UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ESCOLA DE INFORMÁTICA APLICADA

JOGO DA VIDA: CONCEITOS E APLICAÇÕES

LEANDRO ZOUCAS DE LIMA

PEDRO PAULO MARQUES DE OLIVEIRA

Orientadora

FLAVIA MARIA SANTORO

RIO DE JANEIRO, RJ – BRASIL

JUNHO DE 2014

JOGO DA VIDA: CONCEITOS E APLICAÇÕES

LEANDRO ZOUCAS DE LIMA

PEDRO PAULO MARQUES DE OLIVEIRA

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovada por:

_____

FLAVIA MARIA SANTORO (UNIRIO)

_____

KATE CERQUEIRA REVOREDO (UNIRIO)

RIO DE JANEIRO, RJ – BRASIL.

JUNHO DE 2014

Agradecimentos

Este trabalho não seria possível se não fossem todos os professores, servidores e colegas da UNIRIO que tive a honra de conhecer. Mas cabe ressaltar aqui a Professora Flavia que acolheu as ideias loucas e confusas desses dois estudantes e abraçou a causa junto a eles. E à Professora Kate, que se dispôs a ler todas essas páginas!

Preciso registrar a amizade do Pedro Paulo (Acabamos esse trabalho, cara! Acredita nisso?), Sardinha, Alice, Daniel, Marcos, Murilo, Tharik, pessoal da FLUNIRIO (João e Vinicius), Jell, Pedro e melhor parar por aqui antes de cometer mais injustiças.

Agradeço imensamente ao Mps e ao Thiago Moreira, da IBM, que sempre me ajudaram profissional e pessoalmente. Agradeço também a TODOS os verdadeiros e grandes amigos que fiz no CEFET/RJ e na PSCJ/Méier. Amigos de fé! Menciono a Tayza para não haver sangue, mas todos os outros sabem quem são.

Também não posso deixar de citar a namorada Anna Clara, a irmãzinha Bárbara e a minha mãe que aturaram tantos dias de impaciência e tensão por causa do trabalho, e que sempre me apoiaram nos momentos mais complicados. Isso vale pra Deus também. Desculpe-me por me queixar tanto, Deus, e obrigado por TUDO. Você sabe do que estou falando. ;)

Dedico, por fim, este trabalho a duas pessoas: ao meu pai, que foi o maior responsável por eu entrar no curso de Sistemas de Informação da UNIRIO, me orientando nessa escolha e que tinha um orgulho enorme disso. Uma pessoa que de coração tão bom que Deus não podia esperar mais para tê-lo mais próximo. Dedico também à Elise, minha afilhada de quase 2 anos que me mostrou que quando uma célula se desliga, outras nascem e a Vida continua. Assim como no Jogo da Vida.

Leandro Zoucas

Agradecimentos

Agradeço primeiramente a Deus que me iluminou durante esta longa caminhada. Agradeço aos meus pais e a minha irmã que, com muito carinho e apoio não mediram esforços para que eu alcançasse esta etapa da minha vida. Mãe, seu cuidado e dedicação foi que me permitiram seguir em frente. Pai, sua presença significou segurança e certeza de que não estou sozinho nesta caminhada. Irmã, seu companheirismo e apoio também me impulsionaram, não poderia deixar de citar também a Valentina, minha sobrinha que está por vir mas desde já, é um inspiração.

Agradeço aos amigos que fiz neste início de jornada, em especial ao Leandro Zoucas que seguiu comigo, enfrentando as mesmas dificuldades, anseios e receios em relação aos prazos mas, principalmente pela amizade que surgiu ao longo do curso. Agradeço também aos amigos Daniel Gribel, Vinicius Rodrigues e Thiago Sardinha pela amizade, companheirismo, incentivo e apoio.

Agradeço também a todos os professores do curso, que me acompanharam durante a graduação, em especial à Profa. Flávia Santoro que nos orientou no sentido real desta palavra, no desenvolvimento deste trabalho, por sua paciência e por sempre nos receber com um sorriso. Agradeço também à Profa. Kate Revoredo por aceitar fazer parte da nossa banca no último instante e pela paciência. É um prazer tê-la na banca examinadora.

Por último mas, não menos importante, agradeço ao curso de Bacharelado em Sistemas de Informação que forneceu à estrutura necessária para que eu pudesse ampliar meus conhecimentos (percebendo que este é apenas o início da jornada e não o fim) e crescer academicamente.

Pedro Paulo Oliveira

RESUMO

Este Trabalho de Conclusão de Curso procura explicar os principais conceitos relacionados ao Jogo da Vida, bem como de autômatos celulares similares. Exemplifica aplicações destes autônamos, apresenta o estado da arte de forma geral (com algumas das principais descobertas até agora) e apresenta o conceito e um protótipo de um novo jogo, o Warlife, baseado no jogo Immigration que, por sua vez, baseia-se no Jogo da Vida de Conway.

Palavras-chave: Jogo da Vida, Vida, Conway, Autômato Celular, Golly

ABSTRACT

This Final Project intends to explain the main concepts related to the Game of Life and other similar cellular automata. It shows some examples of their applications, presents a bird`s eye view of the state of the art (with the main findings until now) and proposes the concept and a prototype of a new game, the Warlife, an extension of Immigration Game, that was based on Conway`s Game of Life.

Keywords: Game of Life, Life, Conway, Cellular Automaton, Golly

Índice

Índice de Figuras

Capítulo 1 - Introdução

1.1 Motivação

Os autômatos celulares são uma importante ferramenta computacional que começou a ser estudada no final da década de 40 pelo matemático John Von Neumann. Nessa época, o também matemático Stanislaw Ulam sugeriu a Neumann a utilização dos autômatos celulares para a idealização de sistemas biológicos [25], dando maior ênfase para sistemas biológicos auto-reprodutivos ao qual davam o nome de espaços celulares. Em outras palavras, Von Neumann tinha como objetivo de sua investigação o desenvolvimentos de regras matemáticas que simulassem os princípios evolutivos da natureza [6]. Essas regras, que seriam as mesmas para todos os componentes de um determinado sistema, deveriam partir de uma configuração inicial aleatória e cada componente do sistema passaria por uma evolução que sofreria influência direta dos seus vizinhos e do conjunto de regras. Embora essas regras sejam as mesmas para todos os componentes do sistema, a situação dos componentes vizinhos pode variar indefinida e complexamente com o tempo, podendo originar novos sistemas e chegando até a sua autoreprodução.

Os princípios dos autômatos celulares foram desenvolvidos sob a filosofia dos sistemas distribuídos e da Teoria do Caos. Esta última, especialmente, afirma que comportamentos complexos de sistemas podem emergir de ações locais.

O matemático John Conway desenvolveu o chamado Jogo da Vida, o exemplo clássico de autômatos celulares. O Jogo da Vida, ou Life em inglês, foi profundamente estudado por matemáticos e outros cientistas contemporâneos de Conway por iniciativa própria puramente pelo desejo veemente de desenvolver esta teoria. Mais tarde, começou a ser estudado na tentativa de utilizá-lo aplicando suas simulações para os mais diversos fins, desde o estudo de combate aos incêndios florestais [30] até o combate ao câncer [24], passando pela química [1]. Um outro exemplo bastante interessante é o caso de aplicar a abordagem dos autômatos celulares para modelar o processos de urbanização, os resultados obtidos em [20] podem ser gerados por regras de transição definidas localmente. Essas regras podem ser representadas por um conjunto simples de declarações do tipo "*if-else*". Quanto mais se entende sobre um sistema sendo modelado, melhor se pode ajustar as regras para uma representação mais precisa do sistema.

1.2 Objetivos

Este trabalho tem como objetivo discorrer sobre a teoria e aplicações dos autômatos celulares, mais especificamente o Jogo da Vida, bem como definir um sistema (versão do jogo da Vida para 2 pessoas) para auxiliar no ensino do Jogo da Vida (treinar o raciocínio de como o Jogo da Vida se desenvolve) e também que envolva o estudo estratégico de combate, com o auxílio da plataforma de código aberto Golly, que será analisada brevemente no capítulo 4.

1.3 Organização do texto

O presente trabalho está estruturado em capítulos e, além desta introdução, será desenvolvido da seguinte forma:

• Capítulo II: se aprofunda no tema, o Jogo da Vida, contando um pouco de sua história, suas regras, os padrões bem conhecidos que se desenvolvem no jogo e por último conceitua brevemente o comportamento caótico que é uma forte característica do assunto em estudo.

• Capítulo III: examina algumas variações do Jogo da Vida, as respectivas mudanças nas regras dessas variações e apresenta algumas aplicações baseadas em Life.

• Capítulo IV: elabora e desenvolve um protótipo do que seria o Warlife, uma versão estendida do jogo Immigration que é vista no capítulo III. Seu principal objetivo é ajudar no estudo do raciocínio de como o Jogo da Vida se comporta.

• Capítulo V: Conclusões – Reúne as considerações finais, assinala as contribuições da pesquisa e sugere possibilidades de aprofundamento posterior.

Capítulo 2 – Principais Conceitos

2.1 Histórico

Em 1968, o matemático John Conway desenvolveu um importante autômato celular chamado Jogo da Vida, tendo como objetivo projetar um conjunto de regras matemáticas simples capaz de gerar padrões complexos de vida, mostrando como um conjunto de regras básicas pode orientar um sistema complexo. Um fato interessante é que chegou-se as regras por meio do estudo empírico das mesmas, até que se chegasse a um universo onde os autômatos celulares alcançassem um determinado equilíbrio entre nascimento e morte de células [28].

Os atômatos celulares foram inicialmente estudados por John Von Neumann e Stanislaw Ulam na tentativa de modelar sistemas biológicos autorreprodutivos, ou seja, sistemas que se reproduzem fazendo cópias de si mesmos, tais como plantas. Dessa forma, o conceito de autômato celular está fortemente ligado a Von Neumann, que estava interessado nas conexões entre Biologia e a Teoria dos Autômatos. A principal questão abordada por Neumann era: que tipo de organização lógica é suficiente para um autômato ser capaz de reproduzir a si próprio? [25]

O Jogo da Vida em seus primórdios era estudado e desenvolvido de muitas maneiras mas, certamente, o computador não era uma delas. Ao invés disso, geralmente eram usados quadros negros, papel e lápis, tabuleiros de damas e moedas, máquinas de escrever entre outros.

Há relatos em [2] de que por volta dos anos 70, os analistas de sistema interessados no tema, usavam mainframes da IBM para criar programas que pudessem reproduzir computacionalmente o Jogo da Vida. No entanto, como requeria muita capacidade computacional, era deixado executando durante uma noite inteira e era abortado por técnicos operadores que acreditavam ser algum tipo de erro.

Em 1970, um artigo de Martin Gardner na Scientific American sobre o Jogo da Vida de Conway certamente promoveu o jogo.[14]

A simplicidade e imprevisibilidade do jogo no início estimulou dezenas de acadêmicos e os levou a se corresponderem trocando conhecimentos adquiridos independentemente por cada um, sem nenhum tipo de patrocínio, movidos apenas pelo interesse científico. Assim, seguiam criando desafios uns para os outros, resolvendo-os e verificando sua validade. Uma das descobertas a princípio mais interessante foi a existência de padrões que se replicavam indefinidamente, essa descoberta elevou o número de pesquisadores que se correspondiam sobre o tema sem precedentes de maneira a revelar novas descobertas para o próprio Conway [2].

Foi criada então uma publicação trimestral exclusiva sobre a invenção de Conway de nome LIFELINE. Sua assinatura custava um dólar anual e inicialmente contava com 150 clientes. Em fevereiro de 1971, Martin Gardner, um matemático americano, escreveu sobre autômatos celulares apresentando um conhecimento técnico mais aprofundado sobre o qual o Life era baseado. No final deste mesmo ano, Ed Fredkin, um físico americano, levou as simulações do jogo para laboratórios de inteligência artificial que possuíam na época uma grande capacidade de processamento. Gardner comentou que era a primeira vez que presenciava a evolução de padrões do Jogo da Vida de maneira tão rápida, diferente da paginação manual através da saída de mainframes uma página (geração) de cada vez. Fredkin chegou a sugerir que o jogo da vida poderia ser um modelo que descrevia como partículas subatômicas se comportavam [2].

Os entusiastas de Life eram semelhantes a um grupo de taxonomistas, dando nomes aos variados tipos de formas que surgiam a partir das regras do jogo. Interessante notar que isso é justamente o contrário do que ocorre na ciência: geralmente, parte-se de um conjunto de dados e tenta-se determinar quais princípios básicos ou regras geram os resultados. Ao invés disso, os jogadores de Vida já possuíam os princípios básicos definidos por Conway e buscavam determinar qual seria o universo resultante de tais princípios [2].

Em 1973, no final do seu terceiro ano, a LIFELINE parou de ser publicada, pois havia se tornado um fardo muito pesado e consumia muito tempo para continuar devido a prioridades de família, profissionais e pessoais de apenas entusiastas.

O Jogo da Vida trataria de apenas um jogo superficial ou haveria algum real significado por trás de suas regras aparentemente simples? Gardner escreveu o seguinte a respeito da simplicidade encontrada na natureza em relação ao jogo, em um de seus artigos para a *Scientific American*:

“Uma questão intimamente relacionada é se as próprias leis naturais são simples ou complexas. A maioria dos biólogos, particularmente os que trabalham com o cérebro e o sistema nervoso, estão impressionados com a complexidade da vida. Por outro lado, embora a teoria quântica tenha se tornado imensamente mais complicada com a descoberta de novas partículas e suas estranhas interações, a maioria dos físicos mantêm uma forte fé na simplicidade final das leis básicas. Isso era especialmente verdadeiro para Albert Einsten, que escreveu: ‘ Nossa experiência justifica que acreditemos que a natureza é a realização das mais simples ideias matemáticas concebíveis.’”[2]

Mais recentemente, com a maior capacidade de processamento dos computadores, resultados que antes estavam fora do alcance hoje podem ser obtidos. Desta forma, o Jogo da Vida começou a ser estudado na tentativa de utilizá-lo aplicando suas simulações para os mais diversos fins.

Tentativas de mudar as regras estabelecidas inicialmente por Conway foram feitas para tentar se chegar a algum resultado interessante. Inicialmente, nenhuma das tentativas obteve muito sucesso. Esses autômatos celulares foram chamados de *Life-Like*. Alguns possuem resultados semelhantes aos de *Life* e outros resultados curiosos mas, nenhum foi estudado tanto como como o *Life* original.

2.2 As Regras do Jogo

Os autômatos celulares podem ser construídos em uma, duas, três ou mais dimensões mas, apenas os construídos em duas dimensões estão no escopo deste trabalho. As regras do Jogo da Vida foram definidas cuidadosamente através de inúmeros experimentos. Tais regras definem o estado de uma determina célula qualquer na próxima iteração com base no estado atual das células vizinhas. No caso do Jogo da Vida, os vizinhos de uma célula são todas as 8 células que a circundam (como o grid é teoricamente infinito, as regras se aplicam analogamente às células na "borda" da tela ou do papel onde Vida está representado como mostra a Figura 1) [2].

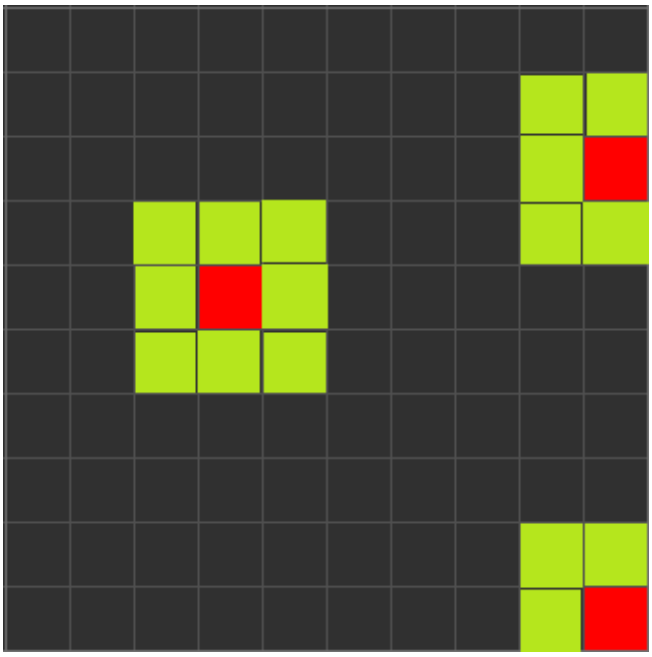

Figura 1 - Células aleatórias em vermelho e seus vizinhos em verde.

Algumas regras causaram a morte das células precocemente, outras causavam o nascimento excessivo de células. Life busca o equilíbrio entre estas tendências, tornando difícil predizer se

determinada estrutura vai desaparecer totalmente ou crescer indefinidamente. Desta forma, chegou-se às seguintes regras [2]:

1-Uma célula morta com exatamente três vizinhos vivos se torna viva (nasce, vide Figura 2).

2- Uma célula viva que tenha de dois a três vizinhos vivos, permanece viva (sobrevivência, Figura 3).

3- Em todos os outros casos, as células morrem ou permanecem mortas (superpopulação ou solidão, como na Figura 4).

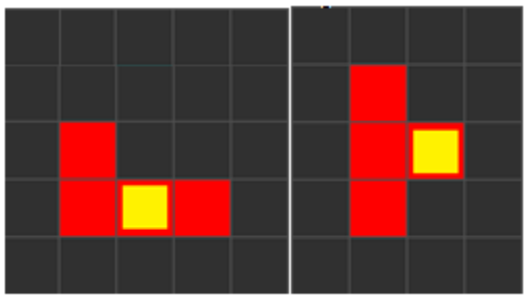

Figura 2 - Células atualmente mortas em amarelo nascerão por conta das células atualmente vivas, em vermelho

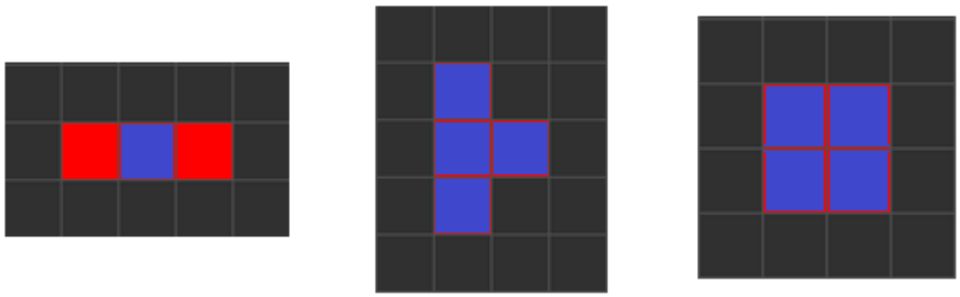

Figura 3- Células azuis vivas permanecerão vivas por conta de suas vizinhas. As células vermelhas morrerão.

