Análise das Bacias de Atração dos Autômatos Celulares

Bernardo Siqueira Esteves dos Reis e Eurico Luiz Próspero Ruivo

Faculdade de Computação e Informática - Universidade Presbiteriana Mackenzie (UPM)
São Paulo - SP - Brazil

bernardo.sereis@gmail.com, eurico.ruivo@mackenzie.br

ABSTRACT

Cellular Automata (CAs) are dynamic structures that have been studied not only for their mathematical and computational interest, but also for their representation in models of complex systems in the real world, such as in the proliferation of diseases, fluid dynamics and growth. urban, as well as by solving decision problems.

They are represented by lattices composed of cells, which can assume different states according to their local transition rules. Traditionally, this rule updates all cells at the same time, that is, it acts synchronously. However, this approach presents limitations in the representation and solution of real and decision problems, therefore, for this study, updating by neighborhood priority was used. This update prioritizes the execution of each cell according to its neighborhood and not its position in the grid.

Updating a rule using neighborhood priority has the same effect and form as synchronously updating another specific rule. By finding the equivalent rule of an update scheme with neighborhood priority, it is possible to develop the basin of attraction of this synchronous rule.

The Basin of Attraction represents the initial configurations of the state space of an CA, which converge in accordance with the CArules, to a certain stable state. This study analyzed the basins of attraction of CAs, finding patterns in their configurations and evolutions that solve decision problems, such as determining long-term behaviors, stability and classification of CAs, evaluating their ability to resolve these problems.

Keywords: Cellular Automata, synchronous, neighborhood priority, basin of attraction, decision problems.

RESUMO

Os Autômatos Celulares (ACs) são estruturas dinâmicas que têm sido estudados não apenas pelo interesse matemático e computacional, mas também tanto pela sua representação em modelos de sistemas complexos reais, como por exemplo na proliferação de doenças, dinâmica de fluídos e o crescimento urbano, como pela solução de problemas de decisão.

São representados por reticulados compostos por células, que podem assumir diversos estados de acordo com suas regras de transição local. Tradicionalmente esta regra atualiza todas as células ao mesmo tempo, ou seja, ela age de forma síncrona. Porém esta abordagem apresenta limitações na representação e solução de problemas reais e de decisão, portanto para este estudo utilizou a atualização por prioridade de vizinhança. Esta atualização prioriza a execução de cada célula de acordo com sua vizinhança e não sua posição no reticulado.

Atualizar uma regra utilizando a prioridade de vizinhança possui o mesmo efeito e forma que atualizar de forma síncrona uma outra regra específica. Ao encontrar a regra equivalente de um esquema de atualização com a prioridade de vizinhança é possível desenvolver a bacia de atração desta regra síncrona.

A Bacia de Atração representa as configurações iniciais do espaço de estados de um AC, as quais convergem em conformidade com as regras do AC, para um determinado estado estável. Este estudo analisou as bacias de atração dos ACs, encontrando padrões em suas configurações e evoluções que resolvam problemas de decisão, como determinar comportamentos de longo prazo, estabilidade e classificação dos ACs, avaliando a sua capacidade de resolução destes problemas.

Palavras-chave: Autômatos Celulares, síncrona, prioridade de vizinhança, bacia de atração, problemas de decisão.

1 INTRODUÇÃO

1.1 Contextualização e problema de pesquisa

Esta pesquisa trata dos Autômatos Celulares (ACs) (KARI, 2005; WOLFRAM, 2002) e suas bacias de atração (WUENSCHE, A.; LESSER, M., 1992). Os ACs são estruturas dinâmicas, que podem modelar diferentes sistemas complexos do mundo real. São representados por reticulados compostos por unidades chamadas de células, que podendo assumir diversos estados, são conectadas em uma malha regular (denominada reticulado) e sujeitas a uma função (ou regra) local que determina o próximo estado de cada unidade com base em seu estado atual e no estado das células vizinhas a ela.

Normalmente esta regra age de forma síncrona sobre as células, atualizando elas em um passo de tempo. O desafio desta atualização encontra-se na sua limitação para representar modelos reais e resolver problemas de decisão. Portanto este trabalho utiliza-se da atualização por prioridade de vizinhança, que prioriza a execução de cada célula de acordo com sua vizinhança e não sua posição no reticulado. E especialmente utilizando esquemas de atualização múltipla, que permite que as células sejam atualizadas diversas vezes durante a execução de um passo de tempo.

Em termos gerais, as bacias de atração representam regiões do espaço de estados em um autômato celular para as quais as configurações iniciais convergem ao longo do tempo, seguindo as regras de transição local e as interações entre as células, até se estabilizarem em um estado estável. Estas configurações e transições podem apresentar padrões que resolvem problemas de decisão.

As trajetórias de estado que se estabilizam em certos conjuntos de estados, representam pontos de atração no espaço de estados dos ACs. O estudo das bacias de atração permite compreender a dinâmica e o comportamento complexo dos ACs, fornecendo análises sobre como diferentes regras locais e interações entre as células influenciam a evolução do sistema.

