife com $dt = 1$. Considerando então os parâmetros originais propostos por Rafler, não foram encontrados outros padrões, apenas transformações rápidas causadas por interações entre *smooth gliders*. Outrossim, percebe-se que estas formas são compostas predominantemente de células com estado 1, que é o maior valor possível, e que células com estados menores costumam "contornar" a silhueta do *smooth glider*, acompanhando as células de estados maiores indefinidamente.

Em seguida, foi esboçado o gráfico combinando as densidades populacionais das duas condições iniciais estudadas para o SmoothLife.

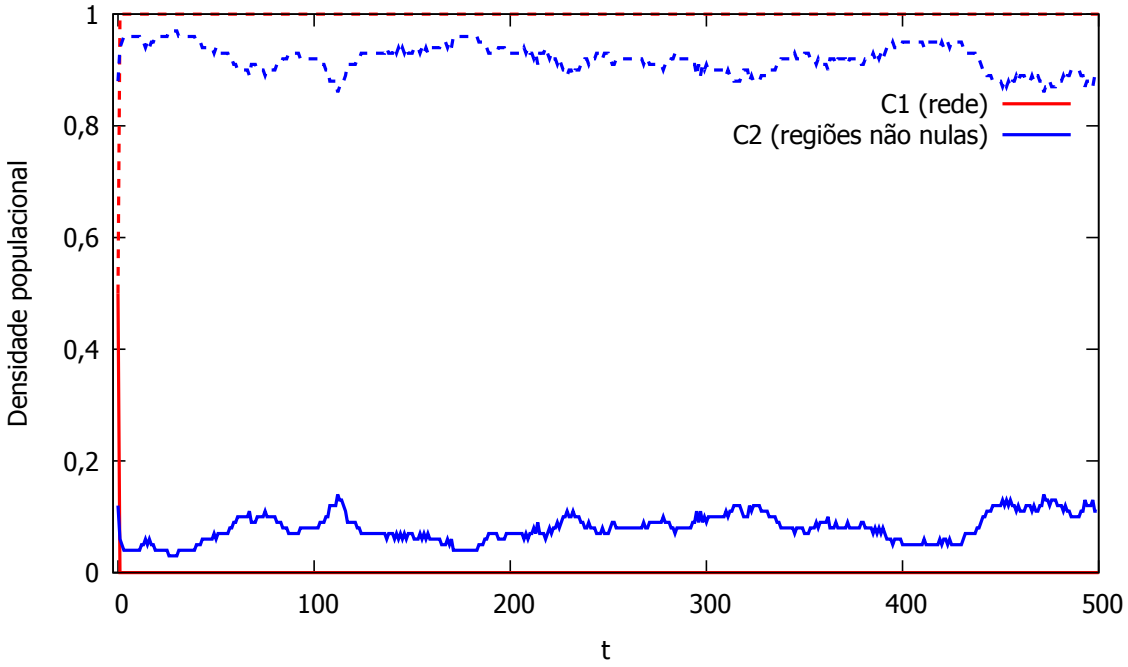


Figura 6.21: As evoluções temporais das densidades populacionais para cada condição inicial em SmoothLife discreto ($dt = 1$). Autoria própria.

A curva acima para a condição inicial de rede pseudoaleatória confirma o rápido declínio de toda a população ativa, causando a predominância do estado 0, que equivale à inatividade completa da rede.

Para a condição inicial de regiões não nulas, porém, nota-se instabilidade durante toda a simulação, sem clara convergência para um valor específico. Então, é possível inferir que este autômato com estes parâmetros não possui um fim previsto, sendo este fato mais um contraste

ao Jogo da Vida de Conway.

Por fim, há períodos breves de densidade constante entre os picos e vales apresentados nas curvas. A primeira situação pode ser explicada pelo surgimento e pelas interações entre *smooth gliders* durante toda a simulação. Enquanto estes espécimes desaparecem em poucas gerações, são facilmente formados por quaisquer outros padrões, inclusive em colisões entre os próprios *gliders*, como em um tipo de autorreplicação. Os períodos breves de população constante se devem ao deslocamento livre destes pela rede, sem desaparecer ou interagir com o meio. Evidentemente, tais gerações são interrompidas por interações entre estes *gliders*, que eventualmente se encontram em algum ponto da rede.

Discreto (dt = 10 e dt = 100)

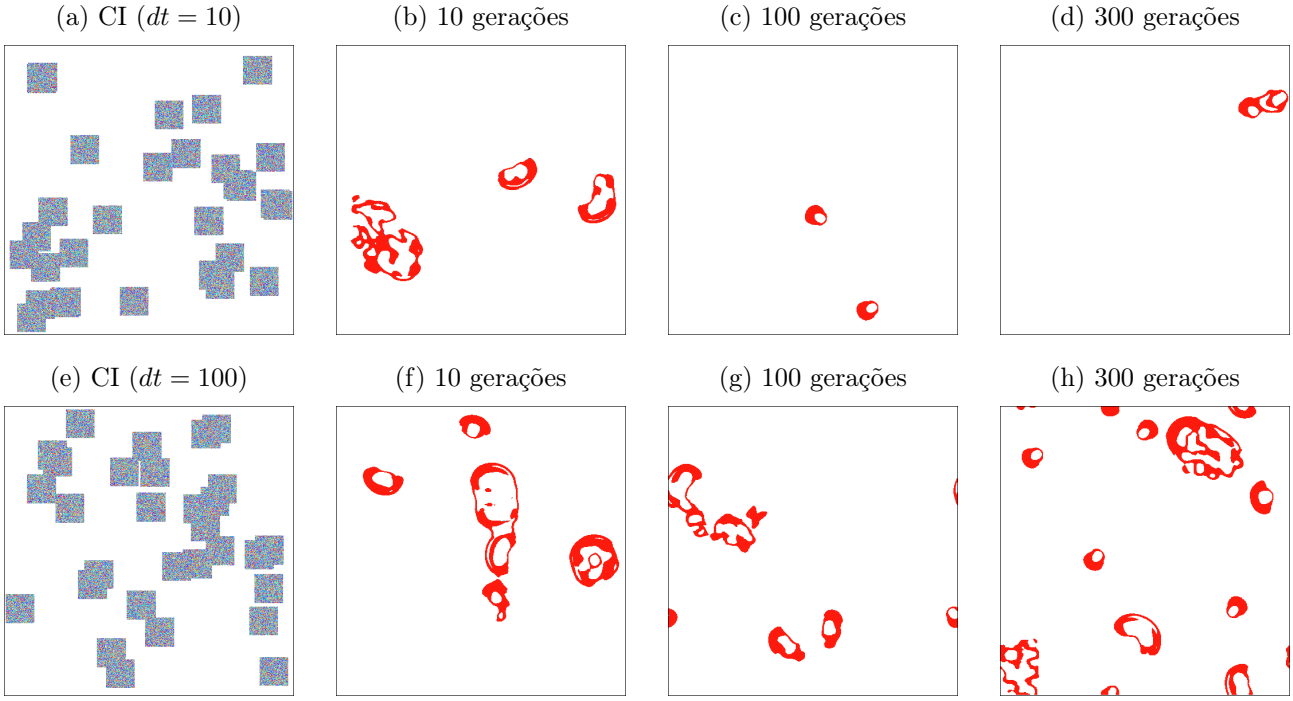


Figura 6.22: Evoluções temporais para SmoothLife com valores discretos de dt. Autoria própria.

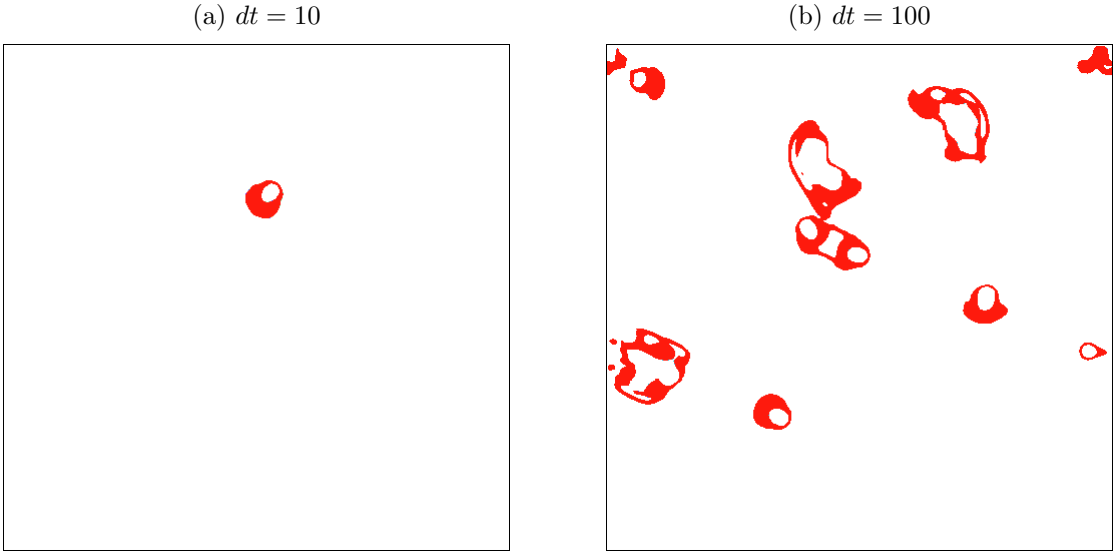


Figura 6.23: As gerações finais para $dt = 10$ e $dt = 100$. Autoria própria.

A partir das figuras acima, observa-se que o valor de dt parece influenciar diretamente no andamento das simulações, "acelerando" a evolução para o mesmo número de gerações.

Para os parâmetros com $dt = 10$, nota-se que a rede rapidamente converge para apenas dois *smooth gliders*, que se deslocam livremente até colidirem na segunda metade da simulação. Ao analisar todos os *snapshots* produzidos, nota-se que esta interação inicialmente resulta em dois *gliders*, e um rapidamente desaparece enquanto o outro se desloca com sua forma instável. Este *smooth glider* sobrevivente se expande e divide-se então em novos pares. Alguns dos *gliders* assim criados são perfeitamente estáveis e se deslocam até uma interação, enquanto outros reproduzem o comportamento de fissão da figura criadora ou desaparecem. Assim, a simulação

se encerra com apenas um *smooth glider* perfeitamente estável em toda a rede, que, na ausência de outros padrões, se desloca indeterminadamente em uma direção fixa.

Por outro lado, para $dt = 100$, a rede mostra-se repleta de *smooth gliders* e resíduos formados por suas colisões nas gerações observadas. Como na simulação para $dt = 10$, enquanto muitos destes padrões demonstram o fenômeno de autorreplicação, alguns desaparecem e outros são estáveis e independentes e se deslocam arbitrariamente até ocorrer uma interação. Por conta da quantidade de espécimes na rede e a velocidade aumentada das interações, gerando alterações drásticas de uma geração a outra, esta simulação se demonstra mais volátil que a anterior.

Então, foi feito o gráfico de densidades populacionais para as duas simulações, disposto a seguir.

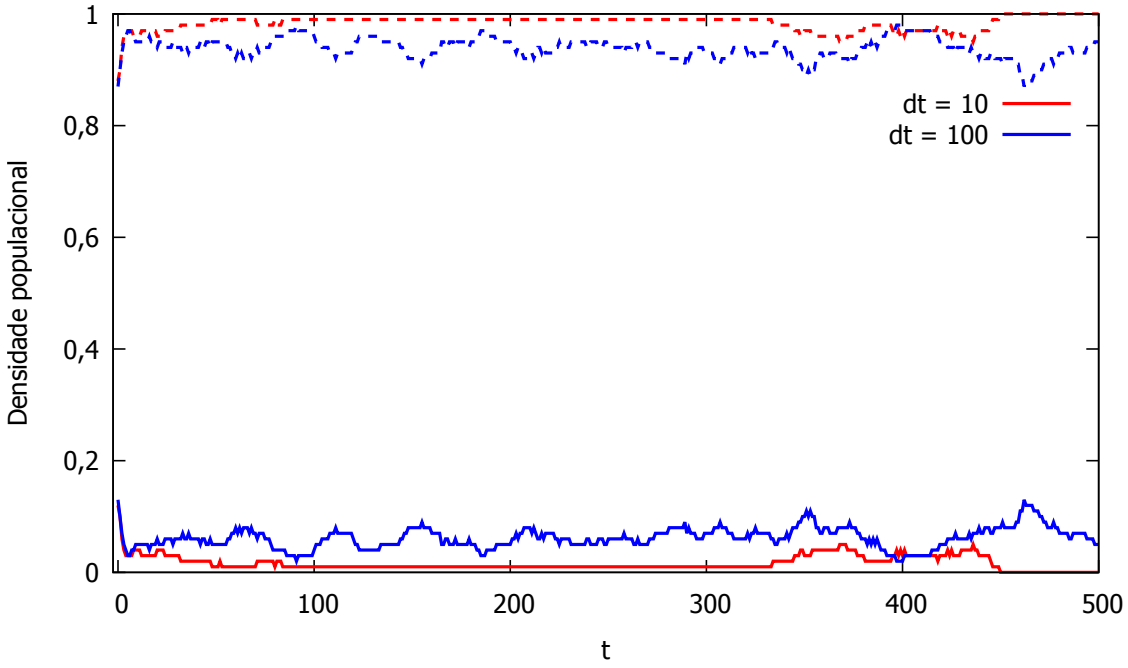


Figura 6.24: As evoluções temporais das densidades populacionais para $dt = 10$ e $dt = 100$. Autoria própria.