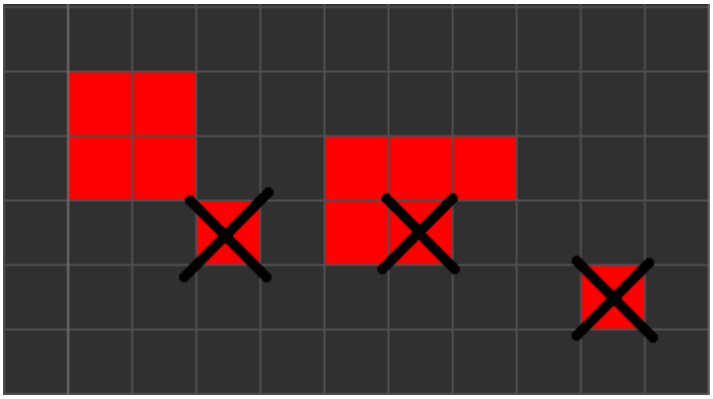

Figura 4 - Células riscadas morrerão por terem nenhum, 1 ou mais de 3 células vizinhas

Inicialmente é definido um determinado agrupamento de células e as regras acima são aplicadas em períodos de tempo chamados de gerações (iterações). Cada célula viva representa uma unidade da população. Nota-se claramente que o tempo no Jogo da Vida passa, discretamente, em forma de iterações, que são chamadas de gerações.

É por conta destas regras que dizemos que o Jogo da Vida de Conway é um modelo ecológico, onde a distribuição espacial e de frequência dos agentes dependem de regras que governam a sobrevivência e a reprodução. São também chamados de modelos evolutivos, embora o uso deste último termo seja um pouco mais delicado (poderiam surgir agentes completamente diferentes e inesperados ao longo do jogo?)

Esta é uma forma muito eficaz de observar como estruturas, padrões e comportamentos podem evoluir e tornar-se complexos, a partir de regras simples. Um dos aspectos que mais chamam a atenção é o fato de cada célula não depender somente de suas propriedades, mas depender de um conjunto de outras células. Assim como na Teoria dos Jogos, cada ação é definida através de um conjunto de outros indivíduos, que geram padrões no macro [8].

No Jogo da Vida assim, como na natureza, diversos fenômenos são observados. Um fenômeno recorrente é o da "migração" dos autômatos. Na verdade, não ocorre uma migração de fato, são os autômatos que ao agirem de maneira estabelecida pelas regras morrem em uma determinada célula mas uma de suas vizinhas nasce. Assim, ao observarmos este fato por um determinado período de tempo, temos a impressão de que está ocorrendo uma "migração" dos autômatos. Contudo, este fenômeno é semelhante à migração de populações por conta de escassez de recursos, o que nos confere conhecimento para estimar o que poderia acontecer com populações sob determinadas condições.

Todavia, a natureza é mais complicada, além de não temos certezas quanto às suas regras. Já o jogo da Vida nos permite observar sistemas os quais temos conhecimentos de suas regras. Mas assim como são estudadas formas mais simples de vida animal (como minhocas), para descobrir coisas sobre formas de vida animal mais complexas (como humanos), pesquisadores podem estudar o jogo da vida para aprender comportamentos e padrões em sistemas mais complexos [8].

As regras descritas acima são tudo o que é necessário saber sobre o jogo para fazer alguma descoberta, diferente de outros jogos nos quais programadores criam um conjunto elaborado de cenários possíveis. Em Vida, as referidas regras por si só criam padrões. Desta forma, temos a extrema necessidade de adequar muito bem as regras dependendo do fenômeno a ser estudado.

É importante ressaltar desde já que as aplicações práticas do jogo são muitas: da criptografia à geração de números pseudo-aleatórios, da análise combinatória de jogos ao estudo de redes, e até em entretenimento, passando pela composição musical. Podemos pensar que qualquer fenômeno espacial no decorrer do tempo pode ser aparentemente modelado em autômatos celulares como o Jogo da Vida. Biólogos, economistas, estatísticos e artistas se interessam pelo jogo. No entanto, neste trabalho, procuraremos focar nas aplicações do Jogo da Vida (e algumas outras versões) no campo da computação, matemática e Sistemas de Informação.

2.3 Formas Bem Conhecidas

Cada agrupamento de células vivas definido no grid pode ser chamado de desenho, imagem, figura etc. Alguns desenhos são bem conhecidos e bem documentados, portanto sabe-se que algumas imagens iniciais tornam-se estáveis depois de um determinado número de gerações. Conway as chamava de vida parada Outras se tornam oscilantes indeterminadamente. Um fato interessante é que imagens sem simetria inicial tendem a se tornar simétricas. Uma vez que isso ocorre, a simetria não é perdida embora possa aumentar em riqueza [8].

Uma questão interessante levantada por Conway é se haveria alguma imagem que cresceria indefinidamente. A primeira imagem desse tipo foi encontrada por William Gosper em novembro de 1970 [2].

“Um desafio proposto por Conway era criar uma configuração que geraria crescente quantidade de células indefinidamente. Este desafio foi resolvido por Gosper em 1970 - quando tempo computacional era caro e os computadores eram muito lentos comparados aos padrões atuais. Ele desenvolveu uma forma que cuspia continuamente fluxos de gliders - um glider gun, assim por dizer.

Curiosamente, sua configuração do gun glider não foi exibida como pequenos quadrados agradáveis mas ao invés disso como uma saída primitiva de uma máquina de escrever (figura 5); isso enfatiza os limitados recursos disponíveis em 1970 para a busca de estruturas tão complexas. Rapidamente uma indústria caseira se desenvolveu - todos os tipos de intrincadas configurações iniciais foram descobertas e exploradas" (página 4 de [2]).

Os primeiros resultados acerca de formas bem conhecidas e seus padrões foram obtidos pelo próprio Conway. O mais interessante é que esses primeiros resultados não foram alcançados por meio de processamento computacional, mas ao invés disso, as chamadas vidas paradas e os osciladores mais simples foram descobertos por meio do desenvolvimento de várias pequenas configurações com o auxílio de folhas de gráfico, quadros negros, tabuleiros e peças e, claro, as regras. Desta maneira, durante as pesquisas iniciais, Conway descobriu que o R-pentomino, imagem que veremos a seguir, não se estabiliza em um número pequeno de gerações [17].

O R-pentomino começa com apenas 5 células, mas se torna complexo em poucas gerações, foi o primeiro desenho que desafiou Conway em suas tentativas de simulação manual. Alguns dos primeiros programas desenvolvidos para rodar o jogo foram escritos para determinar o destino deste pequeno padrão. Esse era um problema para muitos computadores da época mas os atuais podem rodar a sequência completa diversas vezes em um segundo [8]. A Figura 6 representa o R-pentonino.

O R-pentomino leva cerca de 1103 gerações para estabilizar com uma população de 116 células vivas. Isso explica porque Conway se viu impossibilitado de chegar a este resultado manualmente (mas conseguiu rastreá-lo até a geração 460) [8].

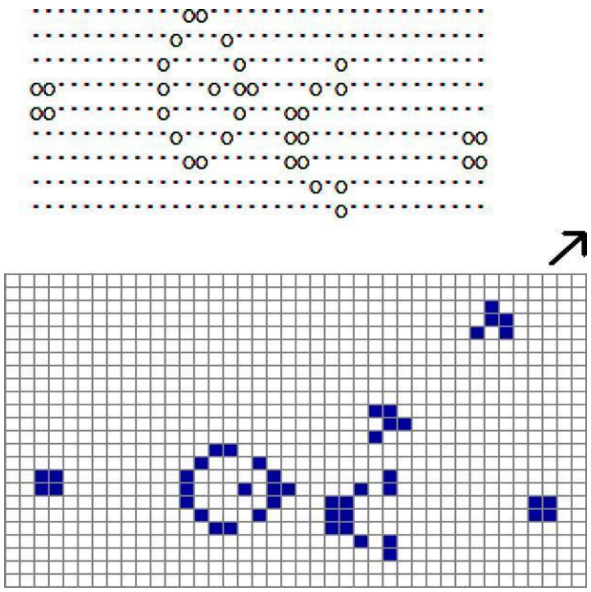

Figura 5 - A máquina de escrever e o autômato celular

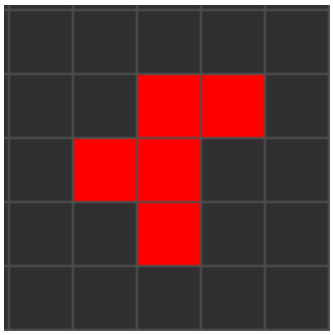

Figura 6 - O R-pentonino

2.3.1 Still Life Objects

Algumas formas ou objetos de comportamento bem definido surgem em Vida. Alguns relativamente comuns são chamados de still life (vida parada). Recebem este nome por permanecerem inalterados geração após geração, contanto que não haja interferência em sua vizinhança. Nesses casos, nenhuma célula morre e nenhuma célula nasce [2]. Alguns desses objetos surgem no decorrer das gerações seguintes a um R-pentomino.

As condições para que tenhamos um objeto desta natureza são que todas as células devem ter no máximo 2 ou 3 células vizinhas vivas e toda célula morta deve ter não mais que 2 células vizinhas vivas [2]. Objetos que se encontram nestas exatas condições serão Still Life. Desta forma, podemos notar que estas condições atendem às regras definidas anteriormente, isto é, são as mesmas mas, estes são requisitos para o surgimento de tais objetos.

A figura mais comum que se encaixa neste perfil é o quadrado, que é simplesmente um quadrado 2x2 de células vivas, como na Figura 7. O quadrado permanece inalterado ao longo do tempo uma vez que cada célula possui exatamente 3 células vizinhas vivas.

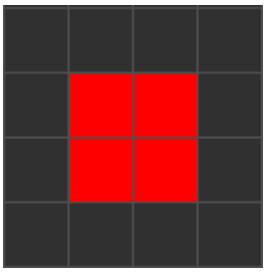

Figura 7- Um quadrado 2x2 de células vivas permanece inalterado ao longo do tempo,

Além disso, na Figura 7, podemos ver que nenhuma célula morta tem mais que 2 vizinhos vivos, atendendo assim às condições colocadas anteriormente.

Algumas outras formas do tipo Still Life conhecidas são a beehive, boat, ship e float. Elas estão na Figura 8 [8].

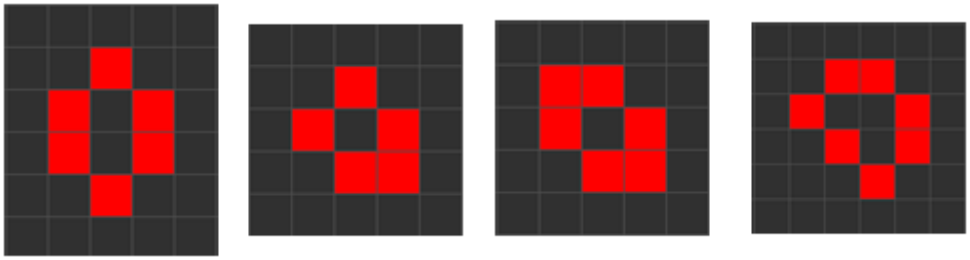

Figura 8- 4 Formas Still Life na seguinte ordem: beehive, boat, ship e loaf

No R-pentomino, podemos ver muitos Still Lifes [8].

A Figura 9 confirma os padrões de simetria citados anteriormente.

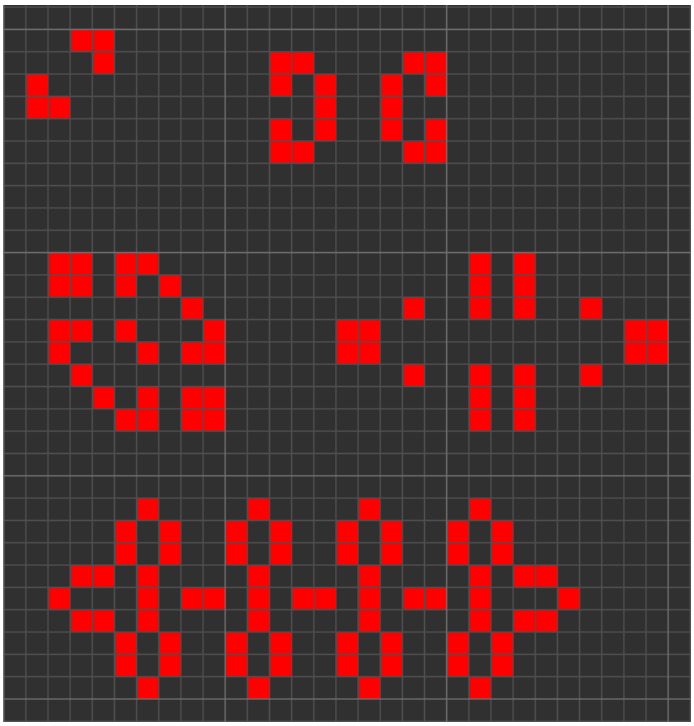

Figura 9 - Simetria em Still Lifes (página 21 de [2])

2.3.2 Oscillators

Osciladores são objetos que mudam de geração para geração repetindo o padrão em ciclos, alternadamente. Os mais simples são aqueles que têm o ciclo de duas gerações, onde o mais comum é o chamado blinker (Figura 10), que possui três células. O blinker se alterna entre três células na vertical e na horizontal a cada período. Novamente no R-pentomino, podemos ver este padrão [8].

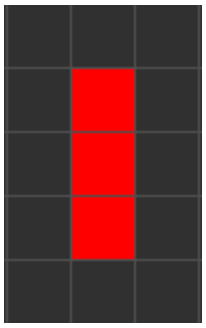

Figura 10 - Blinker, na posição vertical

Na próxima geração, as células Norte e Sul da central morrerão, enquanto as células Oeste e Leste nascerão. A célula central permanecerá viva.

Outro oscilador comum com ciclo de 2 gerações é o toad (Figura 11) [8]:

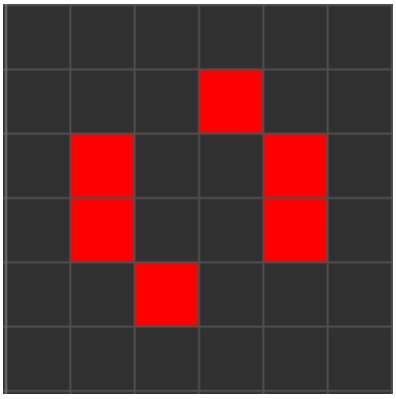

Figura 11 - O oscilador Toad

2.3.3 Gliders

Após observarmos por exemplo, o R-pentomino, podemos notar alguns padrões que se movem, ou seja, deslizam (*glide*) pelo grid, como na Figura 12. O padrão se desloca em diagonal retomando o padrão inicial após algumas gerações, este comportamento recebe o nome Geométrico de "glide reflection", daí o nome desta "classe". Como mencionado anteriormente, são padrões que parecem representar migrações populacionais. Esta foi uma das descobertas mais notáveis no que se refere às pequisas iniciais sobre o jogo. Pois logo após sua descoberta, Conway percebeu que o glider poderia ser o substituto do conceito de fio de Von Neumann como mecanismo de transmissão de informação de uma região para outra em Life [2].

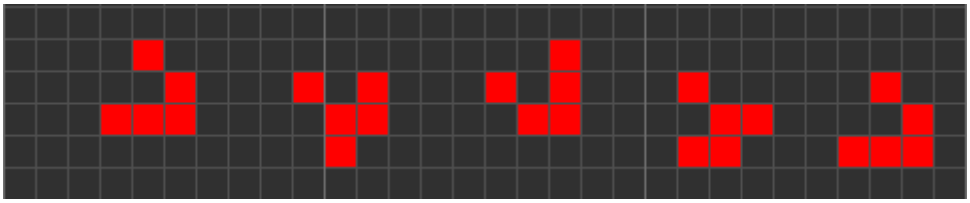

Figura 12- Ciclo de evolução do glider lado a lado (página 22 de [2])

Para esse glider, especificamente, após 4 gerações, a figura retoma sua forma original porém, deslocada em relação à diagonal. A partir da descoberta do glider, iniciou-se uma busca por outras configurações que se movimentassem, objetos desse tipo costumam ser chamados também de "spaceships" ou naves espaciais, assim foram encontradas as três naves espaciais primitivas que podem ser vistas na figura 13 da seção 2.3.4 [2]. É extremamente interessante atentar para o detalhe de que as regras não falam absolutamente nada em relação a objetos que se movem e, apesar disso, padrões de movimentação simplesmente surgem. Esse fenômeno de objetos em movimento é uma ótima demonstração de como padrões complexos podem surgir a partir de regras simples.