1.2 Objetivo geral da pesquisa

Esta pesquisa se baseia no estudo de Thiago De Mattos (2022), o qual explorou as bacias de atração dos ACs do espaço de regras com esquemas de atualização múltipla por vizinhança.

O objetivo principal desta pesquisa é identificar quais regras dos ACs apresentam bacias de atração com padrões que possam resolver problemas de decisão, estes podem ser: problemas de classificação, previsão de comportamentos em longo prazo. Isso envolve analisar e compreender a evolução das diferentes configurações iniciais ao longo do tempo, por meio da observação dos padrões resultantes das configurações finais das bacias de atração.

Ao explorar estas características das bacias de atração dos ACs, é possível descobrir a capacidade dos ACs em produzir soluções estáveis e coerentes para diferentes problemas de decisão.

Nas seções a seguir encontra-se, no Capítulo 2, o referencial teórico, mencionando os conceitos utilizados neste trabalho, definindo os ACs e suas bacias de atração. Em sequência, no Capítulo 3, a metodologia explicando o que foi necessário e feito para a análise. Nos resultados e discussão, no Capítulo 4, são apresentados os resultados dos experimentos, apresentando uma tabela e uma análise dos principais padrões encontrados. No Capítulo 5, encontram-se as considerações finais do trabalho. E por fim, no Capítulo 6, as referências utilizadas nesta pesquisa.

2 REFERENCIAL TEÓRICO

2.1 Autômatos Celulares

Acredita-se que desde os primeiros estudos sobre os ACs que eles podem explicar e modelar fenômenos do mundo real a partir de suas simples estruturas que agem localmente com suas propriedades matemático-computacionais (WOLFRAM, S.). Isto pode ser comprovado, como dito anteriormente, pelo uso deles na simulação da propagação de doenças, dinâmica de fluidos e crescimento urbano (MELOTTI, G.).

Representados por reticulados compostos por células, as quais são conectadas entre si em relações de vizinhança que sujeitas à regra local são atualizadas de acordo com seus estados atuais e os estados das células vizinhas.

Para demonstrar suas capacidades matemático-computacionais podemos ver que há regras de casos mais simples como o dos autômatos celulares elementares (ACEs), que possuem apenas dois possíveis estados para cada célula, um exemplo disto é a Regra 110 e suas equivalentes dinâmicas, capazes de simular uma máquina de Turing universal (COOK, 2004).

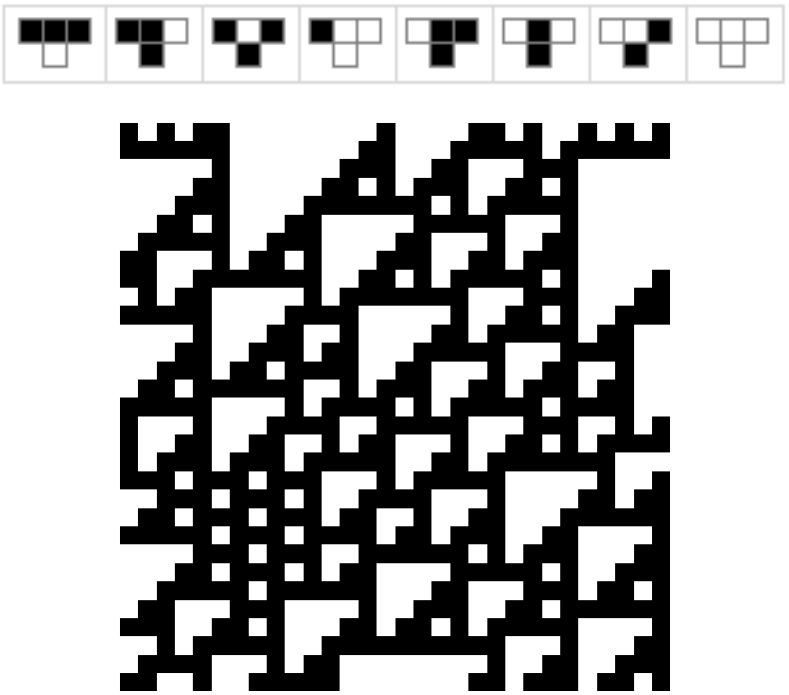

Figura 1: Acima a regra elementar 110 e abaixo a evolução desta com atualização síncrona, considerando quadrados brancos como 0 e os pretos como 1.

Existem diversos tipos de atualização dos ACs, tradicionalmente a atualização utilizada é a síncrona que também desenvolveu a regra 110 na figura 1, este tipo de atualização atualiza todas as células de acordo com a regra das vizinhanças ao mesmo tempo. Porém pode ser limitada para a solução de problemas de decisão Thiago De Mattos (2022), portanto utilizou-se a uma abordagem assíncrona, que proporciona uma maior liberdade e variabilidade nas execuções. Esta abordagem é a atualização por prioridade de vizinhança.

Este esquema de assincronia prioriza a atualização de cada célula de acordo com a prioridade de sua vizinhança. Por exemplo uma atualização com a prioridade (2, _, 1, _, _, _, 2, _), primeiro são atualizadas de forma síncrona as vizinhanças de prioridade 1 e depois são atualizadas as de prioridade 2 e diante, quando todas as prioridades forem executadas completa-se um passo do tempo da evolução do AC. Na prioridade o "_" significa que esta vizinhança não é ativa, logo não é considerada.