A partir da análise das curvas acima, as discrepâncias entre as redes com os dois valores de dt são aparentes. Primeiramente, embora a simulação com $dt = 10$ possua transformações notáveis em suas gerações iniciais, esta atinge estabilidade numérica por um longo período, representado visualmente pela coexistência independente dos dois *smooth gliders*. Na segunda metade da simulação, estes se encontram e causam um novo período de instabilidade, que é rapidamente extinto com a sobrevivência de um *glider* estável. Desta forma, a simulação para $dt = 10$ convergiu para uma rede estável e predominantemente inativa, e manteve uma baixa porcentagem média de células não nulas.

Para $dt = 100$, é possível notar instabilidade durante toda a simulação, sinalizando que esta não encontrou nenhum período prolongado de equilíbrio. Considerando a quantidade maior de espécimes sobreviventes e as rápidas alterações a cada geração, seus períodos estáveis foram numerosos, porém breves, rapidamente substituídos por picos ou vales. Houve interações entre espécimes até a última geração, demonstrando a volatilidade encontrada para esta simulação.

Por fim, comparando as duas simulações realizadas, pode-se inferir que o parâmetro dt está diretamente relacionado com a instabilidade populacional da rede, dado que a simulação com maior dt mostrou-se a com maior quantidade de interações entre espécimes e a maior porcentagem média de células ativas.

SmoothLifeL (dt = 0,1 e dt = 0,01)

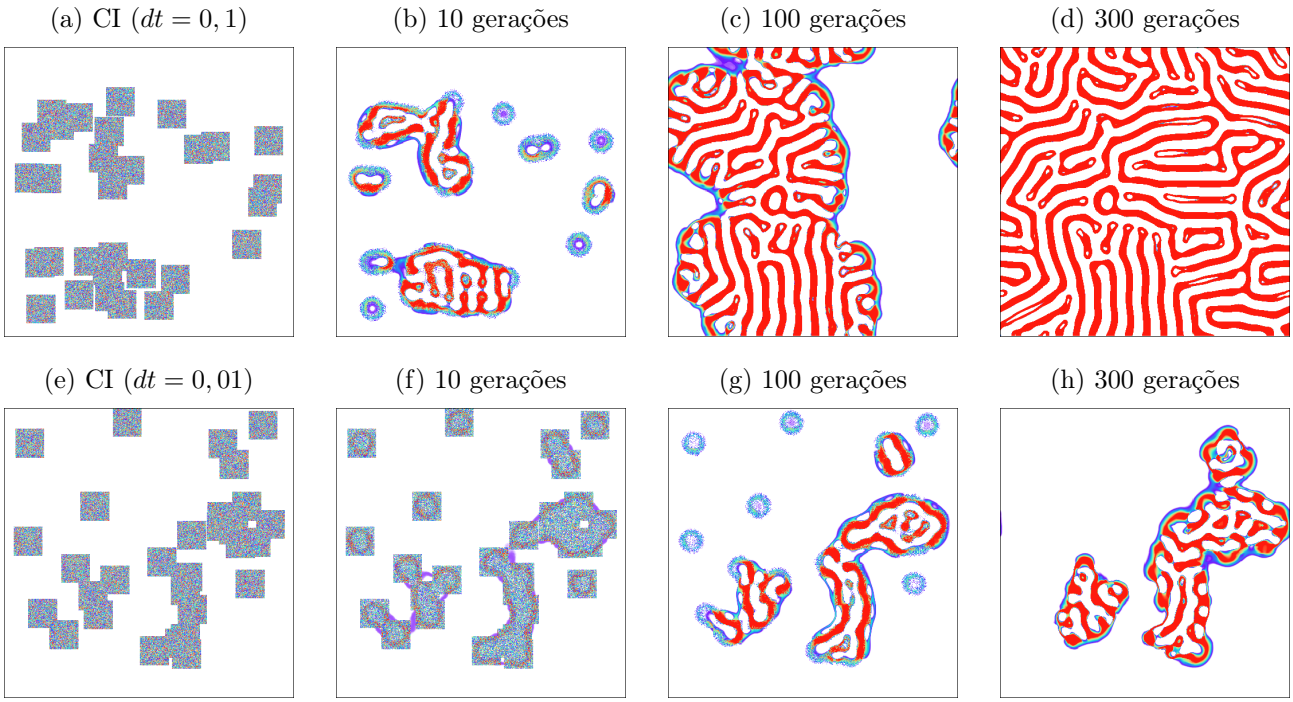


Figura 6.25: Evoluções temporais para SmoothLifeL. Autoria própria.

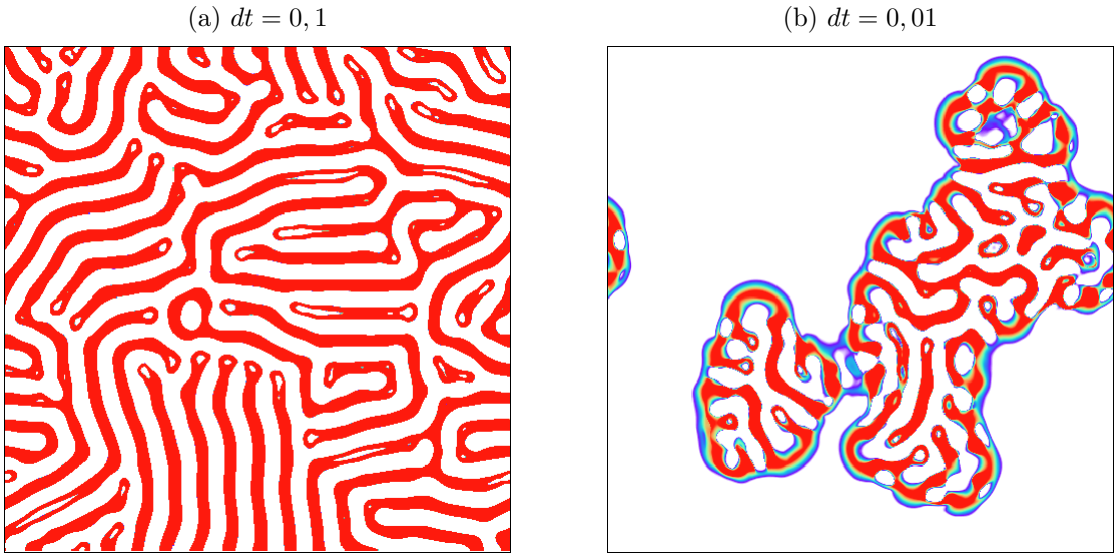


Figura 6.26: As gerações finais para SmoothLifeL, com $dt = 0,1$ e $dt = 0,01$. Autoria própria.

Analisando as figuras acima, observa-se que as simulações aparentam evoluir lentamente, porém ambas parecem convergir para o mesmo fim: a massa expansiva descrita no capítulo 2.

Para $dt = 0,1$, as áreas não nulas próximas rapidamente se unem para formar um único local, enquanto regiões afastadas prontamente desaparecem. A partir disso, inicia-se a expansão, ilimitada, até que as regiões não nulas unem-se e continuam a expansão para toda a rede, dominando-a. Para $dt = 0,01$, a mesma situação ocorre, porém mais lentamente. Devido aos parâmetros serem os mesmos, divergindo apenas no valor de dt, a forma obtida para a massa expansiva também é semelhante.

Em seguida, têm-se as curvas obtidas acerca das densidades populacionais para cada simulação.

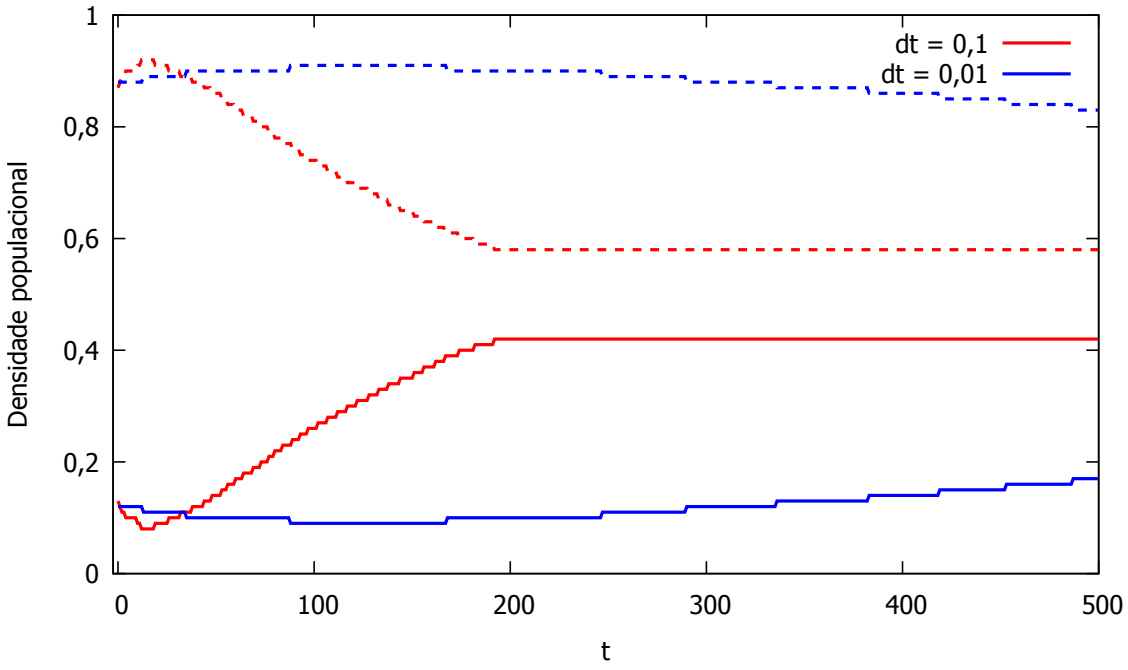


Figura 6.27: As densidades populacionais para SmoothLifeL. Autoria própria.

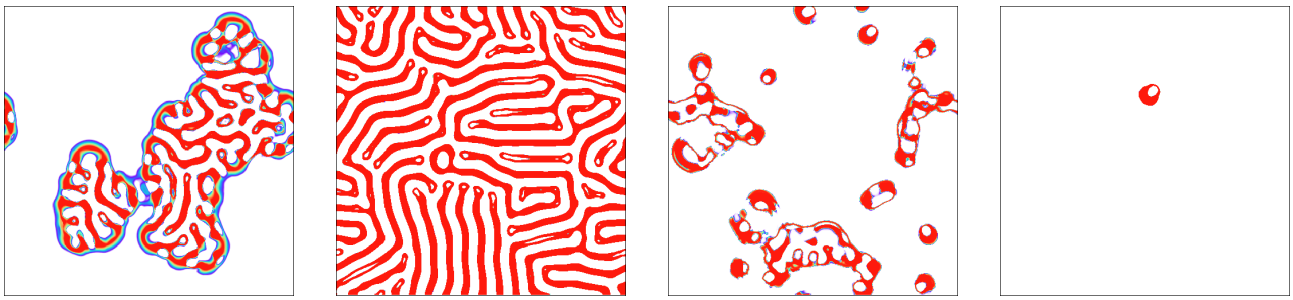
A partir do gráfico acima, nota-se então que as simulações desta seção possuem clara tendência de aumento da população ativa, fato representado visualmente pela obtenção da massa expansiva. Apesar disto, como visto por meio das curvas para $dt = 0,1$, a rede ainda é predominantemente ocupada por células nulas. Observando as imagens esboçadas para a rede, evidencia-se que isso ocorre por conta dos sulcos presentes na massa expansiva, que surgem em maior número do que as regiões preenchidas.

Para $dt = 0,01$, a tendência de crescimento da população ativa ainda é percebida, porém substancialmente mais lenta que a simulação anterior. As curvas obtidas para este valor de dt confirmam a dedução anterior de que, executada por mais gerações, esta rede obteria o mesmo fim que a anterior.