2.3.4 Outros objetos interessantes

As naves espaciais ortogonais que se deslocam para a direita, esquerda, para cima ou para baixo, diferentemente dos gliders que se movem diagonalmente. Estes padrões são muito menos comuns que os

gliders mas, importante em certos aspectos. Eles possuem 3 tamanhos diferentes (light, médium e heavy weight, do menor para o maior respectivamente), como mostram as Figuras 13 e 14:

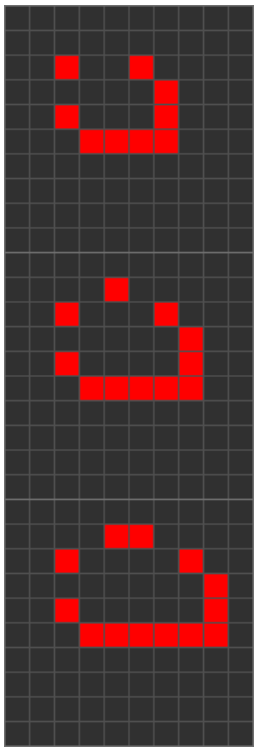

Figura 13 - As naves light, medium e heavy weight, respectivamente

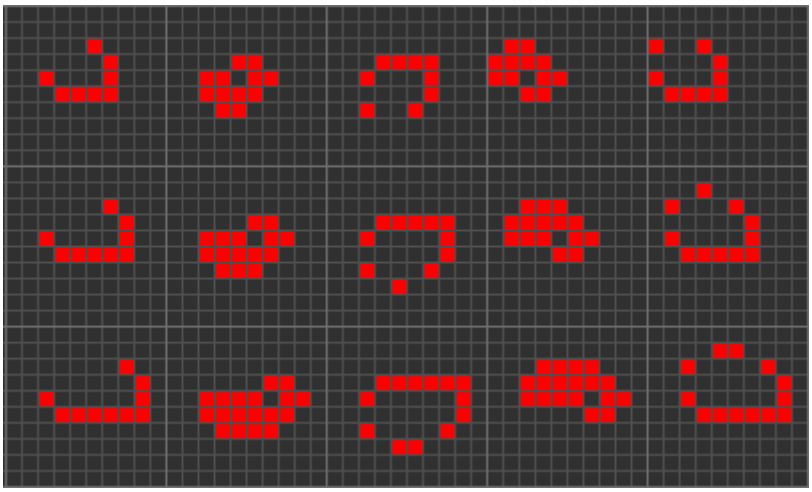

Figura 14 - As naves light, medium e heavy weight ao longo do tempo

Por último, 2 padrões que começam simples e evoluem para osciladores: uma linha de 10 células vivas (figura 15) se torna um oscilador de ciclos de 15 gerações chamado pentadecathlon:

Figura 15 - Linha de 10 células vivas

O padrão da Figura 16 se torna um oscilador de ciclo de 3 gerações chamado pulsar:

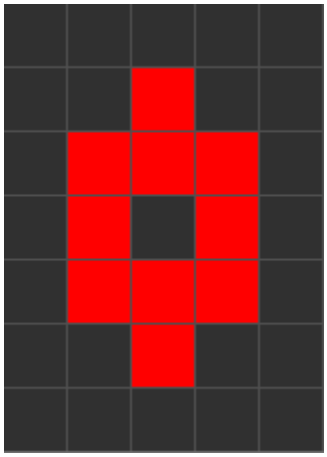

Figura 16 - O oscilador Pulsar

Dizemos que um padrão se estabilizou quando a população parou de crescer e/ou diminuir. Por exemplo, o R-pentomino estabiliza na geração 1103 [36] porque neste momento consiste de gliders e osciladores e fica claro que os gliders nunca perturbarão os osciladores por conta de uma colisão ou mesmo outros gliders. Uma pergunta interessante então a ser feita é: será que eventualmente todos os padrões se estabilizam? Como citado anteriormente, foi descoberta uma imagem que incluía guns (figuras que geram um fluxo sem fim de outras imagens, algo como se fosse uma arma disparando) que eram estacionárias e disparavam gliders ou outras naves espaciais, "puffers" que se movem deixando uma "trilha de fumaça" para trás e "rakes" que se movem e emitem naves espaciais.

Desde então, foram construídas até mesmo portas lógicas por meio de fluxos de gliders e outras naves espaciais que enviam informações que podem ser interpretados como sinais elétricos são interpretados por computadores. Apesar de não ser muito prático construir computadores desta forma, qualquer programa poderia ser executado numa máquina deste tipo. Na verdade, alguns computadores de interesse especial chegaram a ser construídos nestes moldes, incluindo um que gerava números primos e outro que emulava o jogo Vida numa escala muito maior e a passos lentos.

2.4 Teoria do Caos

Um último conceito relacionado ao Jogo da Vida a ser definido ainda neste capítulo é o comportamento caótico, que é notável nos autômatos celulares e, inclusive, é o responsável por torná-los tão interessantes. Apesar de conceitos relacionados à Teoria do Caos já aparecerem desde o final do século XIX, ela ficou muito famosa a partir do trabalho de Lorenz, que estudava a previsão do tempo [21]. De forma resumida, Lorenz definiu o caos: onde o presente determina o futuro, mas a aproximação do presente não determina aproximadamente o futuro [10].

O "efeito borboleta", como ficou conhecido (ver Figura 17), é o paradigma que ilustra essa situação. O simples bater de asas de uma borboleta no Brasil poderia causar um tornado no Texas? Para prever o tempo, teríamos que conhecer a quantidade e a posição exata de todas as borboletas do mundo? [3]

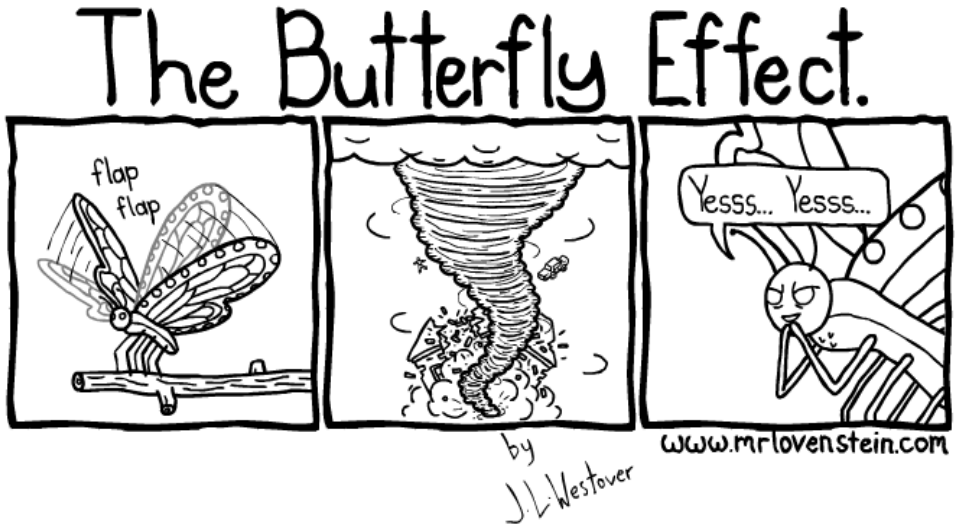

Figura 17 - Tirinha ilustrando o Efeito-Borboleta, como ficou conhecido (Retirada em 27 de maio de 2014 em http://www.mrlovenstein.com/comic/50)

É particularmente interessante mencionar neste trabalho (da área de computação) o “incidente” que Lorenz sofreu. A maçã de Newton de Lorenz caiu quando ele executou pela segunda vez um programa relativo à previsão do tempo. A segunda execução retornou resultados extremamente discrepantes em relação ao primeiro. Percebeu-se que isso se devia ao truncamento dos valores de entrada: 0.506127 foi inputado como 0.506 em uma das vezes. Como uma diferença tão pequena gerou resultados tão diferentes?

O Caos é muito presente tanto na natureza como na matemática, e é aplicável em diversas áreas de estudo, como na meteorologia, biologia, física, economia, engenharia e até filosofia, entre outros campos.

E o que o caos tem a ver com os autômatos celulares? Os ACs possuem esse comportamento caótico, mesmo com as regras sendo determinísticas (sem o fator “aleatório”, e conhecendo todas as informações do universo). Um autômato celular, com uma determinada entrada inicial de dados (estados das células) pode evoluir de maneira completamente diferente de uma outra entrada praticamente idêntica. E é aí que os ACs tornam-se interessantes para diversas aplicações: desde a simulação espacial de fenômenos com comportamentos caóticos, até o estudo da própria teoria do caos, passando por geração de números aleatórios (com sementes parecidas, gera-se números aleatórios bem diferentes) e criptografia.

Capítulo 3 - Cenários de Aplicação do Jogo da Vida

3.1 Exemplos de Aplicações do Jogo da Vida

O comportamento de células animais pode ser melhor entendido usando regras simples. Comportamentos que parecem inteligentes, como os que vemos em colônias de formigas provavelmente podem ser determinados por regras simples que ainda não sabemos quais seriam [8].

Problemas relacionados a tráfego podem ser resolvidos por meio de ferramentas matemáticas modeladas com o auxílio desse tipo de simulação.

Vírus de computadores também são exemplos de autômatos celulares. É possível que a resposta para achar a "cura" para esses vírus esteja escondida entre padrões que podem ser simulados por jogos deste tipo [38].

Da mesma forma, encontraríamos cura para doenças humanas muito mais rapidamente e facilmente se entendêssemos os mecanismos relacionados ao nascimento e morte das células causadoras da doença.

Por ser uma área ainda muito recente em comparação com outras ciências, constantemente se descobrem novos campos potencialmente suportados por autômatos celulares, uns mais e outros menos parecidos com o clássico Jogo da Vida. Apresentamos a seguir algumas das principais aplicações dos ACs até o momento. É importante frisar, no entanto, que existem muitas outras áreas onde os ACs são aplicáveis, como a química [27], economia, biologia, matemática, computação além de, claro, simulações das mais diversas.

Entre outras aplicações de grande interesse está a exploração de galáxias, já que seria muito mais fácil explorá-la com máquinas autorreplicantes, embora seja teoricamente possível construí-las, inclusive o entendimento para este feito seria facilitado com auxílio de simulações deste tipo, ainda não existe nenhum trabalho significativo na prática [8].

3.1.1 Geração de Números Pseudo-aleatórios

Como já vimos, o comportamento caótico (e logo, imprevisível) dos autômatos celulares a partir de dados iniciais conhecidos e regras bem definidas (determinísticas) torna os ACs interessantes para algumas aplicações. Uma das mais intuitivas seria a geração de números pseudo-aleatórios.

O estudo feito em [41] mostra os resultados de diversos testes feitos em ACs. Estes ACs são de duas dimensões e cada célula tem 2 estados possíveis, mas não são Life-like porque, dentre outros fatores, utilizam a Vizinhança de Von Neumann (14) (4 vizinhos N, S, W, E para cada célula) ao invés da vizinhança de Moore [22], com 8 vizinhos (N, NW, W, SW, S, SE, E, NE para cada célula). Além disso, são usados ACs não uniformes (ou seja, com diferentes regras aplicadas em diferentes células num mesmo grid).

Segundo o próprio [41], existem diversos métodos para a geração do número pseudo-aleatório baseado em autômato celular de duas dimensões. A ideia geral do algoritmo proposto por ele é a seguinte: O AC, inicializado randomicamente, evolui por 4 gerações. Nessas 4 gerações, cada célula gera 4 bits, que são tratados juntos como um hexadecimal. Dessa forma, a cada 4 gerações, um grid de 8 x 8 geraria 64 dígitos hexadecimais.

Considere o AC aleatório da figura 18 como inicial.

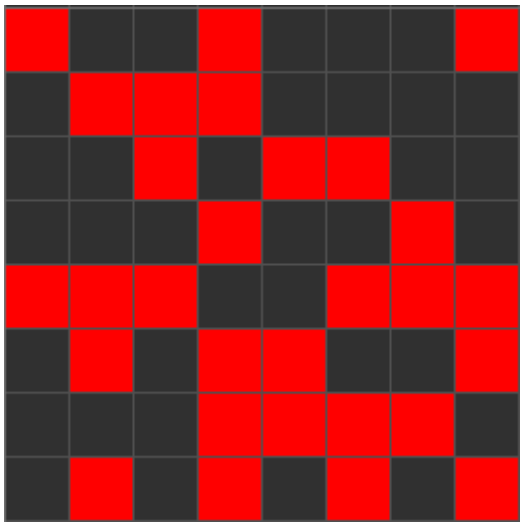

Figura 18 – Um AC inicializado aleatoriamente que será usado como exemplo de semente no algoritmo estudado.

Na figura 18, temos um AC de 2 estados (1 e 0, vivo e morto respectivamente) com grid 8x8. Podemos traduzir o AC numa concatenação de seus valores, considerando a leitura do grid da posição [0,0]...[0,7],[1,0]...[1,7],...,[7,7] (ou seja, da esquerda para direita, de cima para baixo). No caso da instância da figura bla1, teríamos: 1001000101110000001011000001001011100111010110010001111001010101 (64 dígitos, correspondendo às 64 casas do grid 8x8.

O algoritmo propõe a separação deste número enorme em termos de 4 dígitos.

1001 (termo1), 0001 (termo2), 0111 (termo3), 0000 (termo4),...,0101 (termo 16).

Converte-se cada termo em binário por seu respectivo correspondente em hexadecimal. Neste exemplo, teríamos 91702C12E7691E55.

Após esta conversão, o grid inicial (da figura bla1) seguiria para a próxima geração obedecendo às regras do Jogo da Vida. Chegamos então a mais um número hexadecimal de 16 dígitos obtidos como no exemplo anterior. Ao final de 4 gerações, teremos 4 números hexadecimais de 16 dígitos, então simplesmente justapomos esses 4 números obtendo assim um número hexadecimal de 64 dígitos.

Como descrito em [41], opta-se por um método baseado na simplicidade para chegarmos ao número pseudo-aleatório.

Uma abordagem parecida foi usada em [19], com autômatos celulares Life-like. Os estados das células em bits também foram concatenados compondo blocos, que servem de entrada para algumas funções específicas, gerando números pseudo-aleatórios de alta qualidade.

3.1.2 Geração de Primos com Jogo da Vida

Em 1991, Dean Hickerson desenvolveu um padrão no próprio Jogo da Vida muito interessante. Mais do que a própria aplicabilidade do padrão, mostra o "poder" dos autômatos celulares, ainda mais de um tão simples como o Jogo da Vida. Primers [18], como ficou conhecido, é um padrão que, inicialmente, calcula os números primos.

Light-weight spaceships são disparados no sentido Oeste e alguns são destruídos por guns de gliders. A cada geração 120N+100, a nave da vez somente "sobrevive" e passa do último obstáculo (um pentadecathlon) se e somente se N for primo[1].

Um esquema pode ser visto na figura 19 (embora o ideal seja acompanhar o jogo em movimento. A figura 19 mostra o padrão PRIMERS por volta da geração 2140. Note que a última nave a se aproximar do pentadecathlon não foi destruída. Ela representa o número 17, porque estamos na geração 120 * 17 + 100. Como essa nave conseguiu passar de todos os obstáculos, significa que o número N que ela representa (17) é primo, assim como passaram as naves nas gerações (120 * 13 + 100), (120*11+100) entre outras.

[1] Embora algumas referências, como [18], digam que o teste deve ser feito a cada geração 120N+10, após alguns testes, os autores deste trabalho verificaram resultados positivos considerando a geração 120N+100, como consta na referência [29].

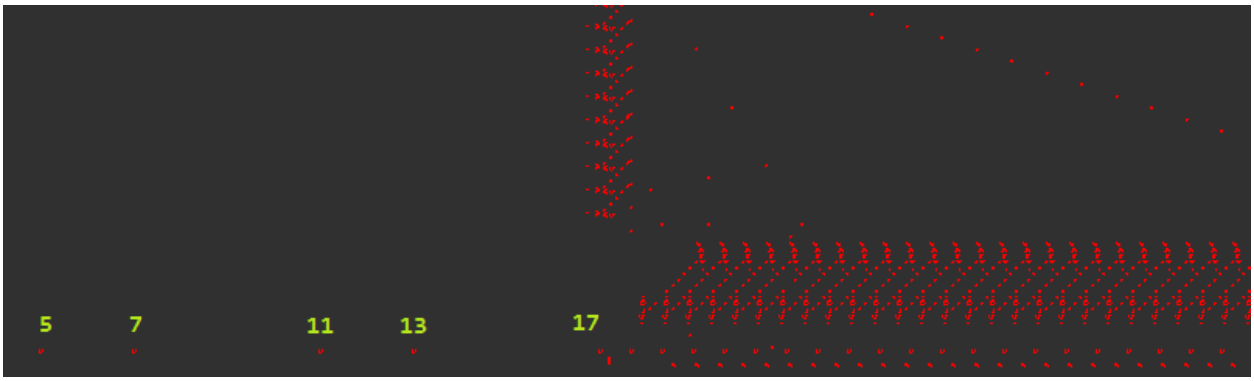

Figura 19 - As naves que passaram pelo pentadecathlon representam os primos.

O código do arquivo PRIMER-LIFE.rle, retirado de [31], está anexado no final deste trabalho.

Obviamente que esse não é o melhor algoritmo para determinar primos, principalmente os grandes, apesar da ideia por trás de PRIMER ser semelhante ao algoritmo conhecido como Crivo de Eratosthenes [12]. Além disso, outros padrões foram desenvolvidos com o mesmo intuito e mais eficientes posteriormente [18], como o de Jason Summers, em 2010 [15], ou a calculadora de números primos gêmeos do próprio Hickerson em 1994 [42].

3.2 Life-Likes

Para falarmos sobre algumas aplicações práticas de autômatos celulares, vamos definir essa classe de autômatos que são muito parecidos com o Jogo da Vida, que aqui chamaremos de Autômatos Celulares (AC) Life-Like.

Segundo [22], Autômato Celular é um sistema discreto e dinâmico definido num espaço também discreto que segue regras locais próprias, e onde cada unidade depende do estado de seus vizinhos, que por sua vez regem como o AC vai evoluir ao longo do tempo.

Existem diversas formas de definirmos um AC na linguagem matemática, mas que fogem ao escopo deste trabalho. Baseados na definição formal em [22], dizemos que um AC pode ser descrito com 6 informações:

G: um espaço discreto de n dimensões (Vida, de Conway, tem 2 dimensões) com um conjunto c de células.

E: um conjunto de k estados, geralmente K ∈ □

e: uma função de saída que mapeia o estado corrente de uma célula c□ em c no tempo discreto t. Matematicamente:

e : G X □ → E e a chamada da função poderia ser e(c□, t).

e0: uma função inicial que aloca os estados iniciais de cada célula em G.

v: a função vizinhança que define quem são as células vizinhas de cada célula c□.

d: função de transição que descreve as regras que governam a dinâmica de cada célula c□.

Uma forma bastante conhecida para definirmos a regra de um determinado Autômato Celular é utilizar a notação do software Golly[2]. Uma regra segue o seguinte padrão, basicamente:

Bx,y...z/Sx´y´...z´

Onde os algarismos imediatamente seguintes à letra B (no caso, representados po x,y,...z) referem-se a quantas células vivas vizinhas são necessárias para que uma célula atualmente morta nasça na geração seguinte; e os algarismos imediatamente depois da letra S (no caso, x´, y´,...,z´) referem-se a quantas células vivas vizinhas são necessárias para que uma célula viva sobreviva na geração seguinte. Intuitivamente, podemos notar que as letras B referem-se a *birth* e S, *survival* (nascimento e sobrevivência em inglês, respectivamente.

Como exemplo, pensemos no próprio Jogo da Vida de Conway. Ele segue a regraa: B3/S23, pois uma célula precisa ter exatos 3 vizinhos vivos para nascer e possuir 2 ou 3 vizinhos vivos para sobreviver (logicamente, são 2 ou 3 vizinhos vivos, e não 23, o que nem seria possível se pensarmos que cada célula tem 9 vizinhos no Jogo na Vida). Qualquer outra combinação de vizinhos vivos resulta na morte (ou não nascimento) da célula.

Podemos exemplificar a definição acima definindo uma classe de ACs bastante interessantes por serem muito parecidas com o clássico Jogo da Vida de Conway. Neste trabalho, vamos chamar a família de ACs que segue essas regras de “Life-Like”, e seria descrita de acordo com [22] deste modo:

G: Espaço de duas dimensões (Z^2)

[2] Golly é um software livre que funciona como uma plataforma de aplicação para explorar o Jogo da Vida de Conway e outros autômatos celulares. Inicialmente desenvolvida por Andrew Trevorrow e Tom Rokicki, encontra-se disponível em http://golly.sourceforge.net/ em 27 de maio de 2014.

E: Dois estados possíveis (booleano, 1 ou 0, verdadeiro ou falso…)

Um e qualquer;

um e0 qualquer, com e ϵ E, e0 ϵ E.

v: Vizinhança de Moore [22], onde os vizinhos de uma determinada célula são as 8 células circundantes adjacentes, considerando os cantos.

d: As regras como as já descritas (dependendo do número de vizinhos de cada estado)

Os Autômatos Celulares, especialmente os Life-Like formam uma classe simples, porém poderosa, além de ser a forma clássica de um autômato celular, a partir de onde toda essa área de estudo se iniciou.

Desde a criação do Jogo da Vida por Conway, diversos outros jogos sobre autômatos celulares foram desenvolvidos, cada um seguindo diferentes regras relacionadas à morte e vida das células.

Para cada regra Bx,y...z/Sx´y´...z´ associada a um autômato celular, podemos perceber diferentes comportamentos, alguns mais estáveis como o Jogo da Vida de Conway mas, outros que levam a explosões de crescimento de células pelo grid bastante instáveis e outros ainda que assumem comportamento caótico.

A seguir, serão exemplificados alguns autômatos life-like.

3.2.1 High Life

Uma das variantes notáveis do jogo de Conway é o chamado HighLife, concebido por Nathan Thompson em 1994, suas regras podem ser expressas pela notação vista anteriormente, por B36/S23. Por ter se originado a partir do Jogo da Vida, muitos de seus padrões mais simples são semelhantes como osciladores, beehive, blinkers, gliders, contudo, quando os padrões começam a ficar mais complicados já não são encontradas mais semelhanças.

Um dos interesses especiais, gerados pelo HighLife vem do padrão chamado replicador que é uma imagem que após chegar em um estado final bem conhecido entra em loop, adotando comportamento de se autoclonar em determinados períodos de tempo. Um comportamento como esse não foi encontrado no Jogo da Vida [4].

O comportamento replicante tem seu início após 12 gerações, então reproduz outras duas cópias após a 24ª geração, seguida por mais quatro clones após a 48ª geração e assim por diante. Generalizando temos 2^n cópias do replicador na geração 12(2^n -1) [4]. Veja um resumo na figura 20.

Uma das questões que despertam grande interesse é o fato de que os replicadores parecem emergir de modelos populacionais não lineares, ao invés disso surgem espontaneamente de condições iniciais de desordem, o que levanta a possibilidade de haver uma explicação mais profunda para este fato.

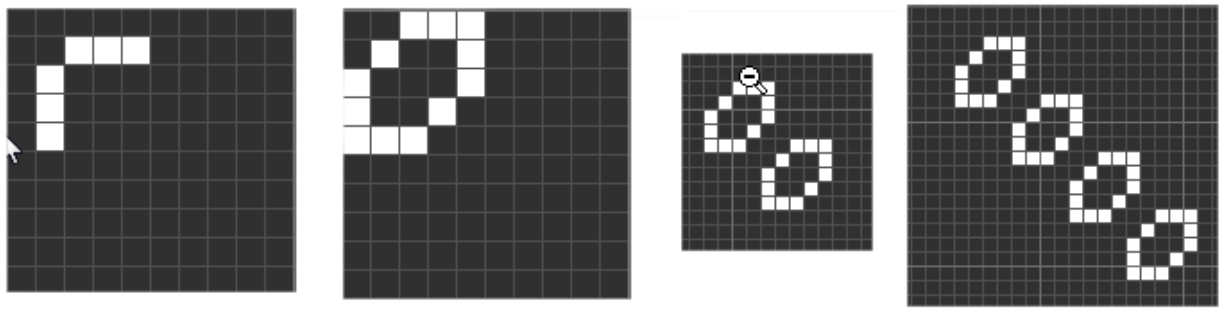

Figura 20 - O replicante; o mesmo replicante após 3 gerações; o mesmo após mais 12 gerações e após 36 gerações.

Interações entre replicadores e outros padrões também produzem resultados interessantes e não necessariamente culminam com a destruição dos replicadores, ao invés disso, podem gerar novos padrões. Um exemplo disso é o chamado Bomber, um padrão do tipo spaceshipe descoberto por Nathan Thompson que é formado a partir de um blinker e um replicador. Assim, colocando-se um blinker no caminho de um replicador há uma reação que dá origem ao bomber como podemos ver na Figura 21.

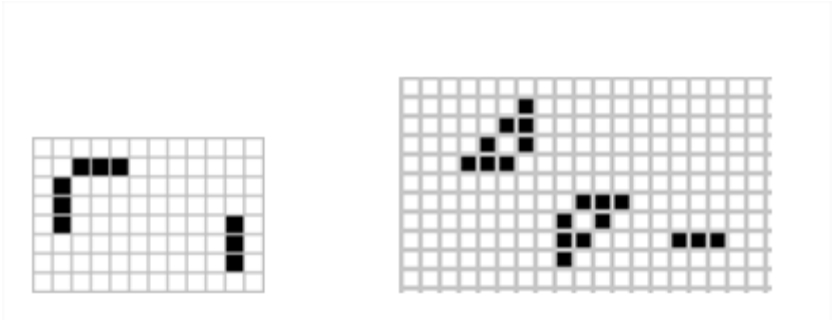

Figura 21- O replicador em formação e um blinker e, mais tarde, o bomber formado.

Dean Hickerson resolveu parcialmente problemas relacionados ao desenho de padrões resultantes de “colisões” entre replicadores e outros padrões bem conhecidos criando um programa de busca dessas configurações chamado random torus, mas seus métodos podem ser aplicados a apenas um subconjunto de problemas desse tipo [4].

3.2.2 Seed

Outro Life-Like com regras no mínimo interessantes é o Seed que pode ser descrito por B2/S, ou seja, segundo suas regras uma célula precisa de duas células vizinhas para nascer mas, nunca permanece viva. Apesar de parecer inusitada essa regra é relativamente comum nas variantes do Jogo da Vida e em sua terminologia as variantes os Life-Like deste tipo recebem o nome de phoenix ou fênix.

Inicialmente investigado por Brian Silverman e nomeado por Mirek Wójtowicz, ainda que as células vivas morram constantemente, como os requisitos para que uma célula nasça são mínimos; uma configuração com poucas células dá origem a uma espécie de desenho dinâmico que se assemelha a uma explosão caótica que se projeta de maneira a cobrir todo o grid. Apesar da aparente desordem são encontrados padrões no Seed, como os gliders da Figura 22 e Figura 23 [13]:

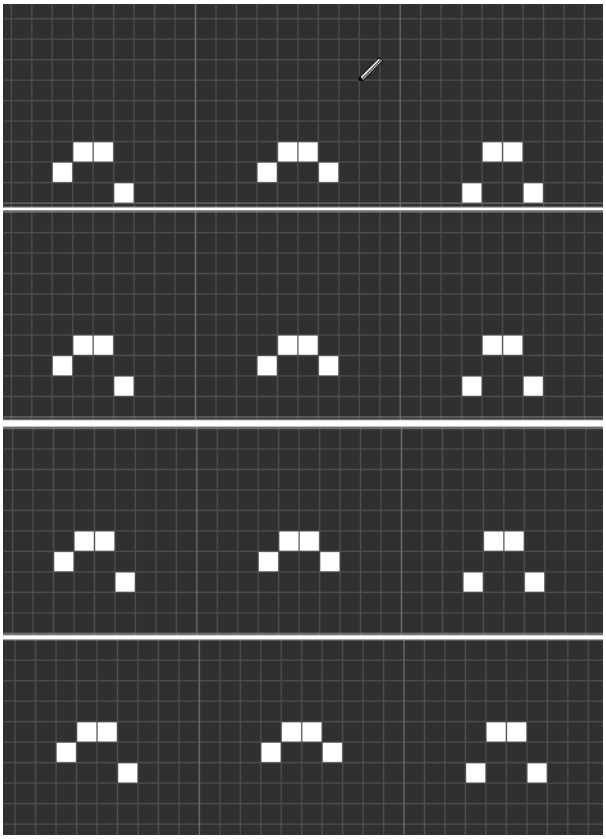

Figura 22- 3 Gliders em 4 gerações consecutivas, que se deslocam para o Norte.

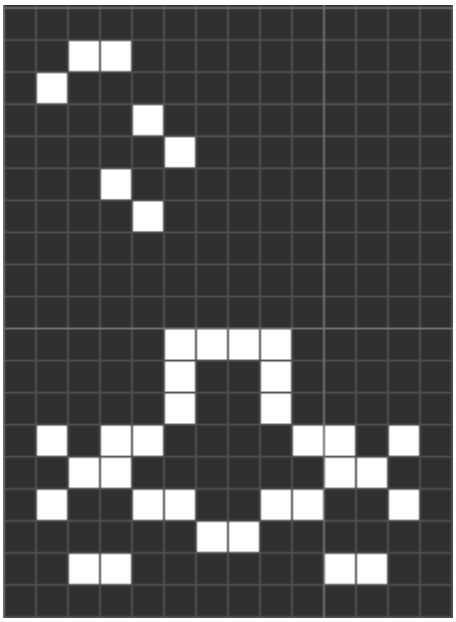

Figura 23 - 2 Gliders em Seeds.

3.2.3 Day and Night

Em abril de 1997, Nathan Thompson descobriu uma variante do jogo de Conway que tem uma interessante propriedade, além de spaceships igualmente interessantes [4].

As regras do Day and Night são especificadas pela notação "B3678/S34678"; desta forma, são requeridos 3, 6, 7 ou 8 vizinhos vivos para o nascimento de uma célula e 3, 4, 5, 6, 7 ou 8 células vizinhas para que uma célula permaneça viva [4].

A propriedade de maior interesse é que sua regra é simétrica no que diz respeito às células mortas e vivas. Isso é, se você toma arbitrariamente objetos neste jogo e os inverte, fazendo células vivas morrerem e vice-versa, então os objetos invertidos têm a mesma evolução que o objeto original teria. É igualmente interessante o fato de que esta variante exibe comportamento complexo, há muitos osciladores, spaceships e guns que combinados emitem periodicamente spaceships de variados tipos [4].

3.2.4 Versões coloridas Life-Like

De maneira geral, nestas variantes do Jogo da Vida os diferentes estados são representados por cores diferentes e suas regras podem ser generalizadas por:

Como em Life, uma célula viva morre de solidão se estiver cercada por menos que dois duas células vivas na vizinhaça de Moore

A célula morre de superpopulação se estiver cercada por mais que três células vivas em sua vizinhaça de Moore

Uma célula nasce somente se estiver cercada de três células vivas em sua vizinhança de Moore

3.2.4.1 Immigration

O Immigration é uma versão bastante conhecida do Jogo da Vida, para ser jogado por 2 jogadores. As células possuem 3 estado (1 morto e 2 estados vivos de cores diferentes). Na Figura 24, estão representadas por preto e branco, as seguintes regras em [16] que determinam qual deve ser a cor das células vivas:

1. Células vivas retém suas respectivas cores permanentemente, até que morram por superpopulação ou solidão (ou seja, uma célula viva não muda de cor a não ser que morra).
2. Quando uma célula nasce, recebe a cor majoritária de seus vizinhos.

Muitos podem ser o objetivo do jogo. Um possível seria de dizimar a cor adversária, ou então ocupar a maior parte do grid, por exemplo.

O Immigration, a primeiro momento, é uma versão feita para entretenimento. Diversas regras podem ser combinadas e algumas alterações podem ser feitas. É exatamente essa a ideia do capítulo 4 deste trabalho, onde optaremos por algumas regras, vamos definir algumas variáveis do jogo e veremos um script para jogarmos o jogo dentro da plataforma Golly.

Apesar de ter esse aspecto lúdico, é possível estudarmos como se comportam alguns objetos de forma interessante no Immigration. Por exemplo, é possível produzir osciladores que possuem períodos maiores que seus equivalentes no Jogo da Vida.

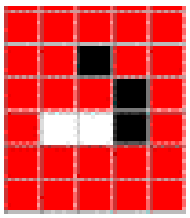

Figura 24 - Grid do Immigration com células brancas e pretas, retirada de [16]

3.2.4.2 QuadLife

Outra variante do Jogo da Vida é o QuadLife, com quatro estados nos quais a célula estaria considerada viva. Estes estados são tradicionalmente representados por quatro cores, podemos assumir azul, amarelo, verde e roxo.

Suas regras são ser descritas por [22]:

Qualquer configuração consistindo de apenas duas cores se comporta exatamente como o Immigration.

Quando uma célula nasce, recebe a cor majoritária de sua vizinhança. Contudo, se seus três vizinhos possuem diferentes cores cada um, a célula que nasce toma a quarta cor restante.

Como em Immigration, uma vez que uma célula recebe determinada cor, permanece com esta cor.

Assim, por exemplo, uma célula morta cercada por um vizinho azul, um vizinho verde e um vizinho roxo, nascerá com a cor amarela.

3.2.4.3 Rainbow Game of Life

O Rainbow Game of Life é similar ao Immigration mas, possui uma regra de coloração diferente [43]:

Cada nova célula recebe uma cor que é a média aritmética, no sistema RGB, dos seus três vizinhos.

Um grande interesse neste jogo surge da possibilidade de um exame de padrões de propriedades "genéticas". Por exemplo, para estudar o impacto dessas propriedades seria interessante observar uma célula inicial da cor preta e então observar a propagação de cinza nas próximas gerações.

De maneira mais concreta, tomemos a configuração da Figura 25:

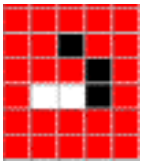

Figura 25 - Configuração arbitrária no Rainbow Game of Life, retirada de [16]

Verifica-se que as células pretas irão espalhar seus genes igualmente entre as outras células de tal forma que todo o padrão depois de um determinado número de gerações não vai atingir tons de cinza homogêneos como mostra a Figura 26 [43]:

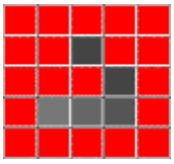

Figura 26- Domínio de células cinza por causa das pretas, retirada de [16]

Um segundo exemplo apresentado na Figura 27 mostra um padrão inicial e seu estado depois de oito gerações [43]:

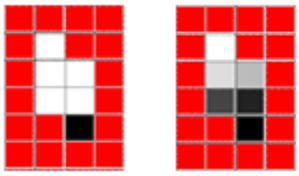

Figura 27- Padrão arbitrário no Rainbow e 8 a configuração 8 gerações depois, retirado de [16]

Podemos perceber após um número grande de gerações as células tornar-se-ão homogêneas.

3.2.4.4 Wireworld

Este é um autômato celular com um propósito bem diferente dos apresentados até então, por isso não se encaixa nos padrões descritos anteriormente. Todavia o incluímos para ilustrar a grande diversidade de aplicações em problemas aplicados nos quais simuladores originados em Life podem ser úteis. O Wirewolrd ou "mundo dos fios" foi inventado por Brian Silverman em 1984 e simula dispositivos eletrônicos e portas lógicas (veja Figuras 28 e 29) por meio de suas células que representam elétrons viajando através de condutores.

A célula pode estar em um de quatro estados possíveis [27]:

- Em branco (preto), uma célula em branco sempre permanece em branco.
- Azul, representa a cabeça do elétron e no momento seguinte sempre torna cauda.
- Cobre, representa a cauda do elétron e no momento seguinte sempre se torna fio.
- Amarelo, representa o fio e se tornará cabeça ou cauda se houver um vizinho seu que seja azul.

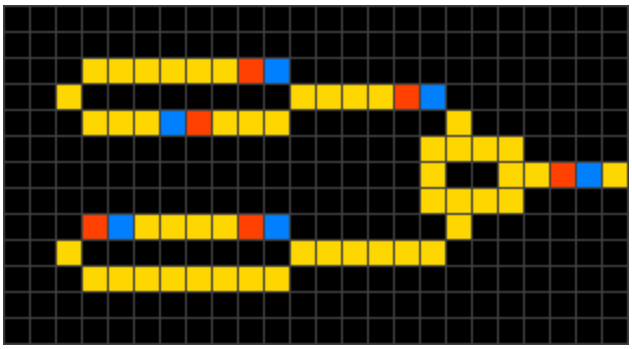

Figura 28 - 2 Geradores de clock e 1 XOR, retirados de [16]

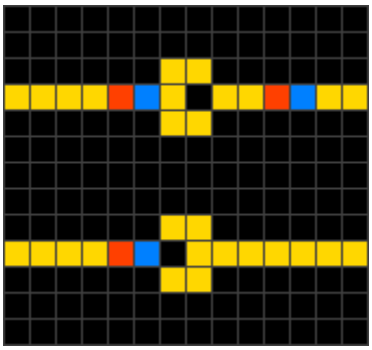

Figura 29 - 2 Diodos, retirados de [16]

3.2.5 - Exemplo de aplicação prática: CRIPTOGRAFIA

Como já foi mencionado, uma característica que torna um Autômato Celular interessante em diversos aspectos e, especialmente à criptografia, é o comportamento caótico. A dificuldade em fazer o processo inverso e a necessidade de se percorrer todos os passos da criptografia fazem dele um modo promissor de encriptação de dados. Veremos superficialmente um algoritmo de criptografia baseado em um autômato celular Life-Like proposto em [22] e [19].

Foram testadas várias regras Life-Like e a que apresentou melhor desempenho para a criptografia foi o AC cuja célula nasce se possuir exatamente 1, 3, 5 ou 7 células vizinhas vivas e ela sobrevive se possuir exatamente 2, 4, 6 ou 8 vivas, como podemos ver em [22] e [19]. O AC então utiliza a regra B1357/2468.

A ideia geral do algoritmo é a seguinte: uma chave K é convertida em semente para o estado inicial do AC baseada em mapa logístico, que é uma função matemática que associa a um dado número x um outro número x´ [23]. Esse número x´ servirá como semente para que possamos configurar o estado inicial do AC. Feito isso, o AC evolui sucessivamente seguindo suas regras. A cada geração, a nova configuração dos estados das células correspondem a um novo número pseudo-aleatório. Assim, são gerados números pseudo-aleatórios (similares aos que vimos na seção 3.2.1). Nestes números pseudo-

aleatórios, o método aplica a operação XOR junto ao texto claro, fazendo a encriptação. Analogamente, a decriptação seria o processo inverso. A figura 30 mostra este processo.