Comparações

Nesta seção, estão dispostas novamente as gerações finais para cada valor de dt utilizado, a fim de melhor analisar o impacto do valor de dt no andamento de SmoothLife, com outros parâmetros iguais. Evidentemente, para uma análise justa, utilizou-se os resultados para a mesma condição inicial de regiões não nulas.

(a) $dt = 0,01$ (b) $dt = 0,1$ (c) $dt = 1$ (d) $dt = 10$



(e) $dt = 100$

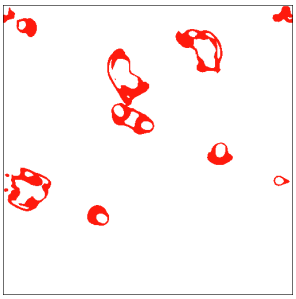


Figura 6.28: As gerações finais das simulações para cada valor de dt. Autoria própria.

A partir destes resultados, nota-se que valores decimais de dt, como $dt = 0,01$ e $dt = 0,1$, possuem uma propensão à expansão e união de áreas não nulas próximas, culminando em uma massa expansiva e eventualmente tornando a rede numericamente estável. Dessa forma, é possível deduzir que SmoothLifeL possui uma tendência maior à estabilidade do que SmoothLife discreto.

Em contrapartida, valores inteiros de dt como $dt = 1$, $dt = 10$ e $dt = 100$ são aptos a auxiliar na formação de padrões autônomos, como o *smooth glider*. Tais formas, quando interagem, não possuem a inclinação à expansão infinita como vista em SmoothLifeL, mas rapidamente se expandem e contraem até desaparecer ou criar novos *gliders*. Assim, este cenário de interações e criação de *smooth gliders* se repete indefinidamente. No caso específico da simulação para $dt = 10$, a rede encontrou um fim estável com a sobrevivência de um único *glider*, o que não é comum ou esperado. Então, trata-se esta ocorrência como consequência da geração aleatória da condição inicial e da subsequente evolução temporal.

Assim, pode-se inferir que valores decimais de dt influenciam em uma tendência da rede a formar uma massa expansiva, independente da condição inicial gerada. Além disso, valores inteiros de dt costumam produzir formas independentes, cujas características dependem do restante dos parâmetros, e a diferença fundamental entre os valores discretos utilizados está na velocidade de transformação da rede de uma geração a outra.

A fim de também comparar as densidades populacionais para cada valor de dt, foi confeccionado um novo gráfico reunindo tais curvas, exposto a seguir.

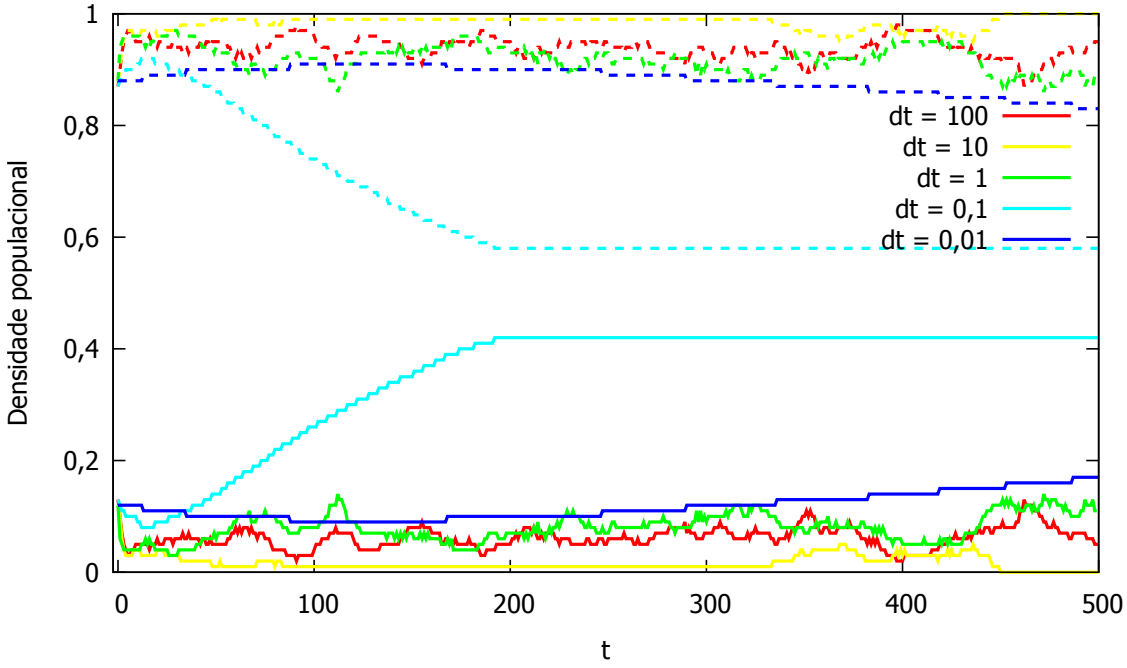


Figura 6.29: O gráfico de densidades populacionais para todos os valores de dt estudados. Autoria própria.

Como discutido anteriormente e reiterado por meio das figuras finais de cada rede, nota-se que valores decimais de dt tendem a densidades populacionais constantes, mas não necessariamente quantidades iguais de células nulas e não nulas. A disparidade entre valores decimais de dt sugere que valores menores como $dt = 0,01$ causam evolução mais lenta, embora os dois valores influenciem a rede ao mesmo fim de alterações que se neutralizam, mantendo densidades populacionais constantes pelo restante das gerações.

Em contraste, valores discretos de dt parecem não trazer convergência a uma densidade populacional específica, como demonstrado por suas curvas caóticas. Enquanto $dt = 1$ aparenta manter maior densidade média de células ativas, em inúmeros pontos é superado por $dt = 100$, e em algumas regiões as três simulações discretas se aproximam em densidade populacional. Portanto, não é possível inferir uma relação entre o valor discreto de dt e a evolução da rede, assim como não é possível prever um fim específico para estas simulações.

6.2.3 Outros Parâmetros

Para as condições iniciais destas simulações, a fim de desestimular a formação rápida de massa expansiva com valores decimais de dt, substituiu-se as regiões não nulas de 30 quadrados de 50×50 células para 15 quadrados de dimensões 45×45 células. Os parâmetros utilizados para cada simulação se encontram na Metodologia.

Simulação 1

(a) Condição inicial (b) 10 gerações (c) 100 gerações (d) 300 gerações

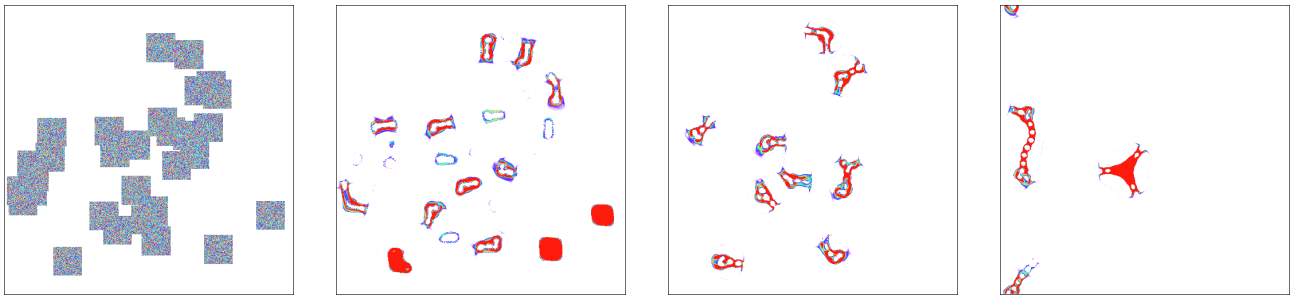


Figura 6.30: A evolução temporal de SmoothLife para os parâmetros mencionados. Autoria própria.

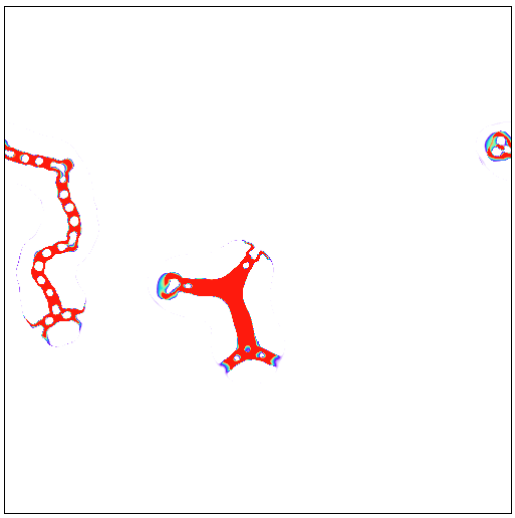


Figura 6.31: A geração final para a Simulação 1. Autoria própria.

Para os parâmetros utilizados e a condição inicial de quadrados não nulos, observou-se a formação rápida de espécimes autônomos e diferentes do usual *smooth glider*, mas que possuem sua própria forma de deslocamento através da rede. Porém, muitos destes espécimes experienciaram uma gradual desintegração em um ponto deste movimento, reduzindo substancialmente a população ativa da simulação. Os padrões sobreviventes passaram a se deslocar e estender suas próprias formas pela rede por meio da criação de "correntes". Tais correntes autorreplicam, como visto nas últimas gerações, ou desaparecem, como ocorreu para o espécime próximo ao centro.

Assim, os padrões observados mantêm um ciclo de criação e destruição de anexos, enquanto ao menos parte de sua estrutura estabiliza por algumas gerações antes de iniciar sua expansão ou acompanhar o deslocamento de outra parte.

Como de costume, foi criado um gráfico de densidade populacional conforme a progressão temporal, exposto a seguir.

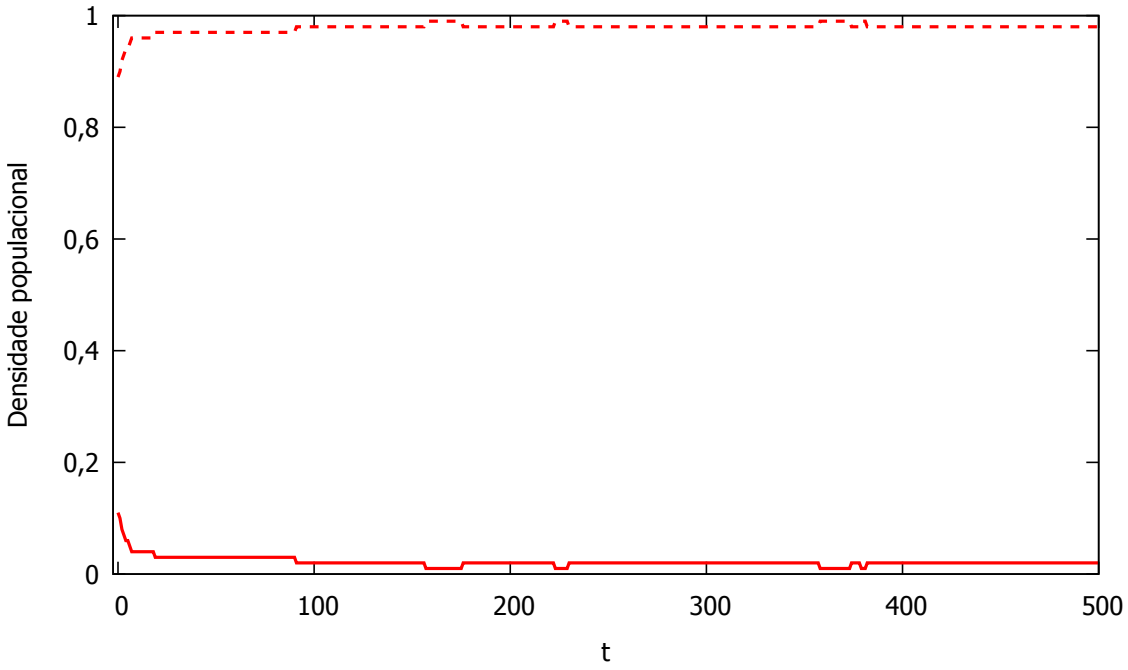


Figura 6.32: A evolução temporal da densidade populacional para a Simulação 1. Autoria própria.

Ao analisar este gráfico, nota-se a baixa variação da população da rede conforme a simulação progrediu, com vários períodos de porcentagens constantes e poucas variações. Tais variações, de fato, foram apenas reduções temporárias da população ativa da rede, então esta não obteve aumento em nenhum período por toda a simulação. Este fato se deve aos valores muito próximos dados ao intervalo de nascimento, $b_1 = 0,092$ e $b_2 = 0,098$. Este intervalo estreito, junto ao intervalo amplo de desaparecimento ($d_1 = 0,256$ e $d_2 = 0,607$), permitiram à rede apenas neutralizar algumas quedas de população, mas não superar a porcentagem obtida após as primeiras dezenas de gerações: em média, apenas 2% da rede foi povoada por células não nulas.