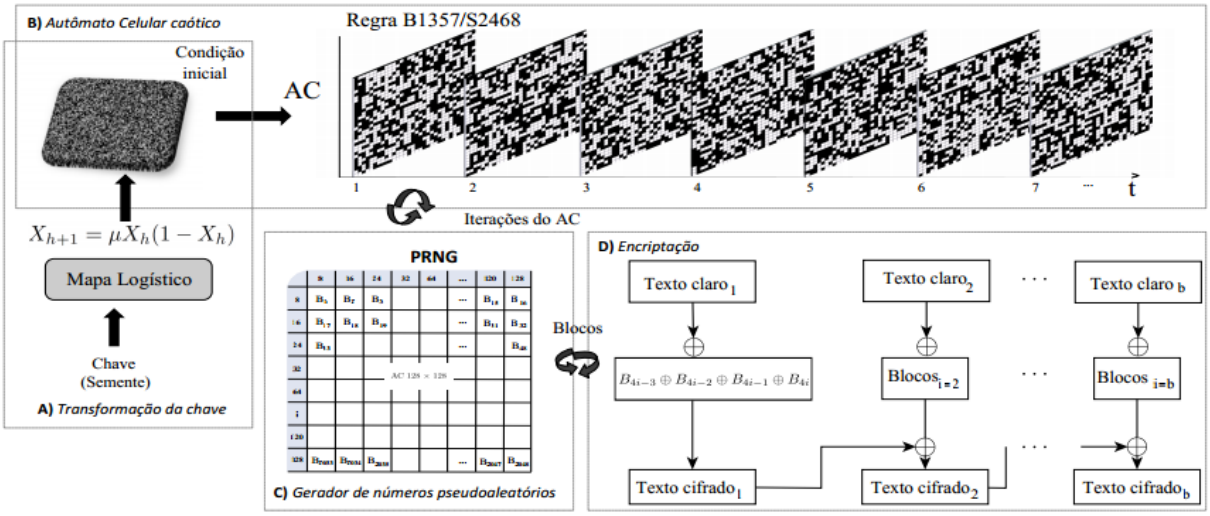

Figura 30 - Resumo do processo de encriptação (Figura 5.1.1 retirada de [19])

Os trabalhos de [22] e [19] mostraram que é possível explorar muito bem o caos dos ACs, especialmente os Life-Like, uma vez que são ótimos geradores de números pseudo-aleatórios.

A simplicidade do algoritmo, bem como seu baixo-custo e os resultados obtidos comparados a alguns dos principais métodos de criptografia também favorecem o uso de ACs caóticos na criptografia.

Imaginando que um "invasor" consiga capturar algum dos estados do autômato celular, ele precisaria dos demais estados para poder descriptografar a mensagem inteira corretamente, sendo esse um dos principais pontos a favor dos ACs na criptografia.

O trabalho [19] ainda propõe um segundo método de criptografia baseado em AC caótico e redes complexas, um algoritmo de hash que foge ao escopo deste trabalho.

3.3 Algumas simulações

Apesar do conceito de autônomo celular ser muito mais amplo, limitaremo-nos a alguns exemplos mais simples e parecidos com os life-like nesta parte do trabalho. Vamos ver, basicamente, aplicações mais complexas que nos itens 3.1 e 3.2 por serem modelos de sistemas mais complexos.

Um conceito importante: por enquanto vimos ACs onde as mesmas regras se aplicam a todas as células. Como a natureza não é tão simples, é natural pensarmos em modelos onde diferentes regras se apliquem a diferentes células num mesmo Autômato Celular. Quando isso acontece, dizemos que se trata de um Autômato Celular não homogêneo.[28]

3.3.1 Tráfego

O sistema de transporte terrestre urbano possui grande importância social e econômica para as cidades, sejam de pequeno, médio ou grande porte. No Brasil, por exemplo, a frota nacional ultrapassou a marca de 80 milhões de veículos (82.819.581 segundo o Detran [11]). O estado de São Paulo lidera essa lista, registrando mais de ¼ do total, dos quais 8 milhões estão somente na cidade de São Paulo. Por isso, não é raro vermos a capital paulista batendo recordes de congestionamento, como o de 309 quilômetros registrado no dia 26 de julho de 2013 [44].

Tais eventos impactam diretamente não apenas a qualidade de vida da população, mas também sua capacidade econômica, já que no Brasil o transporte de mercadorias se dá predominantemente por via rodoviária.

Portanto, diversas pesquisas na tentativa de melhorar o trânsito de veículos têm sido feitas [39]; dentre elas aquelas que buscam analisar os efeitos de mudanças nas regras de circulação do trânsito urbano e rodoviário através de técnicas computacionais, sendo uma delas, técnicas de simulação.

Entre as diversas formas de simular o tráfego, aquelas que utilizam Autômatos Celulares receberam uma atenção especial dos pesquisadores nos últimos anos, principalmente a partir dos anos 90, quando os computadores começaram a ganhar mais capacidade de processamento."

Em [39] é implementado um modelo de fluxo de tráfego num autômato celular não homogêneo (regras diferentes para células diferentes) em duas dimensões baseado no modelo CAUTS (Cellular Automata for Urban Traffic Simulation, Autômato Celular para Simulação de Tráfego Urbano). É um modelo bem poderoso, que simula trânsito para avenidas e ruas secundárias, considera semáforos e paradas de ônibus e até eventos como acidentes ou veículos com problemas, como podemos ver na Figura 31. O modelo de [39] usa as células para representar espaços de 5,5 metros (carro popular no Brasil com veículos ao redor). Uma célula possui somente dois estados (ocupada ou não), mas é representada por uma 4-upla (direção primária -a faixa em que se encontra -, direção secundária - em qual rua deve entrar, por exemplo -, limite de velocidade e o indicador do estado).

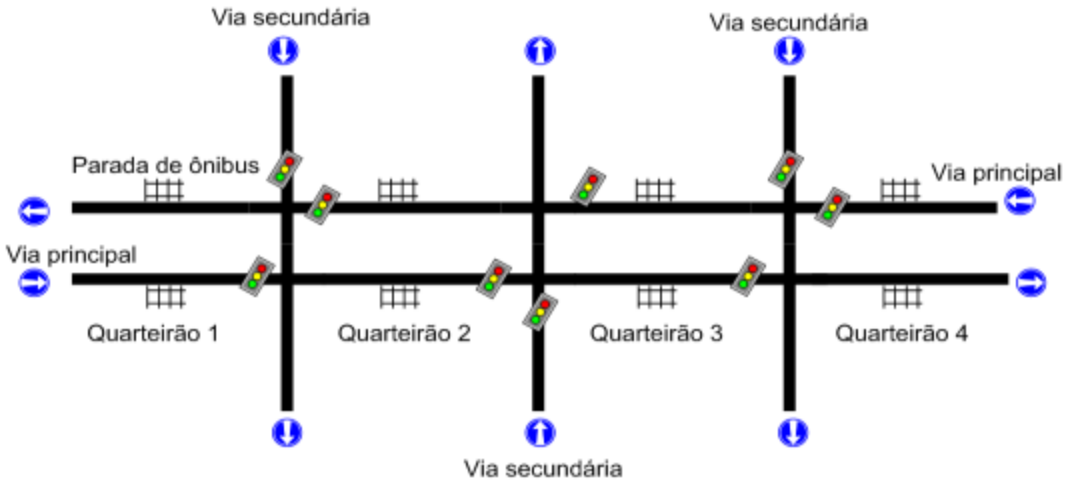

Figura 31 - Layout do modelo e seus componentes principais em destaque (Figura 5.1 de [39])

.

3.3.2 Autômatos Celulares e Epidemias

A importância de se estudar a epidemiologia está associada ao fato de se tentar criar métodos preditivos que possam descrever com alguma precisão o comportamento de epidemias para que, com essas informações, possam ser adotadas políticas de prevenção evitando eventos de proporções pandêmicas e a morte de milhões de pessoas.

Devido à relevância deste assunto, vários pesquisadores buscam desenvolver modelos matemáticos que possam contribuir para compreensão e a erradicação de doenças infecciosas.

Um dos modelos mais estudados é o modelo matemático denominado SIR (Suscetível - Infectado - Recuperado) de Kermack e McKendrick [46]. O modelo SIR permite analisar determinadas características de doenças infecciosas, tais como as constantes de tempo características da fase epidêmica, o patamar endêmico, e a existência de limiares nas taxas de propagação para possibilitar a erradicação de doenças infecciosas. Existem muitos outros modelos, principalmente porque ainda é uma área a ser mais desenvolvida. A partir do modelo SIR (ou outros modelos), pode-se desenvolver regras para determinados autômatos mais complexos com o objetivo de analisar, espacialmente, como um fenômeno (doença, por exemplo) se comporta [9].

É também interessante notarmos que da mesma forma que os ACs podem ser utilizados para o estudo generalizado de epidemias, também é possível por meio de mudanças de parâmetros adaptar o cenário afim de estudar casos específicos como podemos ver a seguir.

“A transmissão da hepatite B foi estudada por Ahmed, Agiza e Hassan (1998), dividindo-se a população de infectados em dois grupos: os assintomáticos, que têm baixa probabilidade de transmitir a doença durante toda a sua vida; e os doentes, que possuem maior probabilidade de transmissão, mas que se recuperam após um período de aproximadamente 20 dias. As regras de infecção são probabilísticas e as regras de cura são determinísticas.” [35].

São mencionados vários estudos em [37], que foram feitos testando hipóteses. Por exemplo, foram feitos para analisar o efeito de uma epidemia considerando o fluxo da população. Outros estudos consideram cada célula não um indivíduo, mas uma concentração mais ou menos densa de pessoas, e também sobre o efeito da vacinação, aplicada em diferentes regiões do espaço.

3.3.3 Autômatos Celulares aplicados no Combate ao Câncer

O crescimento cancerígeno é um dos mais intrigantes assuntos na ciência contemporânea. Por esse motivo, e por envolver uma gama de questionamentos não respondidos acerca deste assunto, diversos cientistas buscaram a formulação de vários modelos de crescimento com o propósito de examinar uma ou várias características principais do câncer, tais como a metástase, a disponibilidade de nutrientes, a competição por recursos, as atividades realizadas pelo sistema imunológico etc.

Em 2001, foi desenvolvido um modelo que consistia de autômatos celulares juntamente com equações diferenciais para descrever um modelo que representasse o crescimento cancerígeno . Os tipos de células representadas são: células normais, cancerígenas, cancerígenas mortas ou espaços vazios. A difusão de glicose, de íons H+ e a densidade vascular são grandezas utilizadas e que são calculadas através de equações diferenciais. Como resultado, esse modelo permitiu observar que quanto maior a concentração de H+ maior seria o crescimento desse tumor. Também se observou que a densidade vascular, quando se encontra menor que um determinado valor, induz células normais e cancerígenas a morrerem, já quando maior leva as células cancerígenas a perderem suas vantagens para com as células normais [24].

3.3.4 Células Biologicas

Para sermos justos com a própria nominação CELULAR, não podemos deixar de mencionar Autônamos Celulares que simulam o comportamento de células biológicas. Um estudo brasileiro (projeto de iniciação científica) procura simular uma competição entre uma célula biológica e um outro organismo

[26]. É instigante como podemos estudar modelos da natureza para construir modelos computaconais (ACs) com o objetivo de, em seguida, usar estes modelos computacionais para estudar a natureza. E é mais interessante ainda constatar que apesar de complexa, a natureza se baseia muitas vezes em regras simples. No caso do protótipo de [26], o modelo é um Autômato Celular homogêneo!

Capítulo 4 - WarLife, um estudo de caso

Como já vimos na seção **3.2.4.1**o jogo Immigration é uma versão do Jogo da Vida para 2 pessoas. Neste capítulo, vamos definir algumas regras e parâmetros para simular, dentro da plataforma Golly, uma versão própria do jogo Immigration, que chamaremos de Warlife.

WarLife é uma versão do jogo Immigration adaptada pelos autores deste trabalho. Sendo assim, o código usado na implementação do mesmo deriva da implementação encontrada em [5] do Immigration Game.

Warlife é um instrumento recreativo que, similarmente ao xadrez, estimula o raciocínio, forçando os jogadores a pensarem com estratégia e de forma caótica (relativa à Teoria do Caos). Lúdico, poderia ser usado para o ensino dos conceitos básicos do Jogo da Vida para crianças e estudantes além de servir como espécie de "ginástica mental" para pesquisadores iniciantes na área dos autômatos celulares, principalmente do Jogo da Vida de Conway.

Posteriormente, poderia ser usado como instrumento de simulação de batalhas em jogos ou mesmo militares. Na verdade, poderia, no futuro, se aprimorado, simular qualquer tipo de confronto, seja ele bélico ou de informações, de interesse ou qualquer outra disputa A x B.

Apesar do Warlife ser de 2 jogadores, a implementação para mais jogadores por vez é fácil, bastando definir os parâmetros como tamanho do tabuleiro, quantas gerações representam uma jogada e algumas regras de transição.

4.1 Regras do Autômato

As características do autômato e as regras do WarLife são baseadas no Jogo da Vida de Conway. Algumas diferenças, no entanto, são cruciais para o WarLife.

O autômato possui 3 estados: 0, 1 ou 2 (que aqui representarão, respectivamente, o estado morto, vermelho e amarelo. Essas cores são arbitrárias e podem ser modificadas na plataforma Golly, como veremos na seção 4.3. No Jogo da Vida eram 2 estados: morto ou vivo (0 ou 1).

O grid é finito (o Jogo da Vida possui grid infinito). Convém notar que essa foi uma decisão arbitrária dos autores do trabalho, que consideraram melhorar a jogabilidade do projeto. Da mesma forma foram escolhidos o tamanho do grid 15x15 e a forma toroidal (o vizinho LESTE da célula (i,15) é (i,1),

num grid de 15 colunas). Vale ressaltar também que, depois de alguns testes, foi verificado que grids menores que 9x9 limitam bastante o jogo, e maiores que 20x20 tornam o jogo muito demorado.

As regraes de transição são as seguintes:

1- Uma célula morta com exatamente três vizinhos vivos nasce da cor da maioria dos seus vizinhos vivos. Ou seja, considere uma célula morta com 3 vizinhos. Se 2 ou 3 deles são vermelhos, ela nascerá vermelha na geração seguinte. Caso contrário, nascerá amarela.

2- Uma célula viva com dois ou três vizinhos vivos permanece viva (sobrevivência) sem mudar sua cor.

3- Em todos os outros casos, as células morrem ou permanecem mortas (superpopulação ou solidão).

4.2 Jogabilidade

Vamos referenciar alguns parâmetros de forma genérica antes de defini-los, para efeito didático.

1. Antes do jogo começar, cada jogador pode colocar N células da sua cor no tabuleiro. Cada jogador pode ver a configuração adversária e definir a sua própria estratégia. Estipula-se um tempo máximo para esta fase inicial.

2. Uma vez iniciado o tabuleiro com N células vermelhas e N+1 células amarelas, é avançada uma geração. N precisa ser obrigatoriamente menor que a metade do tamanho do grid.

3. O jogador vermelho vai então inserir M peças da sua cor no grid, onde ele bem entender (desde que a célula esteja morta) e avança-se uma geração. Logo em seguida, o jogador amarelo é que insere M peças de sua cor no grid, onde ele quiser. Após isso, avança-se então 1 geração. M é um número que será decidido posteriormente, mas convém não ser muito grande. A decisão depende do tamanho total do grid.

4. O passo 3 se repete, com a ordem invertida. Primeiro o jogador amarelo e depois o vermelho. Uma geração é avançada.

5. Os passos 3 e 4 se repetem alternadamente até que um dos jogadores esteja sem mais células ou até a geração G, onde o jogo para e conta-se qual jogador possui mais células da sua cor. Se houver um empate, estipula-se um novo G e o jogo continua. G também não possui nenhuma limitação matemática, mas quanto maior, mais tempo pode demorar para o jogo terminar.

6. A cada B rodadas, é permitido a cada jogador utilizar a superbomba, que destrói todas as células vivas vizinhas (independente da cor) ao seu redor. B é um número arbitrário menor que G. Para o

jogo ficar mais interessa, a superbomba não pode ser o golpe de misericórdia (o jogador não pode utilizar a superbomba se ela eliminar todas as células adversárias). Precisa deixar ao menos uma adversária viva.

Algumas características do jogo são customizáveis. A ideia é que cada jogador monte estratégias diferentes para cada customização de jogo e até se especialize em algumas. A seguir, uma customização que, nos testes de nosso trabalho, pareceu bastante razoável, possibilitando uma boa jogabilidade dinâmica sem tornar o jogo muito lento ou demorado, nem cansativo.

Tamanho do Grid: 15 x 15

Formato do Grid – Toroidal (poderia ser plano)

Número N de Células iniciais de cada jogador: 10

Número M de células novas por rodada: 3

A cada B gerações, faz-se uma nova bomba: 10

A cada G gerações, verifica-se quem tem maioria para declarar um vencedor: 30

O jogo pode ter outras características modificadas, como usar a vizinhança de Neumann ao invés da de Moore, ou outra regra que não a B3/S23. Contudo, nos testes de jogabilidade que fizemos, procuramos ser fiéis ao Jogo da Vida de Conway, porque uma das motivações possíveis do jogo é usa-lo como forma didática para o ensino do Jogo da Vida. O jogo pode ser uma boa forma de iniciantes na área brincarem e entenderem como Vida funciona, acostumando o cérebro e treinando ele a prever gerações.

4.3 Implementação

A plataforma escolhida foi o Golly, um projeto open-source com reputação muito boa no meio. Especializado em autômatos celulares, permite programação com python/perl além de trabalhar com os formatos de arquivo de autômatos celulares mais comuns (5). Poderíamos ter escolhido qualquer outra linguagem, como Java, C e suas variações, Javascript etc, mas a decisão de escolher uma plataforma especializada no assunto foi crucial para o andamento do trabalho, uma vez que o foco dele não é a

codificação e a programação de outras linguagens. Essa decisão economizou tempo, esforço e, principalmente, ampliou o conhecimento sobre autômatos celulares.

Vale a observação que o Golly[3], nativamente, apresenta uma coleção bem grande do estado da arte dos autômatos celulares, como mostra a área esquerda da Figura 32. Esses padrões (ou *patterns*) se referem a uma configuração particular de estados (alguns objetos, alguns cenários etc) seguindo determinadas regras, que podem ser a conhecida B3/S23, ou qualquer uma outra. Estamos interessados em escrever uma regra nova, que vai reger o jogo Warlife.

Criamos o arquivo (o *script* completo está anexado ao final deste item) "Warlife.table". A extensão ".table" é um formato específico para autômatos celulares detalhado em [33]. Foi escolhido por conta da sintaxe fácil e intuitiva para iniciantes. Não foi preciso nenhuma programação especial em *python*, nenhum desenvolvimento/uso de DLLs[4].