Assim, em termos de densidade populacional, os parâmetros citados não promovem mudanças significativas além das gerações imediatas após a aplicação da condição inicial.

Simulação 2

(a) Condição inicial (b) 10 gerações (c) 100 gerações (d) 300 gerações

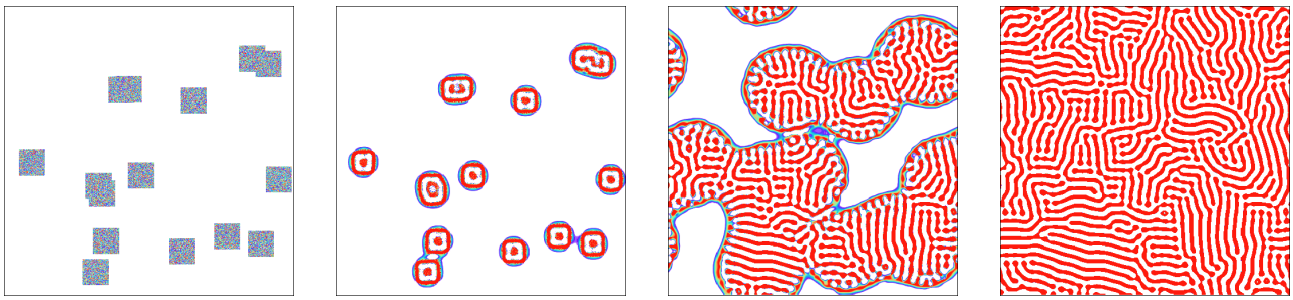


Figura 6.33: A evolução temporal de SmoothLife para os parâmetros mencionados. Autoria própria.

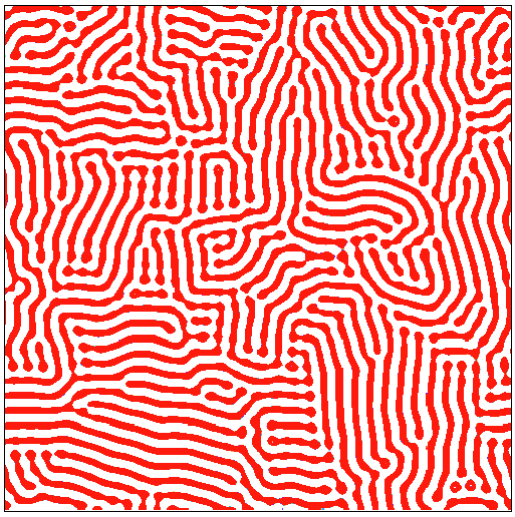


Figura 6.34: A geração final para a Simulação 2. Autoria própria.

Para esta simulação, foi observado que, embora a condição inicial tenha buscado desestimular a expansão indefinida pela rede, esta ocorreu após poucas dezenas de gerações. Quanto aos padrões anteriores a esta expansão, estes possuem forma aproximadamente esférica ou elíptica e um núcleo esférico ou aneliforme. Conforme estes se expandem independentemente, partes da elipse exterior se aproximam do núcleo, formando outros pequenos núcleos elipsoides.

Quando estes padrões interagem, porém, ocorre uma fusão e os núcleos gradualmente se transformam em formas alongadas, enquanto o contorno se expande espontaneamente. Assim, a simulação tende à massa expansiva, embora seu preenchimento seja notavelmente diferente comparado ao encontrado com os parâmetros de Rafler. Como em todas as ocorrências de massa expansiva estudadas, a rede não se manteve perfeitamente estática após esta formação, mas sim sofreu transformações que se neutralizavam, inalterando a população absoluta da rede.

As curvas acerca da densidade populacional desta simulação estão expostas a seguir.

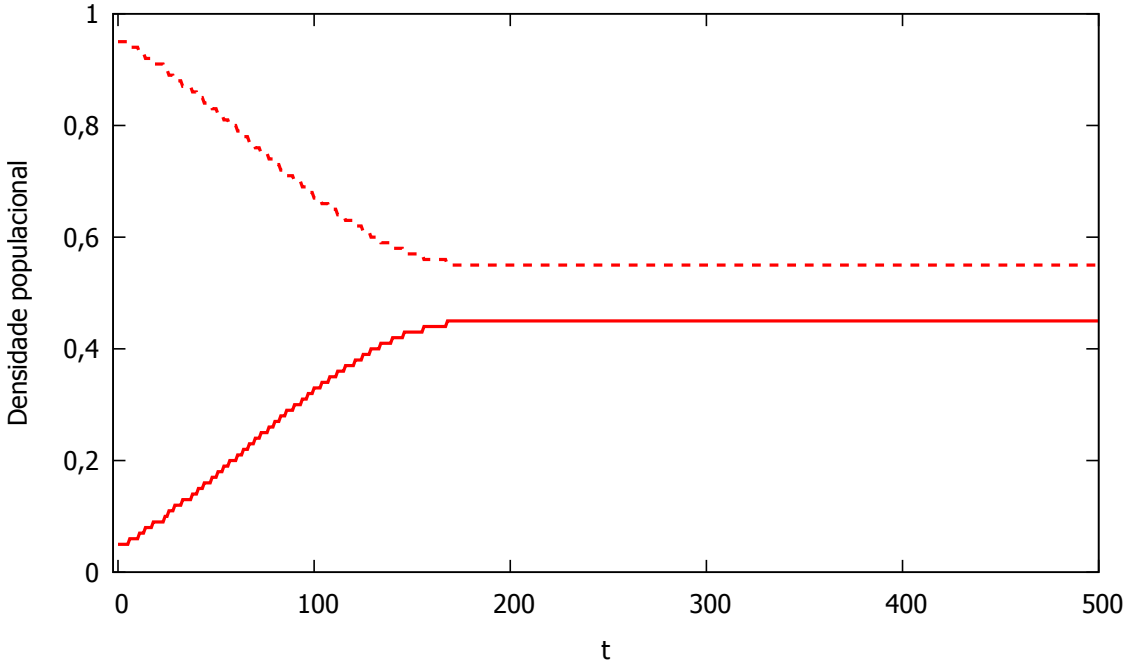


Figura 6.35: A evolução temporal da densidade populacional para a Simulação 2. Autoria própria.

Conforme já observado para o surgimento de uma massa expansiva em SmoothLife, a densidade populacional das células não nulas gradualmente aumenta até atingir um valor constante, que ainda é menor que a porcentagem de células nulas na rede. A discrepância visual entre a massa expansiva para os parâmetros desta simulação e aquela encontrada para os parâmetros de Rafler em nada alteram as curvas gerais do gráfico, sendo as únicas mudanças o tempo em que a densidade populacional constante foi atingida e a proximidade das porcentagens de população ativa e inativa. Nesta simulação, o equilíbrio ocorreu algumas gerações antes, e nele as populações alcançam valores mais próximos: aproximadamente 45% da rede possui células ativas, e 55% é composta de células nulas.

Simulação 3

(a) Condição inicial (b) 10 gerações (c) 100 gerações (d) 300 gerações

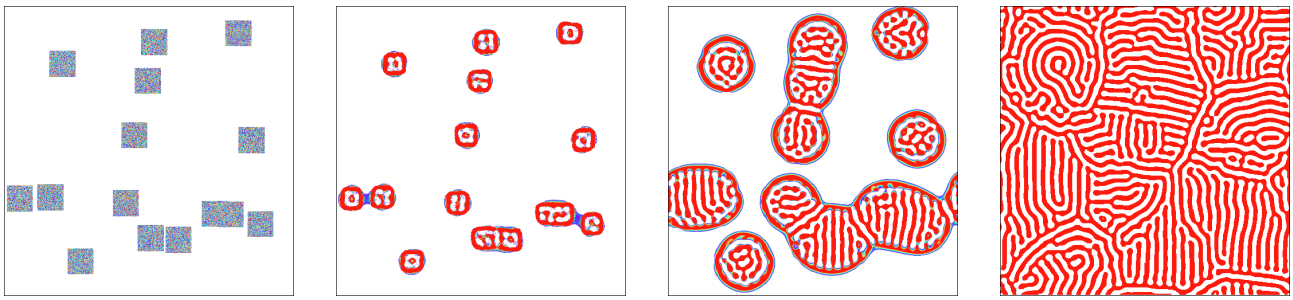


Figura 6.36: A evolução temporal de SmoothLife para os parâmetros mencionados. Autoria própria.

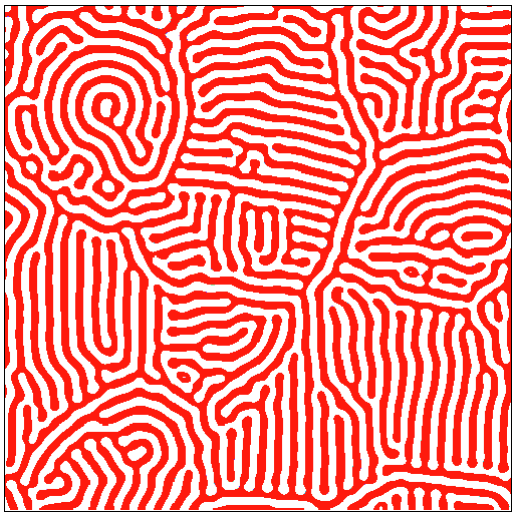


Figura 6.37: A geração final para a Simulação 3. Autoria própria.

Nota-se algumas similaridades desta simulação com a Simulação 2, devido ao uso repetido dos valores de alguns parâmetros (r_a, α_n, α_m e dt). Porém, os novos intervalos de nascimento e desaparecimento introduzem algumas mudanças aos padrões obtidos. Enquanto obtém-se um formato elíptico com núcleo esférico, este evolui mais lentamente. Assim, estes padrões expandem por várias gerações e o núcleo esférico se torna um anel que acompanha a expansão. Eventualmente, porém, o núcleo se transforma em curvas e sulcos, os espécimes começam a se unir e, como na simulação anterior, uma massa expansiva domina a rede.

Para este caso, porém, as curvas e os sulcos são levemente diferentes, e possuem uma tendência maior a reter o formato do espécime original conforme este se espalhava. Além disso, as curvas de células não nulas estão em maior número que nas massas expansivas encontradas em resultados anteriores. Novamente, a rede não se manteve estática, mas sim passou por alterações periódicas e pontuais.

A seguir, tem-se o gráfico para a densidade populacional da Simulação 3.

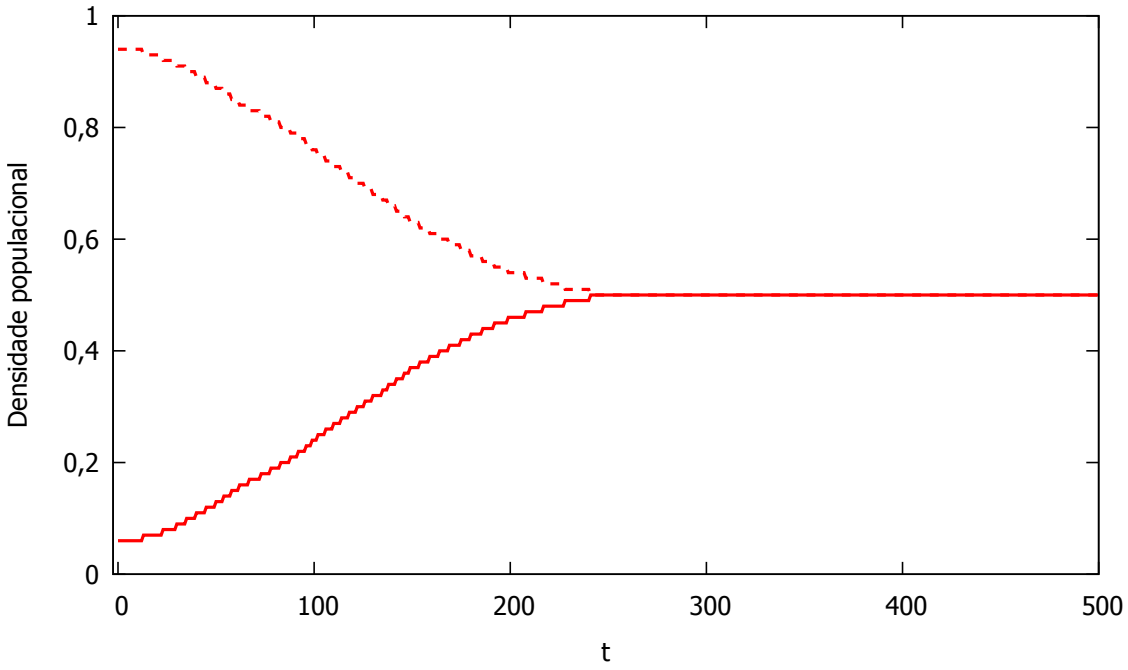


Figura 6.38: A evolução temporal da densidade populacional para a Simulação 3. Autoria própria.

Pela primeira vez neste trabalho, obteve-se uma simulação em que as populações nula e não nula convergiram para valores iguais, apesar da distribuição inicial de poucas células ativas. Para pouco menos da primeira metade das gerações, viu-se a propensão usual de simulações tipo SmoothLifeL a aumentar gradativamente a população não nula através de expansões pela rede. Diferindo de outras simulações, porém, este aumento não se restringiu, e a rede atingiu equilíbrio numérico ao alcançar a dominância da massa expansiva.

Comparações

Nesta seção, as três simulações de SmoothLife com parâmetros distintos aos de Rafler foram posicionadas lado a lado a fim de estudar o impacto de cada parâmetro nas formações encontradas em cada rede.

(a) Simulação 1 (b) Simulação 2 (c) Simulação 3

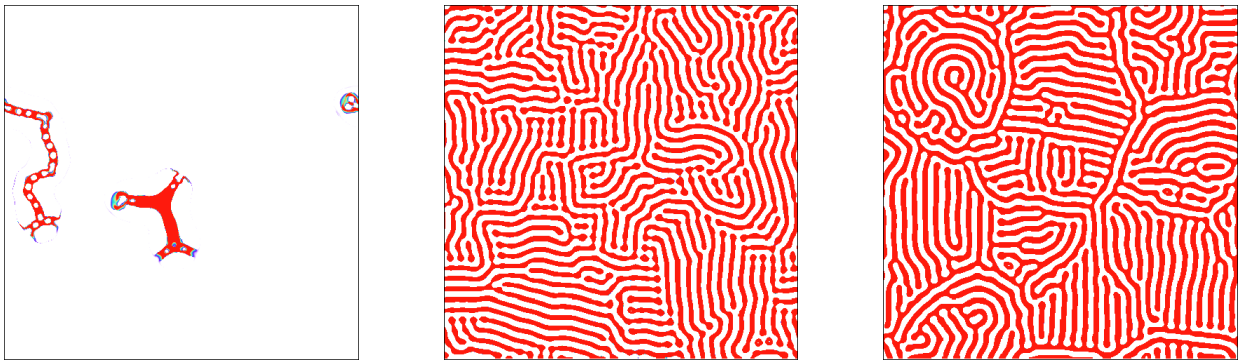


Figura 6.39: As gerações finais das Simulações 1, 2 e 3. Autoria própria.

Tem-se, então, que a Simulação 1 foi a única a não gerar uma massa expansiva. Em vez disso, gerou espécimes que podem se deslocar, que rapidamente desapareceram, e se alongar através da rede, replicando o comportamento de *wickstretchers* do Jogo da Vida de Conway. Fatores contribuintes a este resultado podem ser o estreito intervalo de nascimento, restringindo consideravelmente o aumento da população ativa na rede; bem como α_n e α_m, sendo a única simulação que atribui novos valores a eles. Considerando a função de α_n e α_m na suavidade dos passos temporais, é possível que suas grandezas, juntamente com o intervalo de nascimento, tenham sido essenciais para impedir a formação da massa expansiva como ocorreu na Simulação 2 e na Simulação 3.

Quanto à Simulação 2, esta possui o maior intervalo de nascimento das três, porém não o maior intervalo de desaparecimento. Ademais, tem o raio da vizinhança externa significativamente menor do que o utilizado na Simulação 1, além dos valores padrão para α_n e α_m. Apesar do raio menor, limitando as interações entre espécimes a ocorrerem com maior proximidade, isto não impediu a junção destes, levando à massa expansiva. Como nas simulações que utilizam os parâmetros de Rafler e que resultaram em massa expansiva, a rede atinge equilíbrio com menos células ativas que inativas. É possível que esta ocorrência tenha relação com os intervalos de nascimento e desaparecimento, considerando que são os parâmetros que a diferem da Simulação 3.

Enfim, analisando a Simulação 3 em contraste com as outras duas simulações, nota-se que todos os parâmetros diferem dos usados para a Simulação 1, porém vários são idênticos aos da Simulação 2. O fato desta simulação ter gerado uma massa expansiva com distribuição igualitária de células ativas e inativas, juntamente com os parâmetros semelhantes aos parâmetros da Simulação 2, novamente sugerem que os valores dos intervalos de nascimento e desaparecimento são fundamentais para a evolução da rede.

A seguir, estão expostas as curvas obtidas para as densidades populacionais de cada simulação.

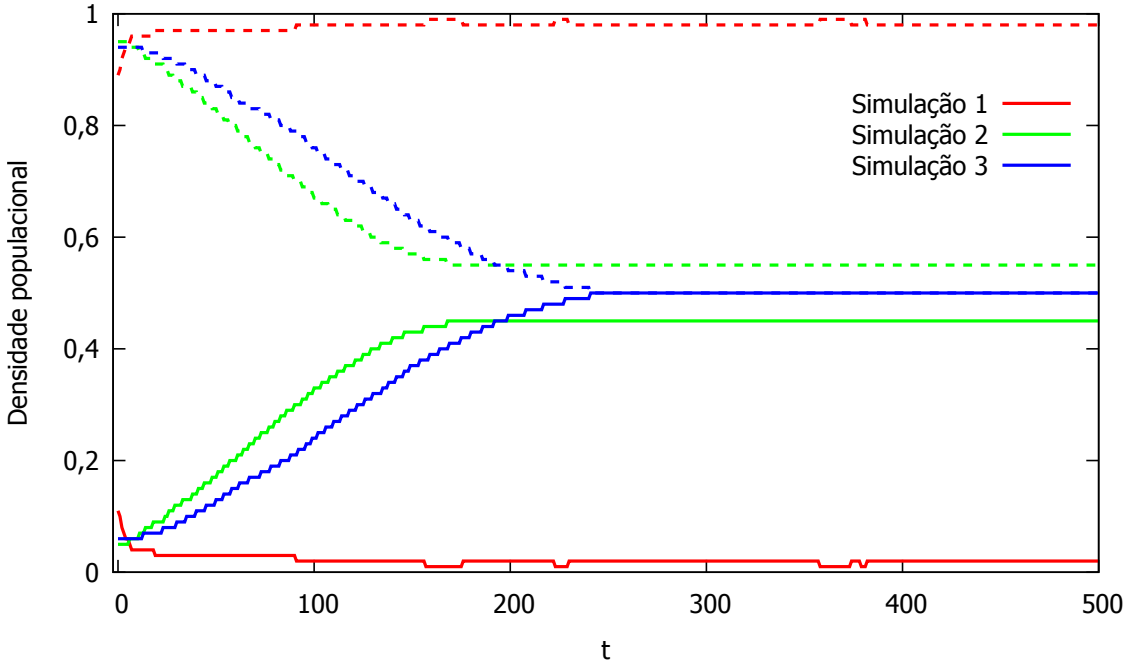


Figura 6.40: As curvas de densidade populacional para as Simulações 1, 2 e 3. Autoria própria.

As curvas de densidade populacional juntas evidenciam, em primeiro lugar, a discrepância entre a Simulação 1 e as Simulações 2 e 3. Enquanto a primeira mantém a queda irreversível da porcentagem de células ativas, periodicamente apresentando breves vales, as últimas possuem aumento gradual da população não nula, estagnando em algum ponto. Porém, as curvas ligeiramente diferentes das Simulações 2 e 3 também demonstram a influência dos intervalos de nascimento e desaparecimento na evolução temporal de SmoothLife; tais intervalos são as únicas grandezas diferentes entre elas. Dado que a Simulação 3 contém os maiores intervalos, estes fornecem maior probabilidade de nascimento de células ativas novas e de desaparecimento de células ativas existentes. Assim, é válido argumentar que tais intervalos mais amplos contribuíram para o equilíbrio entre nascimento e desaparecimento, resultando em uma rede igualmente dividida por células nulas e não nulas.

6.3 Lenia

6.3.1 Famílias

As evoluções temporais de cada simulação, com os parâmetros e funções condizentes para cada família estudada, estão dispostas a seguir.

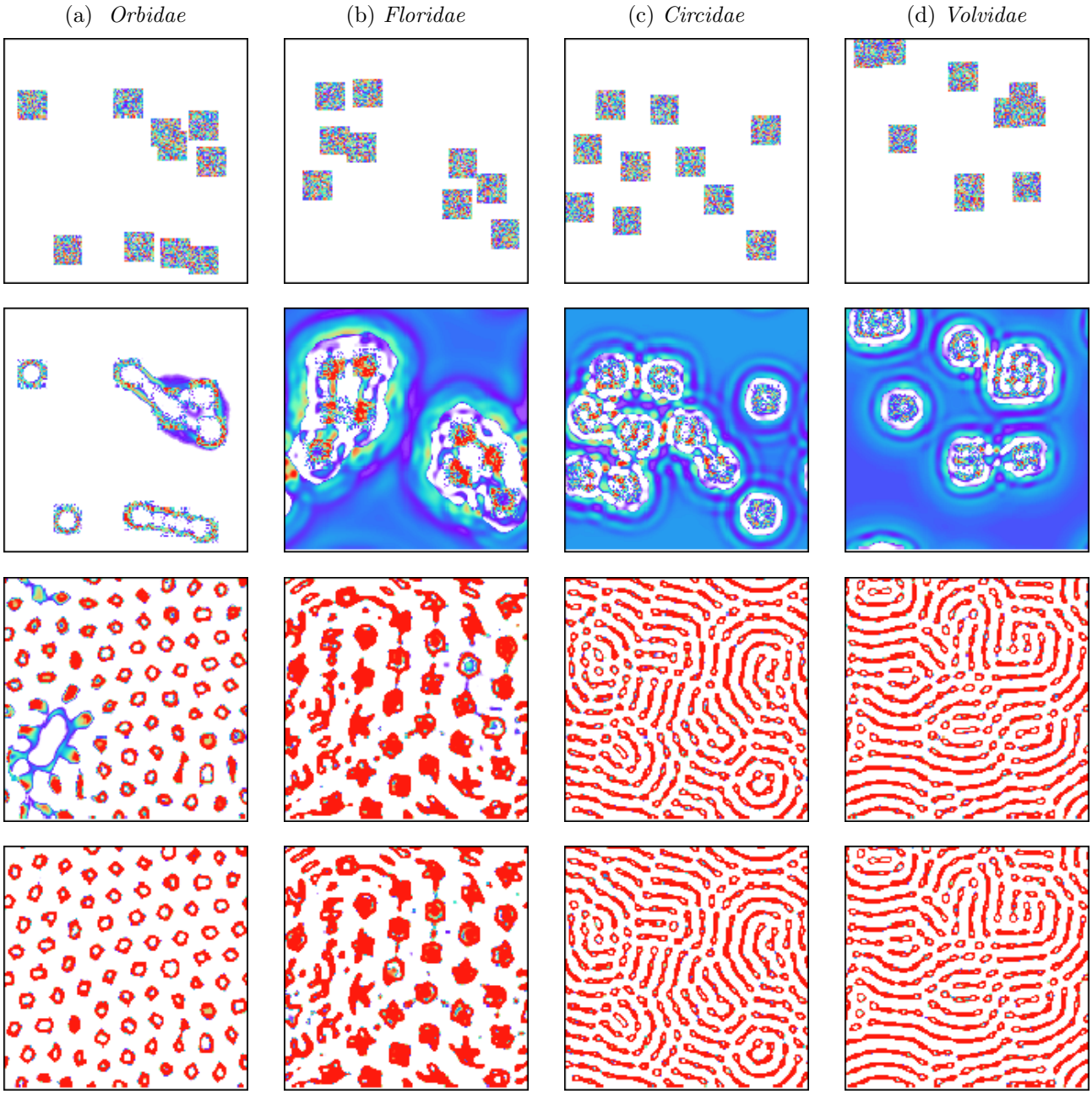


Figura 6.41: A progressão de 200 gerações de Lenia para diferentes famílias e primeira condição inicial. **Primeira linha:** condições iniciais. **Segunda linha:** gerações 10. **Terceira linha:** gerações 100. **Última linha:** gerações 200. Autoria própria.

A partir das figuras obtidas para estas simulações, percebe-se as discrepâncias que surgem ao utilizar diferentes funções de vizinhança e funções de crescimento. Ao final das execuções, todas as simulações resultaram em uma massa expansiva, como frequentemente ocorreu em SmoothLife, porém cada qual com suas características.