Códigos escritos no Formato ".TABLE" são feitos em linhas de texto ASCII. Cada linha pode ser um comentário, uma propriedade obrigatória do autômato celular a ser descrito (número de estados, tipo de vizinhança, tipo de simetria), uma variável ou uma transição. Veremos cada um desses tipos analisando o código de "Warlife.table".

À esquerda da tela da Figura 32, uma coleção de padrões/regras conhecidas, divididas em grandes diretórios. O diretório LIFE se refere aos padrões que adotam a regra do Jogo da Vida de Conway (B3/S23). Dentro dele, outras sub divisões, como as diversas GUNS e, dentre elas, o pseudo-p34-gun.

[3] Como já dito, encontra-se em http://golly.sourceforge.net/ (último acesso em 27 de maio de 2014).

[4] Para saber mais sobre desenvolvimento de DLLs com Golly, recomenda-se a página de ajuda de Python do Golly disponível em:< http://golly.sourceforge.net/Help/python.html> . Último acesso em 27 de maio de 2014).

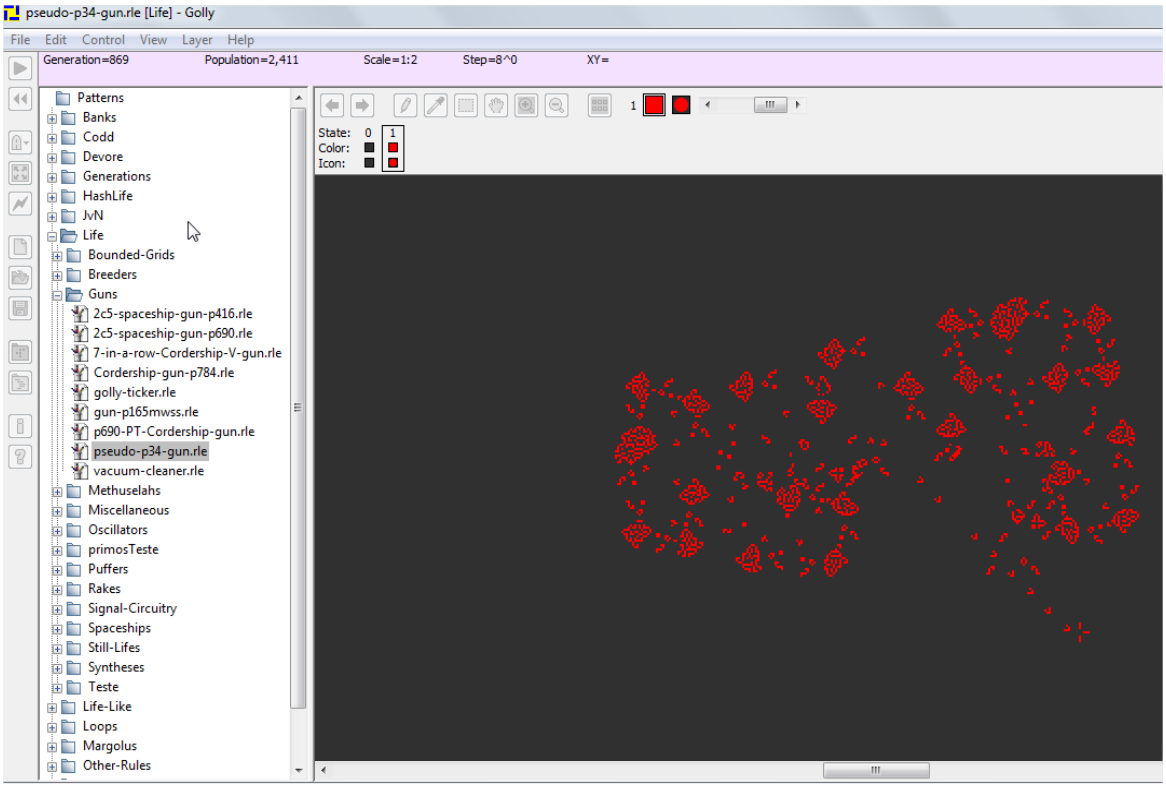

Figura 32 - Tela do Golly

Para um melhor entendimento, analisaremos o código do WarLife, dividindo-o por partes.

Parte 1: Os comentários iniciais

Todo código razoável, do HTML ao Java, possui comentários com informações sobre o programa desenvolvido. Com Warlife não seria diferente. Comentamos (linhas em branco ou as iniciadas com "#") e colocamos informações como nome da regra, autores, referências e instruções relevantes.

```
#  WarLife
#  Desenvolvido por LeandroZoucas@gmail.com
# e Pedro.Oliveira@uniriotec.br, orientados pela professora Flavia.Santoro@uniriotec.br
#  Totalmente inspirado no Immigration Game, que por sua vez baseia-se no Game of Life de
#  Conway
```

O código do Immigration foi acessado em

http://boardgamegeek.com/filepage/82102/immigration-ruleset-for-golly no dia 2014.05.18.

Para usar, coloque este arquivo no diretório RULES dentro do diretório principal do Golly.

Depois, vá no menu CONTROL/SET RULE e entre com:

Warlife:T15,15 para toroidal ou

Warlife:P15,15 para plano.

Para usar, coloque este arquivo no diretório RULES dentro do diretório principal do Golly.

Depois, vá no menu CONTROL/SET RULE e entre com:

Warlife:T15,15 para toroidal ou

Warlife:P15,15 para plano.

Parte 2: Propriedades obrigatórias

São 3 as propriedades obrigatórias de um autômato celular que devem ser descritas no formato ".table":

O número de estados, escrito por um inteiro. Temos sempre n estados, variado de 0 a n-1. No caso do Warlife, são 4 estados: morto, time1, time2, superbomba.

O tipo de vizinhança (Moore, Von Neumann, hexagonal e ainda algumas outras vizinhanças são atualmente suportadas pelo Golly[5]).

O tipo de simetria. O conceito de simetria será melhor entendido ao analisarmos as variáveis e as regras conjuntamente. Por enquanto, é suficiente dizer que usamos a permutativa, e que para cada tipo de vizinhança, existem diferentes tipos de simetrias.

n_states:4

neighborhood:Moore

[5] Ver o Roadmap do Golly. Disponível em <https://code.google.com/p/ruletablerepository/wiki/RoadMap>. Acesso em 27 de maio de 2014.

symmetries:permute

Parte 3: Variáveis

Do mesmo jeito que as simetrias, as variáveis aqui usadas fazem mais sentido quando analisarmos as regras. Agora, é importante notar que as variáveis podem ser entendidas como vetores matemáticos com diferentes valores. Esse vetor representa todos os valores possíveis da variável. Por exemplo, a variável “a” representará tanto o valor “1” como o valor “2”.

Em versões mais antigas do Golly, uma mesma variável podia ser chamada várias vezes numa mesma regra. Como isso não é mais possível, precisamos usar variáveis redundantes, ou seja, variáveis repetidas com os mesmos valores. Existem algumas alternativas, mas não vamos nos aprofundar nisso. [34].

var a={1,2}

var b={1,2}

var c={1,2}

var d={1,2}

var e={1,2}

var w={0,1,2}

var x={0,1,2}

var y={0,1,2}

var z={0,1,2,3}

var j={0,1,2,3}

var k={0,1,2,3}

var l={0,1,2,3}

var m={0,1,2,3}

var n={0,1,2,3}

var o={0,1,2,3}

var p={0,1,2,3}

Entenderemos melhor o porquê de uma variável possuir mais de um valor ao analisarmos as regras.

Parte 4: Regras

Vamos então à parte mais interessante: as regras de transição do autômato! Elas dependem diretamente do tipo de vizinhança e de simetria definidas anteriormente. Além disso, as variáveis, que não são obrigatórias, facilitam (e muito!) a escrita e leitura das regras, como veremos a seguir.

Devido à intima relação das regras com a vizinhança, convém lembrar qual foi escolhida e o padrão referencial próprio do tipo de vizinhança. Por isso, podemos considerar uma boa prática inserir o seguinte comentário:

C,N,NE,E,SE,S,SW,W,NW,C' para a vizinhança de Moore

Neste comentário, estamos lembrando que as regras seguirão a seguinte ordem: o primeiro valor se refere à própria célula central na geração atual. O segundo valor se refere à célula na posição Norte, o terceiro valor, à posição Nordeste e assim sucessivamente. O último valor vai se referir novamente à célula central, mas agora a seu valor final na geração seguinte. Ou seja, se alguma célula estiver no estado de mesmo valor da posição C, e tiver a exata vizinhança descrita nas posições seguintes, a célula deve mudar seu estado para o valor de C`. Na Figura 33, mostramos a célula central em vermelho e o numero referente à ordem que devemos ler seus vizinhos (primeiro 1, depois 2, 3…). A última a ser lida, a posição 9, se refere ao estado final da célula central.

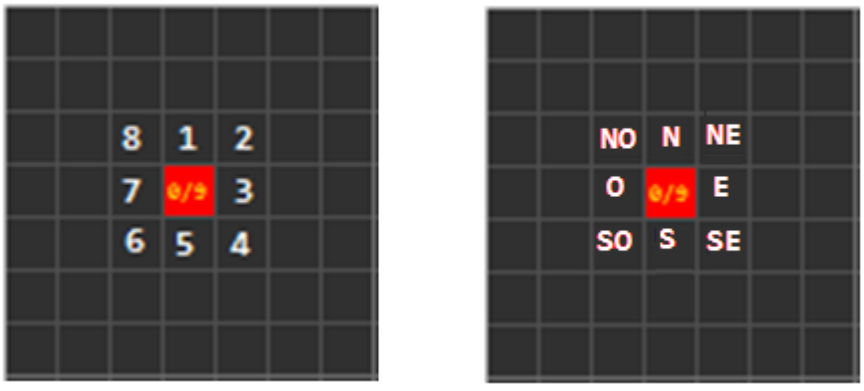

Figura 33 – A ordem das posições dos vizinhos de uma célula.

Na Figura 33, repare que a célula central é a primeira (representando o seu estado inicial) e a última (representando o seu estado na geração seguinte). A leitura é feita em sentido horário, a começar pela célula ao Norte.

As regras de transição no formato ".table" são muito simples de serem entendidas, bastando saber qual o padrão de referenciamento da vizinhança escolhida. Cada regra tem o tamanho de N+2 elementos, onde N é o número de vizinhos de cada célula. Como a Vizinhança de Moore define 8 vizinhos para cada célula, cada regra terá 10 parâmetros. A ideia geral para a leitura é a seguinte:

Considere uma célula qualquer no estado X. Procura-se então as regras de transição que tenham como estado inicial (primeiro parâmetro) o valor X. Se essa célula tiver uma vizinhança que combina exatamente com essa regra (por exemplo, a regra diz que a vizinha ao Norte e Nordeste são as únicas vivas e no estado 1. Então, uma regra definida por X,1,1,0, 0,0,0,0,0,Y se aplica a essa célula, que na geração seguinte passará do estado X para Y.

No caso do Warlife, para uma célula nascer, ela precisa ter exatamente 3 vizinhos vivos. Digamos que ela tenha exatamente 3 vizinhos no estado 1. Podemos escrever a seguinte regra para células neste caso:

```
#CELULA MORTA TEM EXATAMENTE 3 VIZINHOS VIVOS 1 EM N,NE,E
0,1,1,1,0,0,0,0,0,1
```

Estamos dizendo que uma célula morta (posição 0 do vetor, com valor 0) que tem vizinhos no estado 1 na posição N, NE, E e mortos nas posições SE,S,SO,O,NO deverá se tornar viva (estado 1) na geração seguinte (última posição do vetor).

É fácil perceber que as combinações são muitas. Para cada vizinhança que estabelece v vizinhos num autômato de e estados possíveis, temos e^v combinações diferentes para cada regra, se a posição de cada vizinho não importa. No caso específico do Warlife, não importa se os 3 vizinhos vivos são N,NE e E ou S,SO e O. Daí a importância de definir a simetria como PERMUTATIVA. Ao defini-la deste modo, dizemos que a ordem não faz diferença e economiza muitas linhas de código.

Outra questão é a seguinte: muitas vezes não importa o valor da variável para a regra. Por exemplo, se queremos falar de uma célula qualquer no estado inicial i que vai para o estado final f se ela tiver 3 vizinhos e a maioria dos seus vizinhos no estado 2, podemos definir a regra assim:

#Se uma célula i tem maioria de vizinhos no estado 2, podemos defini-la deste modo:

i,2,2,a,0,0,0,0,0,f

Podemos observar neste exemplo acima que embora os estados com valor zero pudessem receber outros valores, a configuração acima "2,2,a" deveria ser mantida, ou seja, poderíamos apenas realizar operações de *shift* (permutar) nesses valores, já que a separá-los inadequadamente poderia pelas regras levar as células à morte pelas regras do jogo, a exceção da permutação desses estados já que "2,2,a", "2,a,2", ou "a,2,2" não faria diferença.

Estamos dizendo nesta regra que a célula tem 3 vizinhos nos valores 2, 2 e q. A variável a está representando um valor qualquer 1 ou 2 (a foi definido anteriormente na parte das variáveis). Logo, a regra descrita na verdade representa as duas seguintes regras:

i,2,2,**1**,0,0,0,0,0,f

i,2,2,**2**,0,0,0,0,0,f

E como definimos a simetria permutativa, podemos permutar esses valores trocando a ordem dos vizinhos, representando todo o conjunto possível de vizinhança com maioria no estado 2.

Com as variáveis a,b,c,d,e,x,y e z definidas, temos as seguintes regras:

NASCIMENTO

0,a,1,1,0,0,0,0,0,1

0,a,2,2,0,0,0,0,0,2

SUPERPOPULACAO

a,b,c,d,e,w,x,y,z,0

SOLIDAO

a,z,0,0,0,0,0,0,0,0

SOBREVIVE COM 2 VIZINHOS MANTENDO SEU ESTADO

1,a,b,0,0,0,0,0,0,1

2,a,b,0,0,0,0,0,0,2

SOBREVIVE COM 3 VIZINHOS MANTENDO SEU ESTADO

1,a,b,c,0,0,0,0,0,1

2,a,b,c,0,0,0,0,0,2

SUPERBOMBA

e,3,z,j,k,l,m,n,o,0

3,z,j,k,l,m,n,o,p,0

A “superbomba” foi um conceito desenvolvido neste trabalho que não estava no Immigration Game original. A partir da rodada 10 (valor arbitrário), cada jogador ganha uma superbomba. Cada superbomba dura no máximo 10 rodadas (ou seja, o jogador pode utilizá-la da rodada 10 à rodada 19), que é também o tempo de fabricação de uma nova bomba. Em outras palavras, cada jogador pode usar somente uma superbomba a cada 10 rodadas, começando pela 10a. O seu grande poder é destruir todas as células vivas vizinhas e se autodestruir. Ela nada mais é que um terceiro estado (estado 2) e todas as células que a tem como vizinha passa ao estado 0.

Embora existisse a possibilidade de definirmos as cores dos times (a nível de código), usamos a configuração padrão do Golly. Para trocar, vá no menu FILE / PREFERENCES/ COLOR e escolha o gradiente que mais lhe agradar, como mostra a Figura 34.

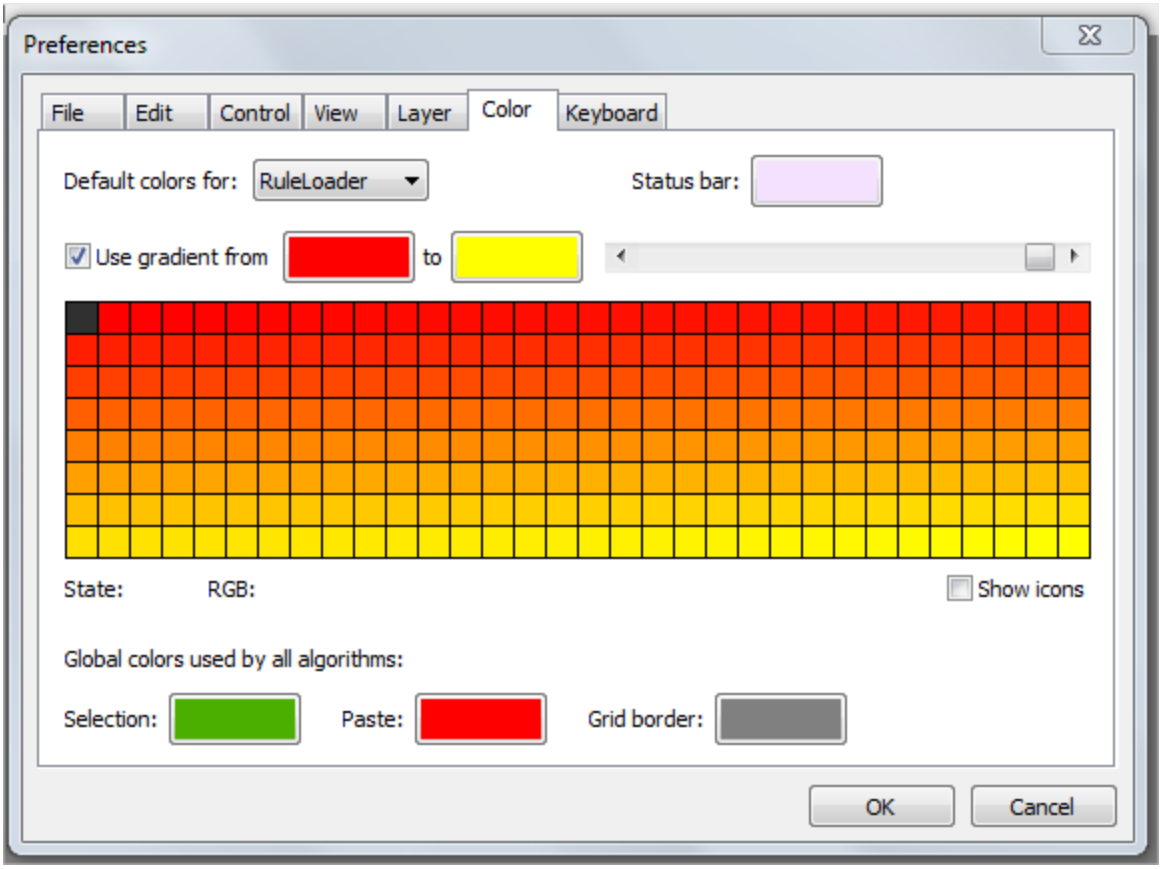

Figura 34 – Guia COLOR no menu FILE;PREFERENCES. Escolha o gradiente de cor que mais lhe agradar.