Para a simulação da família *Orbidae*, por exemplo, não foi observada a formação de sulcos ou curvas longas, mas sim de pequenas formas arredondadas de tamanhos variados. Analisando

a figura 6.3.1, no caso de *Orbidae*, vê-se que ainda havia transformações em curso, demonstradas pela coloração de algumas regiões. Ademais, nesta rede, a massa expansiva surgiu mais lentamente, dominando a rede depois das massas encontradas nas outras simulações.

Revisando estes atributos na Metodologia, nota-se que simulações com funções similares entre si (*Floridae*, *Circidae* e *Volvidae*) produzem resultados semelhantes. Para estas três simulações, há um padrão: forma-se um "mar" de células em estados menores (denotados por azul e violeta), e estes estados aumentam conforme se aproximam das regiões não nulas iniciais, até retornarem a zero, como se fosse mantida uma distância delas. Então, rapidamente surgem massas expansivas, que se unem e dominam a rede em algumas gerações.

Enquanto não se pode atribuir uma causa certa para este fenômeno, observando as similaridades entre as funções e os parâmetros usados para estas simulações, é possível cogitar que há alguma relação entre os tipos de funções utilizadas e a ocorrência descrita. As três simulações utilizam formas polinomiais (K_C e G) e bimodais (função K) para suas funções, e os parâmetros μ e σ não são suficientemente intensos para causar tais mudanças. Como mostrado pelas simulações das famílias *Circidae* e *Volvidae*, que diferem apenas nos parâmetros μ e σ utilizados, estes interferem principalmente no formato dos espécimes.

Ademais, percebe-se ainda que as massas expansivas de cada simulação são distintas entre si e mantêm o movimento das regiões não nulas originais, como "núcleos". Retornando à figura 4.7, nota-se que essas massas mantêm, em parte, a forma original dos espécimes encontrados para as respectivas funções e parâmetros. Este fato é especialmente aparente para as famílias *Orbidae* e *Floridae*.

Em seguida, foi confeccionado um gráfico reunindo as curvas de densidade populacional para cada simulação.

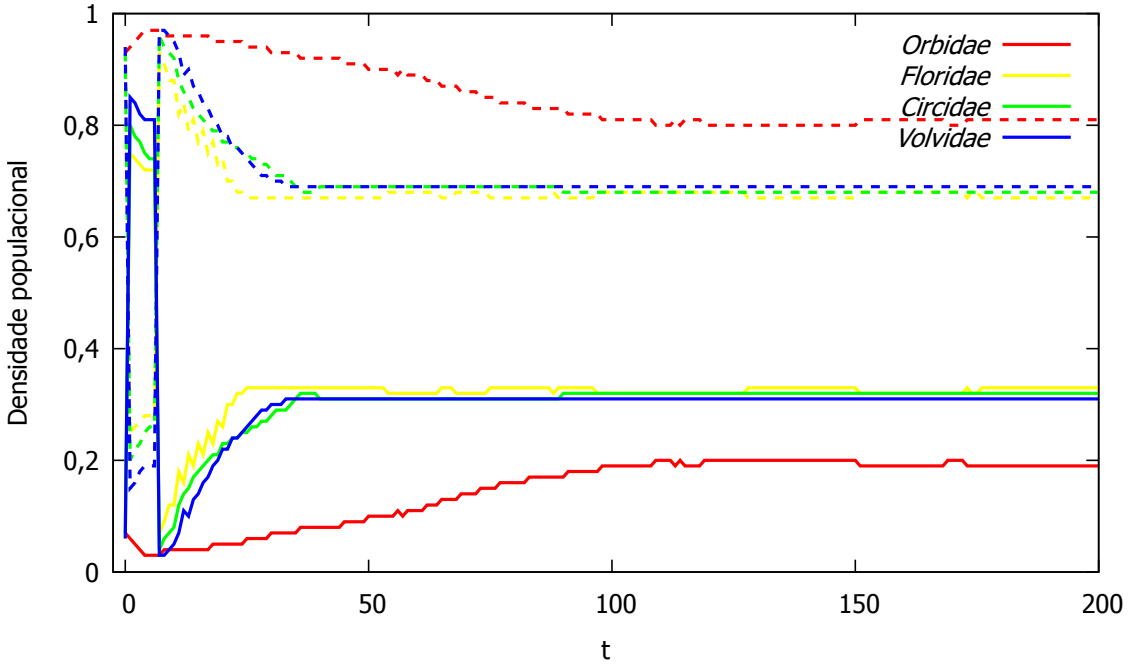


Figura 6.42: As curvas de densidade populacional para as famílias estudadas. Autoria própria.

Analisando as curvas acima, confirma-se que todas as simulações atingiram algum nível de estabilidade, mesmo com funções e parâmetros discrepantes. No entanto, vê-se que a simulação para a família *Orbidae* produziu as curvas menos drásticas. Quanto às simulações a respeito das famílias *Floridae*, *Circidae* e *Volvidae*, notam-se inícios turbulentos, porém com formas bastante semelhantes. Para os três casos, houve uma breve inversão das populações "vivo" e "morto", iniciando com um pico da população ativa e um declínio proporcional, eventualmente

transformando-se em um novo aumento gradual da população ativa.

Das quatro simulações apresentadas, apenas as das famílias *Circidae* e *Volvidae* verdadeiramente se estabilizaram, com *Floridae* e *Orbidae* sofrendo leves oscilações. As semelhanças das curvas de *Circidae* e *Volvidae* se devem ao fato de compartilharem as funções K_C e G, enquanto as discrepâncias são devidas aos parâmetros μ e σ diferentes.

Ao observar as curvas produzidas pelas famílias *Floridae*, *Circidae* e *Volvidae*, é possível traçar paralelos com os autômatos estudados anteriormente, Jogo da Vida e SmoothLife. As formas produzidas após os inícios caóticos sugerem semelhanças com o Jogo da Vida com as regras B1/S12 (figura 6.13) e SmoothLifeL quando $dt = 0,1$, nas simulações das figuras 6.27 e 6.35. É possível deduzir que as semelhanças ocorrem pois todas as simulações citadas produzem padrões expansivos, mas que mantêm vazios constantes, além do valor de dt compartilhado para as simulações citadas de SmoothLife e Lenia.

6.3.2 *Orbium*

(a) Condição inicial (b) 20 gerações (c) 43 gerações (d) 50 gerações

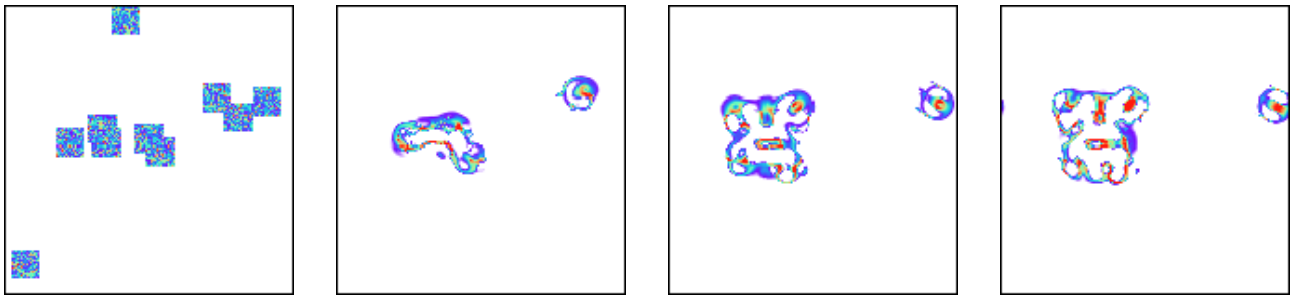


Figura 6.43: A evolução temporal de Lenia para a condição inicial visando estados menores. Autoria própria.

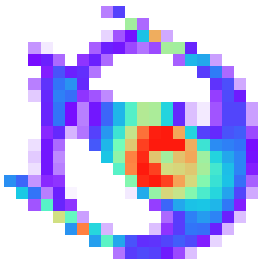


Figura 6.44: O *Orbium unicaudatus* obtido na geração 43 (*zoom* aplicado). Autoria própria.

Para esta simulação, houve o rápido desaparecimento de regiões não nulas isoladas e a união de regiões não nulas próximas, resultando em dois locais principais com rápidas transformações. Nestes locais, ocorreram contrações e expansões, até que do menor espaço surgiu um *Orbium unicaudatus*, cumprindo o objetivo da simulação. Entretanto, este espécime aparentemente possui vida breve, se transformando a partir da geração 50. Não é possível afirmar o fim dos padrões encontrados na última geração, embora os *snapshots* indiquem uma tendência à massa expansiva, como visto na figura 6.3.1.

Considerando as figuras acima e o fato de que o *Orbium unicaudatus* é considerado o *glider* primordial de Lenia, pode-se compará-lo aos *gliders* encontrados em outros autômatos, como o *glider* original do Jogo da Vida de Conway (figura 2.8) e o *smooth glider* para SmoothLife discreto (figura 3.2). Embora visivelmente diferentes, verifica-se que possuem a mesma morfologia básica, designada exclusivamente para deslocamento autônomo. Ademais, tratando de SmoothLife e Lenia, as simulações que os produzem têm dinâmicas semelhantes (figura 6.20), com transformações rápidas e locais.

Capítulo 7

Considerações Finais

O conceito de autômatos celulares, introduzido em 1940 por Stanislaw Ulam e John Von Neumann, alcançou popularidade inédita com a divulgação do Jogo da Vida em 1970, idealizado por John Horton Conway como um experimento de matemática recreativa. A partir das regras de Conway, centenas de conjuntos distintos de regras foram criados, levando ao surgimento de novos modelos, como Larger Than Life e RealLife, que destacaram a importância de transitar de dimensões e parâmetros discretos para contínuos na busca por formas de vida mais complexas.

O autômato celular SmoothLife, proposto por Stephan Rafler em 2011, avançou nessa transição ao empregar funções e números decimais, possibilitando o surgimento de formas de vida com movimento fluido e omnidirecional. Posteriormente, Bert Wang-Chak Chan criou Lenia, que generalizou ainda mais o domínio contínuo com funções complexas de potencial e crescimento, resultando na identificação de mais de 400 espécies distribuídas em 18 famílias. A alta biodiversidade e complexidade dinâmica de Lenia levantaram questões sobre os fatores que caracterizam vida e a busca por vida artificial.

Neste trabalho, foram realizadas simulações para os três autômatos principais citados: Jogo da Vida, SmoothLife e Lenia. As simulações exploraram diferentes condições iniciais, dimensões e parâmetros, analisando como essas variações influenciam as formas de vida geradas e suas dinâmicas. Embora os resultados tenham reproduzido comportamentos amplamente documentados, eles reforçam a riqueza e a versatilidade desses sistemas.

Para o Jogo da Vida de Conway, as simulações expuseram padrões conhecidos, como naturezas-mortas, alternadores, *gliders* e osciladores. Observou-se que condições iniciais com maior densidade de células vivas nem sempre geram mais padrões ao final das simulações. Além disso, conjuntos de regras alternativos resultaram em dinâmicas distintas, evidenciando a flexibilidade deste autômato.

No caso do SmoothLife, simulações mostraram que distribuições iniciais com poucas regiões não nulas são suficientes para gerar padrões estáveis. Um dos resultados obtidos foi o *smooth glider*, equivalente ao *glider* do Jogo da Vida, enquanto a alteração de um único parâmetro levou a resultados diversos, como dinâmicas instáveis e estabilização rápida da rede. Ademais, modificações de todos os parâmetros resultaram em novas formas de vida e comportamentos exclusivos a eles.

Para Lenia, simulações com diferentes funções e parâmetros resultaram em dinâmicas diversas, mas com padrões finais semelhantes, especialmente no caso em que equações e parâmetros similares foram usados. Em uma simulação específica, utilizando uma condição inicial diferente, foi possível obter o *Orbium*, a primeira forma de vida descoberta por Chan, destacando a biodiversidade característica desse modelo.

Analisando os três autômatos celulares, conclui-se que, apesar das discrepâncias nos métodos e objetivos de cada um, existem semelhanças significativas nas formas de vida e dinâmicas observadas. Tais resultados evidenciam o potencial desses sistemas como ferramentas para

representar fenômenos dinâmicos e explorar questões relacionadas à vida artificial.

Este trabalho contribui para a compreensão e a aplicação prática de autômatos celulares em geral, consolidando sua relevância como modelos versáteis e ricos em diversidade e possibilidades para futuras investigações. Para perspectivas futuras, temos o propósito de incorporar as ideias do modelo RPS (*rock, paper, scissors*), previamente estudado, ao sistema Lenia.

Referências Bibliográficas

ADAMATZKY, A. (Ed.). *Game of Life Cellular Automata*. Springer London, 2010. ISBN 9781849962179. Disponível em: <http://dx.doi.org/10.1007/978-1-84996-217-9>. 5, 8, 10, 11, 12, 13, 14, 15

BEDAU, M. A. et al. Open problems in artificial life. *Artificial Life*, v. 6, n. 4, p. 363–376, 10 2000. ISSN 1064-5462. Disponível em: <https://doi.org/10.1162/106454600300103683>. 22

BERLEKAMP, E. R.; CONWAY, J. H.; GUY, R. K. *Winning Ways for Your Mathematical Plays*. A K Peters/CRC Press, 2018. ISBN 9780429487330. Disponível em: <http://dx.doi.org/10.1201/9780429487330>. 5, 9

BITTENCOURT, B. D. C. 2024. Disponível em: <https://github.com/bbittencourt-go/TCC.git>. 33

CHAN, B. W.-C. *Lenia*. 2015. Disponível em: <https://chakazul.github.io/lenia.html>. 22

CHAN, B. W.-C. *Lenia*. 2015. Disponível em: <https://chakazul.github.io/Lenia/JavaScript/Lenia.html>. 34, 35

CHAN, B. W.-C. Lenia: Biology of artificial life. *Complex Systems*, Wolfram Research, Inc., v. 28, n. 3, p. 251–286, out. 2019. ISSN 0891-2513. Disponível em: <http://dx.doi.org/10.25088/ComplexSystems.28.3.251>. 2, 15, 20, 22, 24, 25, 26, 27, 28, 30, 31

EVANS, K. M. *Larger Than Life: it's so nonlinear*. 1996. Disponível em: <https://www.csun.edu/~kme52026/thesis.html>. 2, 15, 17, 18, 22

GARDNER, M. Mathematical games. *Scientific American*, Springer Science and Business Media LLC, v. 223, n. 4, p. 120–123, out. 1970. ISSN 0036-8733. Disponível em: <http://dx.doi.org/10.1038/scientificamerican1070-120>. 1, 2, 4, 6, 8, 9, 22

KADANOFF, L. P. More is the same; phase transitions and mean field theories. *Journal of Statistical Physics*, Springer Science and Business Media LLC, v. 137, n. 5–6, p. 777–797, set. 2009. ISSN 1572-9613. Disponível em: <http://dx.doi.org/10.1007/s10955-009-9814-1>. 2, 17

KOSHLAND, D. E. The seven pillars of life. *Science*, v. 295, n. 5563, p. 2215–2216, 2002. Disponível em: <https://www.science.org/doi/abs/10.1126/science.1068489>. 23

LIFEWIKI. *File:Lifeline vol 1.png*. 2009. Disponível em: <https://conwaylife.com/w/images/d/dd/Lifeline_vol_1.png>. 7

LYSENKO, M. *Conway's Game of Life for Curved Surfaces*. 2012. Disponível em: <https://0fps.net/tag/smooth-life/>. 18

MARTIN, E. *Play John Conway's Game of Life*. 2004. Disponível em: <https://playgameoflife.com>. 10

MCKAY, C. P. What is life—and how do we search for it in other worlds? *PLOS Biology*, Public Library of Science, v. 2, n. 9, p. null, 09 2004. Disponível em: <https://doi.org/10.1371/journal.pbio.0020302>. 23

NEUMANN, J. V. The general and logical theory of automata. In: *Systems research for behavioral science*. [S.l.]: Routledge, 2017. p. 97–107. 1

PIVATO, M. Reallife: The continuum limit of larger than life cellular automata. *Theoretical Computer Science*, Elsevier BV, v. 372, n. 1, p. 46–68, mar. 2007. ISSN 0304-3975. Disponível em: <http://dx.doi.org/10.1016/j.tcs.2006.11.019>. 2, 16, 17

RAFLER, S. *Generalization of Conway's "Game of Life"to a continuous domain - SmoothLife*. 2011. Disponível em: <https://arxiv.org/abs/1111.1567>. 2, 15, 17, 19, 22

RAFLER, S. *SmoothLife*. 2012. Disponível em: <https://sourceforge.net/projects/smoothlife/>. 20, 21

SCHIFF, J. *Cellular Automata: A Discrete View of the World*. Wiley, 2011. (Wiley Series in Discrete Mathematics & Optimization). ISBN 9781118030639. Disponível em: <https://books.google.com.br/books?id=uXJC2C2sRbIC>. 1

SCHRöDINGER, E. *What is Life? & Mind and Matter: The Physical Aspect of the Living Cell*. [S.l.]: Cambridge University Press, 1974. 23

SIMON, C.; BLUME, L. *Mathematics for Economists*. Norton, 1994. ISBN 9780393117523. Disponível em: <https://books.google.com.br/books?id=cxSaQgAACAAJ>. 18

SKONECZNY, S. Cellular automata as an effective tool for modelling of biofilm morphology. *Environment Protection Engineering*, v. 43, p. 177–190, 01 2017. 8

TOFFOLI, T.; MARGOLUS, N. *Cellular Automata Machines: A New Environment for Modeling*. Cambridge, 1987. (MIT Press series in scientific computation). ISBN 9780262200608. Disponível em: <https://books.google.com.br/books?id=HBlJzrBKUTEC>. 1

WEISSTEIN, E. W. *Moore Neighborhood*. 2003. Disponível em: <https://mathworld.wolfram.com/MooreNeighborhood.html>. 8

WOJTOWICZ, M. *Mirek's Cellebration*. 2001. Disponível em: <http://www.mirekw.com/ca/rullex_gene.html>. 16, 22

ZELDES, N. 2007. Disponível em: <https://www.nzeldes.com/Miscellany/Life.htm>. 1

Apêndice A

Códigos

A.1 Jogo da Vida de Conway

```
/* JOGO DA VIDA DE CONWAY
criado em        : 2024/10/31
ult. atualização: 2024/12/28
autor            : Beatriz Bittencourt <beatrizdecbittencourt@gmail.com>
notas            : Executa o Jogo da Vida de Conway (regra 23/3) em arquivos .dat
compilação       : -
execução         : go run jdv-tcc.go
*/

package main

// bibliotecas utilizadas
import (
  "fmt"
  "math/rand"
  "os"
  "time"
)

// constantes como dimensões x e y da rede (Nx e Ny) e o número de gerações (NG)
const (
  Nx, Ny, NG = 150, 150, 1000
)

// variáveis como x e y da célula, número do arquivo .dat, número da célula e contagens de células, gerações e arquivos
var (
  i, j, num, ind, gd, t, vd, ve, vb, vc, vd1, vd2, vd3, vd4, soma int
  phi [Nx * Ny]int // rede
  updt [Nx * Ny]int // rede após atualizações
  dat [2]int // matriz de densidade populacional
)

// função condição inicial, distribui 0 ou 1 para cada célula sucessivamente
func ic() {
  for i = 0; i < Nx; i++ { // para todas as células da rede
    for j = 0; j < Ny; j++ {
      state := rand.Float64() // número pseudoaleatório entre 0 e 1
      if state < 0.5 { // probabilidade
        phi[i*Ny+j] = 1 // vivo
```

```
      } else {
        phi[i*Ny+j] = 0 // morto
      }
    }
  }
}

// função op, imprime a rede ao final de cada geração
func op(num int, phi [Nx * Ny]int) {
  f := fmt.Sprintf("jdv-tcc-%v.dat", num) // arquivo jdv-tcc-0.dat (condição inicial)
  file, _ := os.Create(f) // cria o arquivo
  for i = 0; i < Nx; i++ { // para cada espaço no arquivo
    for j = 0; j < Ny; j++ {
      fmt.Fprintf(file, "%d ", phi[j*Nx+i]) // imprime o valor de cada célula em ordem
    }
    fmt.Fprintf(file, "\n") // pula linha ao atingir Ny
  }
  defer file.Close()
}

// função principal, Jogo da Vida ocorre aqui
func main() {
  rand.Seed(time.Now().UnixNano())
  ic() // ativa a condição inicial
  op(0, phi) // ativa op para imprimir a condição inicial

  for i = 0; i < Nx*Ny; i++ { // para toda a rede
    dat[phi[i]]++ // calcula a densidade populacional de cada estado
  }
  f, _ := os.Create("datjdv-tcc.dat") // cria o arquivo .dat de densidade populacional
  fmt.Fprintf(f, "%.2f  %.2f\n", float64(dat[1])/(Nx*Ny), float64(dat[0])/(Nx*Ny)) // imprime as densidades
  dat[1] = 0 // zera as densidades para serem atualizadas a cada geração
  dat[0] = 0
  for t = 1; t <= NG; t++ { // conta as gerações
    for gd = 0; gd < Nx*Ny; gd++ { // conta todas as células da rede
      for i = 0; i < Nx; i++ {
        for j = 0; j < Ny; j++ {
          ind = j*Nx + i // célula principal; variáveis v abaixo dizem respeito à vizinhança de Moore
          vd = j*Nx + ((i + 1) % Nx)
          ve = j*Nx + ((i - 1 + Nx) % Nx)
          vb = ((j+1)%Ny)*Nx + i
          vc = ((j-1+Ny)%Ny)*Nx + i
          vd1 = ((j+1)%Ny)*Nx + (i+1)%Nx
          vd2 = ((j-1+Ny)%Ny)*Nx + (i-1+Nx)%Nx
          vd3 = ((j+1)%Ny)*Nx + (i-1+Nx)%Nx
          vd4 = ((j-1+Ny)%Ny)*Nx + (i+1)%Nx
          soma = phi[vd] + phi[ve] + phi[vb] + phi[vc] + phi[vd1] + phi[vd2] + phi[vd3] + phi[vd4] // soma dos estados da vizinhança
          if phi[ind] == 1 { // se a célula está viva
            if (soma == 2) || (soma == 3) { // se 2 ou 3 vizinhos também estão vivos
              updt[ind] = 1 // sobrevive
            } else {
              updt[ind] = 0 // passa para o estado morto

```

```
93              }
94            } else { // se a célula está morta
95              if soma == 3 { // se há exatamente 3 vizinhos vivos
96                updt[ind] = 1 // nasce
97              } else {
98                updt[ind] = 0 // permanece morta
99              }
100           }
101         }
102       }
103     }
104     for i = 0; i < Nx; i++ { // para todas as células da rede
105       for j = 0; j < Ny; j++ {
106         phi[j*Nx+i] = updt[j*Nx+i] // troca a rede inicial para a atualizada
107         dat[phi[j*Nx+i]]++ // atualiza a densidade populacional
108       }
109     }
110     fmt.Fprintf(f, "%.2f  %.2f\n", float64(dat[1])/(Nx*Ny), float64(dat[0])
          /(Nx*Ny)) // imprime a densidade populacional atualizada
111     op(t, phi) // cria o arquivo .dat a cada geração (jdv-tcc-1, jdv-tcc
          -2...)
112     fmt.Printf("%d/%d\n", t, NG) // mostra gerações completas em tempo real
          no terminal
113     dat[1] = 0 // zera as densidades populacionais novamente
114     dat[0] = 0
115   }
116 }
```