O arquivo Warlife.table está anexado no final deste trabalho.

Para poder jogar o jogo na plataforma Golly, é necessário salvar este arquivo na pasta RULES do diretório do Golly (preferencialmente) e, ativar a regra indo no menu CONTROL, SET A RULE e digitar "Warlife:T15,15". Usamo '15,15' para nos referirmos ao número de colunas e linhas do grid respectivamente. A letra T refere-se ao grid toroidal.

Chame algum outro estudante de Sistemas de Informação (ou afins) e boa diversão!

Capítulo 5 – Conclusões

O Jogo da Vida começou como apenas um jogo recreativo e mostrou ser aplicável a um número inimaginável de situações e áreas diferentes. Uma estrutura tão simples e tão complexa pode ser vista sob diversos olhares.

Filosoficamente, observamos que Conway experimentou suas regras até que fosse possível atingir um equilíbrio: nem células morrendo exageradamente, nem explosões demográficas. A partir daí, é notável fazer um paralelo com as configurações simétricas que surgem em Life por conta desse equilíbrio ou harmonia que é uma forte característica presente na natureza. Filósofos antigos já acreditavam que a beleza era determinada pela harmonia e simetria. A proporção áurea, muito utilizada em arquitetura e outras formas industriais, é aproveitada, por causar atração em relação à sua estética. Rostos humanos podem ter sua beleza definida também por sua simetria e harmonia, assim como animais escolhem seus parceiros com base na presença de simetria. Os favos de colmeias e seus padrões hexagonais, corujas, borboletas, flocos de neve, o DNA, e outros diversos exemplos podem ser encontrados na natureza. Biólogos acreditam que a simetria é um indicador de bom estado ou de bons genes, pois somente organismos fortes podem manter um desenvolvimento simétrico frente às pressões do ambiente, tais como doença ou falta de alimento [44]. Poderíamos então dizer que Life evidencia esta mesma relação da natureza equilíbrio e proporção, padrão e regularidade, harmonia e beleza, ordem e perfeição, com configurações simétricas que aparecem sem que nenhuma regra referente a este tipo de padrão tenha sido determinada.

O Jogo da Vida, apesar de muito estudado, ainda é um assunto muito limitado a algumas esferas acadêmicas. Apesar de ser uma estrutura voltada à computação, ela deveria ser aproveitada dentro do curso de Sistemas de Informação. Ainda que seja um curso onde a informática não é um fim, mas um meio, ou seja, o foco não é a computação, mas as organizações e como as informações por ela trafegam, os Autômatos Celulares (principalmente o Jogo da Vida) são sim relevantes, dada a imensa diversidade de aplicações. Além disso, representam sistemas, com entrada e saída. No futuro, espera-se que mais e mais sistemas da natureza ou artificiais possam ser modelados nessa estrutura: teremos demanda para profissionais e acadêmicos que entendem do assunto. Um profissional mais amplo como o bacharel em Sistemas de Informação seria o esperado a entender como os sistemas funcionam e como eles podem ser definidos computacionalmente.

Vemos que a documentação sobre o Jogo da Vida ainda é bem escassa na literatura e na internet, apesar do esforço de muitos. Durante o trabalho, ficou clara a carência de uma base melhor documentada,

principalmente no que diz respeito às linguagens de programação usadas pelas plataformas de ACs, como o Golly. Aliás, projetos no Golly são muito bem vindos, uma vez que demonstrou ser uma plataforma útil para o estudo do conceito e bastante intuitiva. Em pouco tempo pudemos imaginar e implementar na plataforma o Warlife.

Entendemos que Golly não seja uma IDE, um ambiente de desenvolvimento. Até porque, nem a edição dos scripts é feita diretamente dentro dele. Mas é uma plataforma, um framework onde podemos trabalhar com autômatos celulares. Warlife foi feito em pouquíssimo tempo (algumas poucas horas incluindo o estudo da documentação da linguagem) porque não tivemos que nos preocupar com toda a implementação. Se fôssemos implementar as gerações, a inserção de células, a vizinhança, enfim, todas as propriedades de um autômato celular, teríamos com certeza gasto muito mais tempo. Já que essas estruturas são nativas no Golly, nos preocupamos somente com a customização do jogo.

Em formações de batalha, a Geometria foi amplamente estudada e empregada e também presente estava a simetria, o exército romano e o grego, por exemplo, são estudados até hoje por seus grandes feitos táticos. Estes exércitos passaram a se basear não somente na dependência da vantagem numérica, qualidade de suas tropas ou simplesmente sorte, ao invés disso, levavam em conta fatores como o terreno, o tipo de tropas inimigas e suas habilidades. A formação mais conhecida do antigo exército Grego era a falange, já os romanos contavam com mais de 7 tipos de formações para diversas situações, incluindo uma das mais famosas, o casco de tartaruga [40]. Atualmente forças policiais se baseiam naquelas formações para conter distúrbios civis assim, o WarLife poderia ser evoluído e adaptado para auxiliar estas forças a simular as diversas formações em diferentes situações, otimizando os recursos humanos taticamente para que o distúrbios fossem contidos o mais rapidamente possível evitando o maiores prejuízos.

WarLife, mais que um simulador, pode ser entendido como “apenas” um jogo. Recreativo mas, principalmente, didático, pois estimula a pensarmos de um jeito que não estamos acostumados. O jogo nos força a pensarmos de forma caótica, prevendo jogadas à frente. Um estudante ou pesquisador novo na área pode usá-lo para exercitar esse novo tipo de raciocínio. Um possível projeto futuro era implementar, depois de evoluído no próprio Golly, um Warlife novo e completo, com sistema automatizado para controle das superbombas, onde um jogador não vê a jogada adversária, onde haja maior customização e, claro, onde haja maior preocupação com a experiência do usuário, com a interação humano-computador.

Resumindo, sentimos que passamos por um terreno muito amplo e pouquíssimo trabalhado. É incrível a diversidade das aplicações e imaginamos o quanto da ponta do iceberg nós já vimos. Uma outra abordagem que não estudamos é a de autômatos celulares independentes de grid, como os padrões

“flocking” (simulando peixes ou pássaros) [7], ou ainda as aplicações artísticas do Jogo da Vida, sem falar na criptografia de imagens, nas simulações sociais.

Neste trabalho não foi discutida uma das mais importantes observações sobre o Jogo da Vida: Nele, é possível implementarmos uma máquina de Turing [31]. E, se uma máquina de Turing pode estar dentro do Jogo da Vida, provamos que tudo que é computável pode ser representado no Jogo da Vida! Esta poderosa característica deve ser melhor estudada, pois a literatura internacional, e ainda mais a brasileira, ainda são carentes nesta área, pelo menos a mais aberta para o público mais leigo.

Podemos pensar também que os autômatos celulares são uma estrutura discreta prima dos grafos: um projeto interessante é pesquisar sobre a relação entre as duas estruturas: qualquer AC pode ser transformado em um grafo? E vice-versa? Caso afirmativo para algumas dessas perguntas, será que existe um algoritmo universal? Seria bem instigante também definir peso diferente para cada célula vizinha de uma célula central (talvez usando grafos). O estado de uma célula se modificaria baseado numa média das demais.

Um projeto interessante era modelar uma rede social em um autômato celular e pensar como uma informação trafega (é compartilhada) entre as diversas células, que são vizinhas se interagem entre si na rede. Algo semelhante a um grafo, mas num tempo discreto.

Na verdade, o mais empolgante de tudo nessa área é a possibilidade de fazer as simulações sociais e trabalhar na área de sociedade e vida artificial. Essas simulações podem sim ser aplicadas a filmes de animação, a vídeo games e, principalmente, para estudo de outras ciências. Entender como a população (de animais, fungos, plantas, vírus ou pessoas) se comportam a partir dessas simulações é muito empolgante, não só para a computação, mas para as ciências humanas (e outras exatas). Num mundo cada vez mais interdisciplinar, onde os diversos campos do saber, apesar de muito específicos, precisam conversar, o uso de autômatos celulares, como o Jogo da Vida, parece bastante promissor.

Pudemos verificar também, neste trabalho, que aplicamos conceitos específicos de diversas disciplinas do curso de Sistemas de Informação. Talvez a área que não tenha sido utilizada foi a de negócio (Análise Administrativa e Empresarial, Empreendedorismo, Administração Financeira)... Um possível projeto, no futuro, é implementar o jogo numa versão móbile e, para isso, esses conceitos deverão ser utilizados ao disponibilizarmos o jogo numa app store. Os conceitos aqui utilizados possuem relação maior com disciplinas de algoritmos, estruturas de dados, de lógica e de computação de modo geral. Mas também envolveu desenvolvimento de código (vimos que podemos aprender o básico de uma linguagem inteiramente nova em pouco tempo). Envolveu conceitos de processo de software (afinal, fizemos várias versões do jogo até chegarmos à atual), gestão de projetos (este trabalho foi um projeto!).

Envolveu também conceitos aprendidos na área de interação humano-computador, uma vez que precisamos fazer diversos testes para chegarmos a uma versão do jogo que atendesse aos objetivos. Tivemos que revisar conceitos da matemática discreta também. E, claro, matérias que envolvam a Teoria dos Sistemas, já que o objetivo desses autômatos é modelar sistemas que encontramos na natureza (ou artificiais).

Com este projeto, esperamos que mais pessoas na área de Sistemas de Informação se interessem e pesquisem sobre este assunto, tão amplo, tão aplicável a tantas coisas e, ao mesmo tempo, tão deixado de lado.

REFERÊNCIAS

[1] A.C.J. de Korte; H.J.H. de Brouwers. A Cellular Automata Approach to Chemical Reactions; 1 Reaction Controlled Systems. Chemical Engineering Journal, abr 2013. DIsponível em: <http://josbrouwers.bwk.tue.nl/publications/Journal94x.pdf>. Acesso em: 5 de maio de 2014.

[2] ADAMATZKY, Andrew. Game of Life Cellular Automata. Springer, 2010. 621p.

[3] American Association For The Advancement of Science, 139th Meeting. Disponível em:<http://eaps4.mit.edu/research/Lorenz/Butterfly_1972.pdf>. Acesso em 5 de maio de 2014.

[4] BELL, D. HighLife – An interesting Variant of Life. Arquivo .zip, 2009. Disponível em:< members.tip.net.au/~dbell/articles/HighLife.zip>. Acesso em: 28 de maio de 2014.

[5] Board Game Geek. Immigration Script. Disponível em: <http://boardgamegeek.com/filepage/82102/immigrationruleset-for-golly>. Acesso em 18 de maio de 2014.

[6] BORRIES, F. V.; WALZ, S. P.; BTTGER, M. GAME OF LIFE: On Architecture, Complexity and the concept of Nature as a Game. Birkhauser Basel, 2007.

[7] BROCK, Z.; KARST, N.; SIRIPONG, M. Bio-inspired Computing. 2005. Disponível em: <http://ca.olin.edu/2005/cellular_automata/caandcomputingfinalpaper.pdf>. Acesso em: 25 de maio de 2014

[8] CALLAHAN, Paul.What is the Game of Life? Disponível em <http://www.math.com/students/wonders/life/life.html>. Acesso em: 25 de maio de 2014

[9] CHANG, Sharon. Cellular Automata Modelos for Epidemics. UC Davis Physics. 2008. Disponível em :<http://csc.ucdavis.edu/~chaos/courses/nlp/Projects2008/SharonChang/Report.pdf>. Acesso em 16 de julho de 2014.

[10] DANFORTH, Christopher M. Chaos in an Atmosphere Hanging on a Wall. 2013. Mathematics of Planet Earth 2013.

[11] DENATRAN. Frota 2014. Disponível em: <http://www.denatran.gov.br/frota2014.htm>. Acesso em 25 de maio de 2014.

[12] ERATOSTHENES, SIEVE OF. In: Encyclopedia of Mathematics. Disponível em:

[13] Fano Experimental Web Server. Seeds (B2/S2). Disponível em: <http://fano.ics.uci.edu/ca/rules/b2s/>. Acesso em 25 de maio de 2014.

[14] GARDNER, M. The fantastic combinations of John Conway's new solitaire game "life". Scientific American, v.223, p. 120-123, 1970. Disponível em: <http://ddi.cs.uni-potsdam.de/HyFISCH/Produzieren/lis_projekt/proj_gamelife/ConwayScientificAmerican.htm>.

[15] GOUCHER, Adam P. Prime Numbers. Game of Life News, fev 2010. Disponível em: <http://pentadecathlon.com/lifeNews/2010/02/prime_numbers.html>. Acesso em: 25 de maio de 2014

[16] GUNNAR, Johnsson. Games of Life in Colour: Introduction. Online, abr, 1999. Disponível em: <http://www.math.kth.se/~gunnarj/LIFE/WLIF/wlcframes.html>. Acesso em 25 de maio de 2014.

[17] JOGO DA VIDA. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2014. Disponível em: <http://pt.wikipedia.org/w/index.php?title=Jogo_da_vida&oldid=38987623>.

[18] JOHNSTON, Nataniel. Generating Sequences of Primes in Conway's Game of Life. Disponível em <http://www.njohnston.ca/2009/08/generating-sequences-of-primes-in-conways-game-of-life/>. Acesso em: 25 de maio de 2014

[19] JUSTO, Marina Jeaneth Machicao. Autômatos celulares caóticos aplicados na Criptografia e Criptoanálise. 2013. Dissertação (Mestrado em Física Aplicada) - Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, 2013. Disponível em: <http://www.teses.usp.br/teses/disponiveis/76/76132/tde-20092013-153518/>. Acesso em: 5 de maio de 2014

[20] LIU, Yan. Modelling Urban Development With Geographical Information System. CRC Press, 2008. 186pp.

[21] LORENZ, Edward N. Deterministic non-periodic flow. 1963. Journal of the Atmospheric Sciences 20 (2): p130–141.

[22] MACHICAO, Marina Jeaneth; MARCO, Anderson G.; BRUNO Ademir G. Caothic Encryption Method based on Life-Like Cellular Automata. Disponível em: <http://www.producao.usp.br/bitstream/handle/BDPI/35478/wos2012-304.pdf>. Acesso em: 25 de maio de 2014

[23] MAPA LOGÍSTICO. In: WIKIPÉDIA, a enciclopédia livre. Flórida: Wikimedia Foundation, 2013. Disponível em: <http://pt.wikipedia.org/w/index.php?title=Mapa_log%C3%ADstico&oldid=34620132>. Acesso em: 25 maio de 2014.

[24] MARTINS, Caroline Collaço. Autômato celular aplicado ao crescimento de câncer. 67p. 2010.Dissertação - Universidade Estadual de Ponta Grossa. Ponta Grossa. Disponível em: <http://www.bicen-tede.uepg.br/tde_arquivos/8/TDE-2010-05-03T145553Z-

[25] NEUMANN, J. V.; BURKS, A. W. Theory of self-Reproducing Automata. University of Illinois Press, 1996.

[26] PASCOAL, F.S.. Sociedade Artificial Fight4Life: Autônamo Celular Modelando Vida Artificial. Relatório Final de PIBIC, INPE. São José dos Campos, 2005. Disponível em:< http://mtc-m16.sid.inpe.br/col/sid.inpe.br/iris%401916/2005/09.30.14.21/doc/sociedade%20artificial.pd

[27] PEGG, E. J. WireWorld. In: MathWorld. Disponível em:<http://mathworld.wolfram.com/WireWorld.html >. Acesso em 28 de maio de 2014.

[28] PFEIFER, R, KUNZ, H., WEBER, M., M. Artificial Life. Institut fr Informatik der Universitt Zrich, 2000.

[29] PRIMER. In LifeWiki. Disponível em: <http://www.conwaylife.com/w/index.php?title=Primer&oldid=13393>. Acesso em 30 de julho de 2014.

[30] QUARTIERI, Joseph; MASTORAKIS, Nikos E.; IANNONE, Gerardo; GUARNACCIA, Claudio. A Cellular Automata Model for Fire Spreading Prediction. In: Latest Trends on Urban PLanning & Transportation. Corfu Island, Greece: M. Jha, jul, 2010. p173-179.

[31] RENDELL, Paul. A Turing machine implemented in Conway`s Game of Life. Online, 2011. Disponível em: <http://rendell-attic.org/gol/tm.htm>. Acesso em: 25 de maio de 2014.

[32] RULETABLE Repository. Life Pattern Emmulators. Online, disponível em:<https://code.google.com/p/ruletablerepository/wiki/TheFormat>. Acesso em 29 de julho de 2014.

[33] RULETABLE Repository. Online, disponível em:<https://code.google.com/p/ruletablerepository/wiki/TheFormat>. Acesso em 27 de maio de 2014.

[34] RULETABLEREPOSITORY. UnboundVarables: Re-enable optional unbound variables in rule-table definitions? Postado em 4 de fevereiro de 2010. Disponível em,:< https://code.google.com/p/ruletablerepository/wiki/UnboundVariables>. Acesso em 28 de julho de 2014.

[35] SASSO, Juliana Bonás; CHIMARA, Henrique Dal Bo; MONTEIRO, Luiz Henrique Alves. Epidemias e modelos epidemológicos baseados em autômatos celulares: uma breve revisão. Caderno de pós-graduação em Engenharia Elétrica. São Paulo, v4, n.1, p.71-80, 2004. DIsponível em: <http://www.mackenzie.com.br/fileadmin/Pos_Graduacao/Mestrado/Engenharia_Eletrica/volume_IV/005.pdf>. Acesso em: 25 de maio de 2014.

[36] SILVER, S. A. Life Lexicon. Release 25, 2006. Disponível em:<http://www.argentum.freeserve.co.uk/lex_r.htm#rpentomino>. Acesso em: 29 de maio de 2014.

[37] SIRAKOULIS, G. C; KARAFYLLIDIS, I.; THANAILAKIS, A. A Cellular Automaton Model for the Effects of Population Movement and Vaccination on Epidemic Propagation. Ecological Modeling, 2000. Disponível em:< http://www.dpi.inpe.br/gilberto/cursos/st-society-2013/Sirakoulis2000.pdf>. Acesso em 28 de maio de 2014.

[38] SZOR, Peter. The Art of Computer Virus Research and Defense. Addison-Wesley Professional, 2005. 744p.

[39] TAVARES, Leonardo Daniel. Um simulador de tráfego baseado em autômatos celulares. 66p. Março de 2010. Dissertação - Universidade Federal de Minas Gerais. Belo Horizonte, Minas Gerais. Disponível em: <http://www.ppgee.ufmg.br/documentos/Defesas/895/Leonardo-Daniel-dis-Rev.pdf>. Acesso em: 25 de maio de 2014.