A.2 SmoothLife

```
/* SMOOTHLIFE (STEPHAN RAFLER)
criado em       : 2024/11/20
ult. atualização: 2024/12/28
autor           : Beatriz Bittencourt <beatrizdecbittencourt@gmail.com>
notas           : Executa o SmoothLife (parâmetros originais) em arquivos .dat
compilação      : -
execução        : go run smoothlife-tcc.go
*/

package main

// bibliotecas utilizadas
import (
  "fmt"
  "math"
  "math/rand"
  "os"
  "time"
)

// constantes como dimensões x e y da rede (Nx e Ny), número de gerações (NG), parâmetros para geração da condição inicial e parâmetros do SL
const (
  Nx, Ny, NG, n_q, max_q, ra, alphaN, alphaM, b1, b2, d1, d2, dt, ri = 500, 500, 500, 30, 50, 21.0, 0.028, 0.147, 0.278, 0.365, 0.267, 0.445, 1.0, ra / 3
)

// variáveis como x e y da célula, número do arquivo .dat, número da célula e contagens de células, gerações e arquivos
var (
  i, j, x_q, y_q, num, t, n int
  phi [Nx * Ny]float64 // float64 para assumir estados decimais
  updt [Nx * Ny]float64 // idem
  dat [2]int
)

// função condição inicial, distribui aleatoriamente quadrados com células entre 0 e 1
func ic() {
  for n = 0; n < n_q; n++ { // cria quadrados até atingir uma quantidade máxima (n_q)
    x_q = rand.Intn(Ny - max_q) // coordenadas iniciais de cada quadrado (x e y)
    y_q = rand.Intn(Nx - max_q)
    for i = y_q; i < y_q+max_q; i++ { // na área dentro do perímetro...
      for j = x_q; j < x_q+max_q; j++ {
        phi[i*Ny+j] = rand.Float64() // ...células assumem estados aleatórios entre 0 e 1
      }
    }
  }
}

// função op, imprime a rede ao final de cada geração
func op(num int, phi [Nx * Ny]float64) {
  f := fmt.Sprintf("smoothlife-tcc-%v.dat", num) // arquivo smoothlife-tcc-(
```

```
    x).dat (a depender do número da geração)
  file, _ := os.Create(f) // cria o arquivo
  for i = 0; i < Nx; i++ { // para toda a rede
    for j = 0; j < Ny; j++ {
      fmt.Fprintf(file, "%f%f ", phi[j*Nx+i], phi[j*Nx+i]) // imprime o
          valor de cada célula em ordem, duas vezes
    }
    fmt.Fprintf(file, "\n") // pula linha ao atingir Ny
  }
  defer file.Close()
}

// equações do SmoothLife abaixo; todas são encontradas no artigo original (
    Stephan Rafler)

// função sigma
func sigma(x, a, alpha float64) float64 { // note abaixo que alpha toma os
    valores de alphaN e alphaM
  return 1.0 / (1.0 + float64(math.Exp(-float64((x-a)*4/alpha))))
}

// função sigmaN (depende de sigma)
func sigmaN(x, a, b float64) float64 {
  return sigma(x, a, alphaN) * (1 - sigma(x, b, alphaN))
}

// função sigmaM (depende de sigma)
func sigmaM(x, y, m float64) float64 {
  return x*(1-sigma(m, 0.5, alphaM)) + y*sigma(m, 0.5, alphaM)
}

// função de transição s(n, m) (depende de sigmaN e sigmaM)
func s(n, m float64) float64 {
  return sigmaN(n, sigmaM(b1, d1, m), sigmaM(b2, d2, m))
}

// função principal - a simulação ocorre essencialmente aqui
func main() {
  rand.Seed(time.Now().UnixNano())
  ic() // ativa a condição inicial
  op(0, phi) // ativa op para imprimir a condição inicial

  for i := 0; i < Nx*Ny; i++ { // calcula a densidade populacional de cada
      estado
    if phi[i] > 0.5 { // condição: vivo se > 0,5 e morto se < 0,5
      dat[1]++
    } else {
      dat[0]++
    }
  }
  f, _ := os.Create("datsmoothlife-tcc.dat") // cria o arquivo .dat de
      densidade populacional
  defer f.Close()
  fmt.Fprintf(f, "%.2f   %.2f\n", float64(dat[1])/(Nx*Ny), float64(dat[0])/(
      Nx*Ny)) // imprime as densidades em %
  dat[1] = 0 // zera as densidades para serem atualizadas a cada geração
  dat[0] = 0

  for t = 1; t <= NG; t++ { // para cada geração
    for i = 0; i < Nx; i++ { // para toda a rede
```

```
103       for j = 0; j < Ny; j++ {
104         m, M := float64(0), float64(0) // m, M, n e N começam em 0,0
105         n, N := float64(0), float64(0)
106 
107         for dy := -(ra - 1); dy <= (ra - 1); dy++ { // calcula
                preenchimentos da vizinhança com condições de contorno
108           for dx := -(ra - 1); dx <= (ra - 1); dx++ {
109             x := (int(float64(i) + dx + Nx)) % Nx
110             y := (int(float64(j) + dy + Ny)) % Ny
111             if dx*dx+dy*dy <= ri*ri { // vizinhança interna, preenchimento m
112               m += phi[y*Nx+x]
113               M++
114             } else if dx*dx+dy*dy <= ra*ra { // vizinhança externa,
                    preenchimento n
115               n += phi[y*Nx+x]
116               N++
117             }
118           }
119         }
120         m /= M
121         n /= N
122         q := s(n, m) // agora que tem n e m, calcula a função de transição
123         updt[j*Nx+i] = 2*q - 1 // atualização da rede
124       }
125     }
126     for i = 0; i < Nx; i++ { // para toda a rede
127       for j = 0; j < Ny; j++ {
128         phi[j*Nx+i] += math.Min(math.Max(dt * updt[j*Nx+i], 0), 1) // troca
              a rede inicial para a atualizada
129         if phi[j*Nx+i] > 0.5 { // atualiza a densidade populacional
130           dat[1]++
131         } else {
132           dat[0]++
133         }
134       }
135     }
136     fmt.Fprintf(f, "%.2f    %.2f\n", float64(dat[1])/(Nx*Ny), float64(dat[0])
          /(Nx*Ny)) // imprime a densidade populacional atualizada em %

          // aumenta a cada geração
137     op(t, phi) // cria o arquivo .dat a cada geração (smoothlife-tcc-1,
          smoothlife-tcc-2...)
138     fmt.Printf("%d/%d\n", t, NG) // mostra gerações completas em tempo real
          no terminal
139     dat[1] = 0 // zera as densidades populacionais novamente
140     dat[0] = 0
141   }
142 }
```

A.3 Lenia

```
/* LENIA (BERT WANG-CHAK CHAN)
criado em       : 2024/12/10
ult. atualização: 2024/12/27
autor           : Beatriz Bittencourt <beatrizdecbittencourt@gmail.com>
notas           : Executa Lenia (funções e parâmetros para Orbium) em arquivos .dat
compilação      : -
execução        : go run lenia-tcc.go
*/

package main

// bibliotecas utilizadas
import (
  "fmt"
  "math"
  "math/rand"
  "os"
  "time"
)

// constantes como dimensões x e y da rede (Nx e Ny), número de gerações (NG), parâmetros para geração da condição inicial e parâmetros do Lenia
const (
  Nx, Ny, NG, n_q, max_q, R, alpha, mu, sigma, T = 150, 150, 200, 10, 18, 13.0 4, 0.15, 0.017, 10.0
)

// variáveis como x e y da célula, número do arquivo .dat, número da célula e contagens de células, gerações e arquivos
var (
  i, j, x_q, y_q, num, t, n int
  phi [Nx][Ny]float64 // estado da célula (2D por exigência do Lenia, e float64 para assumir estados decimais)
  dat [2]int
)

// função condição inicial, distribui aleatoriamente quadrados com células entre 0 e 1
func ic() {
  for n = 0; n < n_q; n++ { // cria quadrados até atingir uma quantidade máxima (n_q)
    x_q = rand.Intn(Nx - max_q) // coordenadas iniciais de cada quadrado (x e y)
    y_q = rand.Intn(Ny - max_q)
    for i = y_q; i < y_q+max_q; i++ { // na área dentro do perímetro...
      for j = x_q; j < x_q+max_q; j++ {
        phi[i][j] = rand.Float64() // ...células assumem estados aleatórios entre 0 e 1
      }
    }
  }
}

// função op, imprime a rede ao final de cada geração
func op(num int, phi [Nx][Ny]float64) {
  f := fmt.Sprintf("lenia-tcc-%v.dat", num) // arquivo lenia-(x).dat (a depender do número da geração)
```

```
  file, _ := os.Create(f) // cria o arquivo
  for i = 0; i < Nx; i++ { // para toda a rede
    for j = 0; j < Ny; j++ {
      fmt.Fprintf(file, "%f%f ", phi[i][j], phi[i][j]) // imprime o valor de cada célula em ordem, duas vezes
    }
    fmt.Fprintf(file, "\n") // pula linha ao atingir Ny
  }
  defer file.Close()
}

// função de convolução manual, dispensa bibliotecas extras
func convolucao(entrada1 [Nx][Ny]float64, entrada2 [][]float64, Nx, Ny int) [][]float64 { // entradas: matrizes que participam da convolução
  len_entrada2 := len(entrada2)
  offset := len_entrada2 / 2
  saida := make([][]float64, Nx) // matriz resultado da convolução
  for i := range saida {
    saida[i] = make([]float64, Ny)
  }

  for i := 0; i < Nx; i++ { // para toda célula
    for j := 0; j < Ny; j++ {
      var soma float64
      for ki := 0; ki < len_entrada2; ki++ { // condições de contorno
        for kj := 0; kj < len_entrada2; kj++ {
          x := (i + ki - offset + Nx) % Nx
          y := (j + kj - offset + Ny) % Ny
          soma += entrada1[x][y] * entrada2[ki][kj] // convolução em si
        }
      }
      saida[i][j] = soma
    }
  }

  return saida //função convolução retorna o resultado
}

// equações do Lenia abaixo; todas são encontradas no artigo original (Bert Wang-Chak Chan, 2019)

// função Kernel, cria a vizinhança para toda célula
func Kernel(Nx, Ny int, R float64) [][]float64 {
  kernel := make([][]float64, Nx) // cria a matriz da vizinhança
  for i := range kernel {
    kernel[i] = make([]float64, Ny)
  }

  for i := 0; i < Nx; i++ { // para toda célula...
    for j := 0; j < Ny; j++ {
      dx := float64(i - Nx/2) // coordenadas da distância polar
      dy := float64(j - Ny/2)
      r_polar := math.Sqrt(dx*dx + dy*dy) / R // fórmula do raio polar
      if r_polar < 1 { // se o raio polar for menor que o raio da vizinhança ...
        kernel[i][j] = math.Exp(alpha * (1 - (1 / (4*r_polar*(1-r_polar))))) // ...cria a área de influência (nesse caso, apenas kernel core)
      }
    }
  }
```

```
104
105   // normaliza a vizinhança
106   var soma float64
107   for i := range kernel {
108     for j := range kernel[i] {
109       soma += kernel[i][j]
110     }
111   }
112   for i := range kernel {
113     for j := range kernel[i] {
114       kernel[i][j] /= soma
115     }
116   }
117
118   return kernel // função Kernel retorna a matriz da vizinhança
119 }
120
121 func g(U [][]float64, m, s float64) [][]float64 { // função G(u; mu, sigma)
      (growth mapping, mapeamento de crescimento)
122   size := len(U) // tamanho da matriz G: tamanho da matriz de potencial U (
        que é a convolução)
123   G := make([][]float64, size)
124   for i := range G {
125     G[i] = make([]float64, size)
126     for j := range G[i] {
127       G[i][j] = 2*(math.Exp(-(math.Pow((U[i][j]-mu), 2)) / (2 * sigma *
            sigma))) - 1 // fórmula da função G(u; mu, sigma) (Chan)
128     }
129   }
130   return G // função g retorna a matriz mapeamento de crescimento G
131 }
132
133 // função principal - a simulação ocorre essencialmente aqui
134 func main() {
135   rand.Seed(time.Now().UnixNano())
136   ic() // ativa a condição inicial
137   op(0, phi) // ativa op para imprimir a condição inicial
138
139   K := Kernel(Nx, Ny, R) // inicia a vizinhança para todas as células
140
141   for i := 0; i < Nx; i++ { // calcula a densidade populacional de cada
        estado
142     for j := 0; j < Ny; j++ {
143       if phi[i][j] > 0.5 { // condição: vivo se > 0,5 e morto se < 0,5
144         dat[1]++
145       } else {
146         dat[0]++
147       }
148     }
149   }
150
151   f, _ := os.Create("datlenia-tcc.dat") // cria o arquivo .dat de densidade
        populacional
152   defer f.Close()
153   fmt.Fprintf(f, "%.2f    %.2f\n", float64(dat[1])/(Nx*Ny), float64(dat[0])/(
        Nx*Ny)) // imprime as densidades em %
154   dat[1] = 0 // zera as densidades para serem atualizadas a cada geração
155   dat[0] = 0
156
157   for t := 1; t <= NG; t++ { // para cada geração
```

```
158     U := convolucao(phi, K, Nx, Ny) // realiza a convolução do estado com a
            vizinhança
159     G := g(U, mu, sigma) // calcula a função G(u; mu, sigma)
160
161 // note que kernel, convolucao e g já se aplicam para toda a rede; não é
        necessário usar laços for externamente
162
163     for i := 0; i < Nx; i++ { // para toda a rede
164       for j := 0; j < Ny; j++ {
165         phi[i][j] += math.Min(math.Max((1/T)*G[i][j], 0), 1) // troca a rede
                inicial para a atualizada, restringindo os valores
166         if phi[i][j] > 0.5 { // atualiza a densidade populacional
167           dat[1]++
168         } else {
169           dat[0]++
170         }
171       }
172     }
173     fmt.Fprintf(f, "%.2f    %.2f\n", float64(dat[1])/(Nx*Ny), float64(dat[0])
            /(Nx*Ny)) // imprime a densidade populacional atualizada em %
174     op(t, phi) // cria o arquivo .dat a cada geração (lenia-tcc-1, lenia-tcc
            -2...)
175     fmt.Printf("%d/%d\n", t, NG) // mostra gerações completas em tempo real
            no terminal
176     dat[1] = 0 // zera as densidades populacionais novamente
177     dat[0] = 0
178   }
179 }
```