[40] TJOE Linda, SALIM Lince, LING Suet Chong,LAI Jeslim.The Geometry of War pags. Mathematics in Arts & Architecture. University of Singapore p. 56-69.

[41] TOMASSINI, Marco; SIPPER, Moshe; PERRENOUD, Mathieu. On the Generation of High-Quality Random Numbers by Two-Dimensional Cellular Automata. Disponível em: <http://www.cs.bgu.ac.il/~sipper/papabs/twodrand.pdf>. Acesso em: 25 de maio de 2014.

[42] TWIN PRIME CALCULATOR. In: LifeWiki. Disponível em: <http://www.conwaylife.com/w/index.php?title=Twin_prime_calculator&oldid=12956>. Acesso em: 29 de julho de 2014.

[43] Variations on the Game of Life. Disponível em: <http://kaytdek.trevorshp.com/projects/computer/neuralNetworks/gameOfLife2.htm>. Acesso em: 28 de maio de 2014.

[44] VEJA. Online, nov 2013. Congestionamento em São Paulo é o maior da história. Disponível em: <http://veja.abril.com.br/noticia/brasil/congestionamento-em-sao-paulo-e-o-maior-da-historia>. Acesso em: 25 de maio de 2014.

[45] VERSIGNASSI, A.; Weingrill, N. O Par Perfeito. In Revista Super Interessante. Novembro, 2008. Disponível em <http://super.abril.com.br/ciencia/par-perfeito-447841.shtml>. Acesso em 28 de maio de 2014.

[46] WEISSTEIN, E. W. "Kermack-McKendrick Model." In: MathWorld--A Wolfram Web Resource. Disponível em:< http://mathworld.wolfram.com/Kermack-McKendrickModel.html>. Acesso em 27 de maio de 2014.

ANEXOS

Arquivo PRIMER-LIFE.rle

#N Primer

#C Produces a westward stream of lightweight spaceships representing

#C prime numbers: a LWSS escapes past the pentadecathlon around

#C generation 120n+100 if and only if n is prime.

#C This is probably not a good way to look for the largest primes!

#O Dean Hickerson, dean@math.ucdavis.edu (11/1/1991)

x = 440, y = 294

412b2o$410b2ob2o13b2o$410b4o13b4o$411b2o14b2ob2o$93b3o11b3o319b2o$93bo

2bo10bo2bo$93bo6b3o4bo16b3o11b3o$93bo5bo2bo4bo15bo2bo10bo2bo$94bo4b2ob

o5bo17bo4b3o6bo285b2o$126bo4bo2bo5bo284bo2bo$125bo5bob2o4bo285bo2b2o$

424bo2b2o$99bo8bo315b4o$99bobo5b3o278bo7bo$107bob2o14bo8bo254bo7b2o$

100bo2bo4b3o13b3o5bobo249bo4bo6b2o30b2o$101b3o4b2o13b2obo258b5o29b4o4b

4o$102bo20b3o4bo2bo280bo3bo3bo4b2ob2o$124b2o4b3o282bo6bo6b2o$131bo279b

o3bo5bo$412b4o$91b3o313b3o$90bo2bo311b2ob2o26b2o$93bo46b3o264b3o17b4o

4b4o$93bo46bo2bo268b4o10bo3bo4b2ob2o$92bo47bo270bo3bo14bo6b2o$106b3o3b

3o25bo274bo13bo$105bo2bo3bo2bo25bo272bo18bo$108bo3bo6b3o3b3o253bo49bo$

96b3o9bo3bo5bo2bo3bo2bo253b2o47bo2b2o$98bo8bo5bo7bo3bo255b2o47bo5bo$

97bo12bo10bo3bo9b3o293bobo2b2o$110bo9bo5bo8bo295bo3b2o$109bobo11bo12bo

296b3o$101b3o19bo$76bo13bo12bo18bobo$75b3o11b3o10bo27b3o303b2o$75bob2o

4bo5bob2o37bo12bo13bo277b4o$76b3o3b3o5b3o38bo10b3o11b3o245bo13b2o15b2o

b2o$76b2o3b2o2bo4b2o49b2obo5bo4b2obo244bo13b4o16b2o$81bo3bo24b3o28b3o

5b3o3b3o245b3o11b2ob2o$81bob2o57b2o4bo2b2o3b2o261b2o$81b2o38b3o24bo3bo

$149b2obo213bo$75b3o73b2o214b2o42b2o$74bo2bo288b2o41b2ob2o$77bo78b3o

250b4o$77bo78bo2bo246bo3b2o$76bo79bo248b2o$156bo249bo3b2o$102b3o52bo
251b4o$102bo2bo303b2ob2o$102bo26b3o279b2o$92bo9bo25bo2bo227bo$91b3o8bo
28bo226bo$55b3o11b3o19bob2o8bo27bo9bo216b3o36b4o$54bo2bo10bo2bo20b3o
36bo8b3o253bo3bo$57bo4b3o6bo20b2o36bo8b2obo239b2o16bo$57bo4bo2bo5bo67b
3o209bo21b4o4b4o14bo$56bo4bo8bo69b2o21bo13bo9b2o163b2o18bo3bo4b2ob2o$
97bo5bo58b3o11b3o7b4o161b2o23bo6b2o$96b3o3b3o57bob2o4bo5bob2o6b2ob2o
184bo$62b2o31b2obo3bob2o4b2o18bo5bo26b3o3b3o5b3o8b2o189bo$62bo7bo24b3o
5b3o4bobo16b3o3b3o25b2o3b2o2bo4b2o198bo$69b3o24b2o5b2o6b2o9b2o4b2obo3b
ob2o29bo3bo28bo175bo2b2o$68b2obo49bobo4b3o5b3o29bob2o30bo173bo5bo$68b
3o29bo20b2o6b2o5b2o30b2o28bo3bo174bobo2b2o$69b2o28b3o97b4o174bo3b2o$
98bo3bo30bo28b3o26bo2b3o117bo64b3o$132b3o26bo2bo25b2o2bobo116bo$72bo5b
o52bo3bo28bo26bo2b3o116b3o$71b3o3b3o18b2ob2o61bo34b4o179b2o$53b3o15bob
2ob2obo20bo62bo34bo3bo178b4o25b2o$53bo2bo15b3ob3o52b2ob2o66bo133bo27b
2o15b2ob2o15b4o4b4o$53bo18b2o3b2o33bo20bo67bo135b2o24b4o16b2o15bo3bo4b
2ob2o$53bo57b2o41b3o3b3o16bo29bo126b2o25b2ob2o36bo6b2o$54bo20bo35bobo
7bo31bo2bo3bo2bo14b3o29bo154b2o36bo$74bobo10bo33b2o33bo3bo17b3o25bo3bo
16bo$73bo3bo9b2o31bobo33bo3bo19bo26b4o17bo$58b2o14bobo9bobo57bo8bo5bo
16bobo43bo3bo128b2o49b2o$57bobo14b3o68b2o11bo19bo46b4o126b2ob2o48bobo$
59bo85bobo10bo192bo3b4o49bob2o$157bobo16bo172b2obo3b2o51b2o$82b3o90b2o
92bo79bo3bo55bo$82bo2bo89bobo90bo80b2obo3b2o$82bo66b3o72b3o41b3o80bo3b
4o$82bo65bo2bo74bo128b2ob2o50b2o$83bo67bo72bobo130b2o50b4o$151bo60bo
11b2o166b2o15b2ob2o$150bo59bo2bo177b4o16b2o$210bo2bo129b4o44b2ob2o$
211bo5b2o8bo114bo3bo46b2o$216b4o8bo117bo$216b2ob2o3bo3bo116bo$73b2o
143b2o5b4o156b2o$72bobo308b2ob2o$67b3o4bo304bo3b4o$67bo2bo90bo215b2obo
3b2o$67bo92b2o215bo3bo$67bo92bobo10bo13bo189b2obo3b2o$67bo104b3o11b3o
52bo29bo29bo29bo47bo3b4o$68bo103bob2o4bo5bob2o51b3o27b3o27b3o27b3o49b

2ob2o$173b3o3b3o5b3o54bo29bo29bo29bo50b2o11b2o$173b2o3b2o2bo4b2o54b2o

28b2o28b2o28b2o19bo39b4ob2o$178b2ob2o172bo38b6o$178bo2b3o165bo5bo15b4o

20b4o$179bob2o167b6o14bo3bo29b2o$181bo192bo28b4o$172b3o2bo3bo191bo29b

2ob2o$171bo2bo3bobo224b2o$174bo4bo221bo$174bo71b2o28b2o28b2o28b2o28b2o

31bobo$173bo72bobo27bobo27bobo27bobo27bobo32bo$164b3o80b2o28b2o28b2o

28b2o28b2o36b2o$163bo2bo236b2ob2o$166bo3b3o230b4o$166bo3bo2bo15bo214b

2o$166bo3bo17b3o194b2o8b4o$166bo3bo16b2obo192b2ob2o6b6o$165bo5bo15b3o

135b2o52bo3b4o7b4ob2o$188b2o134b4o46b2o2bobo3b2o12b2o$167b3o154b2ob2o

44bo2b2o3bo$167bobo76b2o28b2o28b2o18b2o46b2o2bobo3b2o$167b3o76bobo27bo

bo27bobo10bobo57bo3b4o$247b2o28b2o22b2o4b2o9bo2b2o24b2o34b2ob2o$301bo

17bobo16b4o4b4o35b2o5b2o5b4o$145b3o11b3o5b3o156b2o9bo3bo4b2ob2o9bo29b

2ob2o3bo3bo$145bo2bo10bo2bo5bo155b2ob2o12bo6b2o11bo28b4o8bo$145bo6b3o

4bo164b4o12bo15bo4bo29b2o8bo$145bo5bo2bo4bo140b3o14bo7b2o30b5o$146bo4b

2obo5bo129b4o5b5o13b2o16bo$289bo3bo5b3ob2o11bobo12bo3bo62b2o$293bo8b2o

26bo3bo9bo52bo2bo$285bo6bo38bo12b6o42b2o3bo2bo$152b2o6bo88bo29bo7b3o

32bo9b2o9bob3o44bo3b2ob2o$152bo6b3o86b2o28b2o5bo6bo29b2o13b2o9bo48b2o$

159bob2o130bo27bobo13bo37b3o$160b3o126bo3bo83bo$107b2o6b2o43b2o128b4o

53b2o27bo$106b4o4b4o228b4o49b4o$106b2ob2o3b2ob2o227b2ob2o33bo13bo3bo$

108b2o6b2o7b2o47b3o120b4o30b2o15b2o35bo16bo$123b2ob2o46bo2bo118bo3bo

28b2ob2o47bo3bo15bo$123b4o16b3o28bo125bo14b4o10b4o49b4o$124b2o16bo2bo

28bo9b2o8bo104bo14bo3bo11b2o$145bo28bo8b4o8bo51bo70bo$145bo29bo7b2ob2o

3bo3bo52bo68bo$120b4o20bo40b2o5b4o48bo3bo$116b2o2bo2b2o120b4o$117bo3bo

2b2o114b3o25b2o39b2o3b2o$121bo2bo113b2ob2o24b4o36bo6bobo$122b2o67bo31b

2o15b3o24b2ob2o34bo2bo6bo$146b2o44b2o30b2o19b4o20b2o34b2o7b3o$81b5o58b

2ob2o44bo29bo20bo3bo5bo51b2o$80bo4bo31bo26b4o13b3o29bo54bo6bo6b2o$85bo

32bo6b2o18b2o7bo6bo2bo27bo54bo3bo3bo4b2ob2o$84bo29bo3bo4b2ob2o24bobo6b

o56b5o29b4o4b4o$108b4o3b4o4b4o26b2o6bo55bo4bo38b2o45b2o5b4o$107bo3bo

12b2o36bo31bo27bo83b2ob2o3bo3bo$111bo83bo25bo33bo50b4o8bo$103b2o2bo2bo

65bo14bo3bo11b2o45bo3b3o46b2o8bo$103b3o71bo14b4o10b4o43bo3bob3o$103b2o

2bo2bo62bo3bo28b2ob2o43bo6b2o$111bo62b4o30b2o15b2o28b5obo$107bo3bo111b

2ob2o29b4o$108b4o89bobo19b4o31bo$167b4o31b2o20b2o12b2o$166bo3bo31bo36b

2o$170bo67bo23b2o$162b2o2bo2bo51b3o20b2o14b2ob2o$162b3o31bobo20bo3b2o

18b4o13b4o$162b2o2bo2bo27b2o20bobo2b2o17b2ob2o13b2o$170bo8b2o16bo20bo

5bo20b2o$166bo3bo5b3ob2o37bo2b2o$167b4o5b5o38bo$177b3o22b2o17bo$201b4o

12bo$201b2ob2o12bo6b2o$194bo2bo5b2o9bo3bo4b2ob2o$185b2o2bo4bo3bo16b4o

4b4o$170b2o12bo2bo2bo2bo4b2o24b2o$169bobo13b2o2b3o2bo3bo$156b2o11b2o

14b3o2bo3bo2bo5b2o$155bobo31bo11b2ob2o$157bo43b4o$202b2o3$200b2o$155bo

43b4o$153bobo31bo11b2ob2o$154b2o11b2o14b3o2bo3bo2bo5b2o$167bobo13b2o2b

3o2bo3bo$168b2o12bo2bo2bo2bo4b2o24b2o$183b2o2bo4bo3bo16b4o4b4o$192bo2b

o5b2o9bo3bo4b2ob2o$199b2ob2o12bo6b2o$199b4o12bo$103bo71b3o22b2o17bo$

103b2o60b4o5b5o38bo$102bobo59bo3bo5b3ob2o37bo2b2o$168bo8b2o16bo20bo5bo

$160b2o2bo2bo27b2o20bobo2b2o$160b3o31bobo20bo3b2o$160b2o2bo2bo51b3o$

168bo$164bo3bo31bo$165b4o31b2o20b2o$105b2o92bobo19b4o$b6o31b2o63b2ob2o

113b2ob2o$o5bo28b3ob2o58bo3b4o65b4o30b2o15b2o$6bo28b5o54b2o2bobo3b2o

65bo3bo28b2ob2o$5bo30b3o54bo2b2o3bo73bo14b4o10b4o$94b2o2bobo3b2o68bo

14bo3bo11b2o$99bo3b4o86bo$103b2ob2o84bo$105b2o5b2o5b4o28b2o$80bo29b2ob

2o3bo3bo27bobo$81bo28b4o8bo20b2o7bo37bo$76bo4bo29b2o8bo20b4o45bo83bo$

77b5o60b2ob2o44bo84bo$144b2o44b2o79bo4bo20b2o8bo$119bo69bo82b5o19b4o8b

o$120bo175b2ob2o3bo3bo$115bo4bo177b2o5b4o$114b2o3b2o$118bo64b2o5b4o$

181b2ob2o3bo3bo90bobo14bobo$181b4o8bo90b2o5b2o6b2o3bo$182b2o8bo92bo5bo

bo5b3obob2o$119b4o170b2o4b3o4bo$104bo13bo3bo169bo9bo3bo$105bo16bo180b

3o$101bo3bo15bo181bo$102b4o$307bo$297b2o9bo$294b3ob2o4bo3bo$294b5o6b4o

$271b2o22b3o$269b2ob2o$269b4o16b2o$270b2o15b2ob2o$287b4o$135b2o151b2o$

134b4o$127bobo4b2ob2o$136b2o15b2o132bo$128bo22b2ob2o94b3o34b2o$128bo

22b4o94b5o32bob2o$128bo23b2o95b3ob2o31bobo$252b2o32b2o2$128bo22bo$128b

o22b2o128bo$128bo21bob2o113b2o13bo6b2o$150bobo113bobo9bo3bo4b2ob2o3b2o

8bo$127bobo20b2o116bo10b4o4b4o3b4o8bo$288b2o4b2ob2o3bo3bo$296b2o5b4o$

145bo126b2o$146bo6b2o116bobo$142bo3bo4b2ob2o117bo$125b2o16b4o4b4o146b

2o$123b2ob2o24b2o146b2ob2o$123b4o36b2o112b2o22bo2bo$124b2o36b4o110bobo

22bo2bo$162b2ob2o111bo23b2o$164b2o$157bobo$156bo2b2o121b2o21bo$157bobo

121bobo22bo$164b2o117bo3bo14bo3bo$162b2ob2o3b2o8bo105bo3bo12b4o$162b4o

3b4o8bo104bo2b2o$163b2o4b2ob2o3bo3bo109bo$136b5o30b2o5b4o105b2o$135bo

4bo$140bo$139bo$177bo$178b2o$179bo$179bo$178bo3$180bo$181bo$177bo3bo$

161b4o13b4o$160bo3bo$164bo$163bo!

Warlife.table

```
# WarLife

# Desenvolvido por LeandroZoucas@gmail.com e Pedro.Oliveira@uniriotec.br, orientados pela professora Flavia.Santoro@uniriotec.br

# Totalmente inspirado no Immigration Game, que por sua vez baseia-se no Game of Life de Conway

# O código do Immigration foi acessado em http://boardgamegeek.com/filepage/82102/immigration-ruleset-for-golly no dia 2014.05.18.


# Para usar, coloque este arquivo no diretório RULES dentro do diretório principal do Golly.

# Depois, vá no menu CONTROL/SET RULE e entre com:

# Warlife:T15,15 para toroidal ou

# Warlife:P15,15 para plano.
```

```
n_states:4
neighborhood:Moore
symmetries:permute

var a={1,2}
var b={1,2}
var c={1,2}
var d={1,2}
var e={1,2}
var w={0,1,2}
var x={0,1,2}
var y={0,1,2}
var z={0,1,2,3}
var j={0,1,2,3}
var k={0,1,2,3}
var l={0,1,2,3}
var m={0,1,2,3}
var n={0,1,2,3}
var o={0,1,2,3}
var p={0,1,2,3}

#  C,N,NE,E,SE,S,SW,W,NW,C' for the Moore neighborhood

# NASCIMENTO
0,a,1,1,0,0,0,0,0,1
0,a,2,2,0,0,0,0,0,2
```

SUPERPOPULACAO

a,b,c,d,e,w,x,y,z,0

SOLIDAO

a,z,0,0,0,0,0,0,0,0

SOBREVIVE COM 2 VIZINHOS MANTENDO SEU ESTADO

1,a,b,0,0,0,0,0,0,1

2,a,b,0,0,0,0,0,0,2

SOBREVIVE COM 3 VIZINHOS MANTENDO SEU ESTADO

1,a,b,c,0,0,0,0,0,1

2,a,b,c,0,0,0,0,0,2

SUPERBOMBA

e,3,z,j,k,l,m,n,o,0

3,z,j,k,l,m,n,o,p,0