Nesta atualização por prioridade existem dois tipos de esquemas de atualização, a única, que permite apenas que as células sejam atualizadas uma vez em um passo de tempo e o esquema de atualização múltipla, que permite que as células sejam atualizadas diversas vezes durante a execução de um passo de tempo. Esta última foi a utilizada neste trabalho.

O uso da atualização por prioridade de vizinhança em uma regra elementar possui o mesmo efeito e forma que aplicar regras específicas de forma síncrona. Por exemplo: ao aplicar a atualização por prioridade de vizinhança (2, _, 1, _, _, _, 2, _) na regra 110, é o mesmo que aplicar de forma síncrona a regra 270274812, portanto aplicar a prioridade à regra 110, sua execução se torna equivalente à execução síncrona da regra 270274812.

2.2 Bacias de Atração

Uma bacia de atração de um autômato celular pode ser representada por grafos direcionados em que são definidas as possíveis transições de estados dos autômatos até se estabilizarem no nó central.

Foram exploradas, no trabalho de Wuensche e Lesser (1992), as bacias de atração do espaço de regras utilizando esquemas de atualização por vizinhança, tanto múltiplos quanto únicos. Esses esquemas de atualização pertencem a regras que poderiam potencialmente resolver problemas de decisão. E foi neste trabalho que Thiago De Mattos (2022) serviu de inspiração para seu estudo acerca das bacias de atração.

No trabalho de Wuensche e Lesser (1992) foram exploradas as bacias de atração do espaço de regras utilizando esquemas de atualização por vizinhança, tanto múltiplos quanto únicos. Esses esquemas de atualização pertenciam a regras que poderiam potencialmente resolver problemas de decisão. E foi neste trabalho que Thiago De Mattos (2022) serviu de inspiração para seu estudo acerca das bacias de atração.

Nas bacias de atração de cada regra existem dois componentes, grafos em que seus nós de atração são associados à configuração $X = 000 \cdots 0$ (que será tratado como 0), ou seja, os nós da configuração tendem ao valor 0, e os grafos associados à configuração $X = 111 \cdots 1$ (que será tratado como 1), em que seus nós tendem ao valor 1. E estes dois componentes geralmente são o complemento de um e do outro, ou seja, os padrões de 1 podem ser o oposto dos padrões dos atratores de 0.

A partir desta parte cada regra será representada a partir de seu esquema de atualização origem, pois seus valores crescem exponencialmente quando não seguimos o limite de 255 regras, como por exemplo a regra 4291360968 é referente ao esquema de atualização (_, _, 2, 2, _, 1, 2, _).

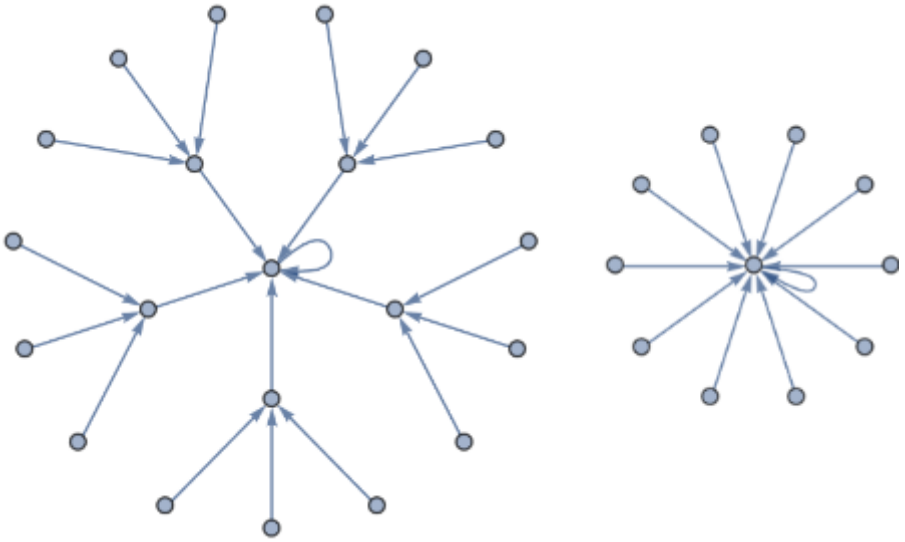

Figura 2: Representação visual de uma bacia de atração de um autômato celular binário (atualização (_, _, 2, 3, _, 1, 2, _) de tamanho 5.). Cada um dos nós representa uma configuração binária. Os arcos representam as transições de estado de uma configuração para outra, considerando a regra do autômato celular em questão. O primeiro grafo são atratores para 1 e o segundo atratores para 0

Nas bacias separadas para este estudo, foi observada uma característica comum: cada configuração de estado do autômato celular tinha uma correspondência com a mesma forma, estrutura e elementos, porém rotacionada. Por exemplo, a configuração (110) era equivalente à configuração (011) após uma rotação para direita.

2.3 Problemas de decisão

Um problema de decisão é um problema que envolve tomar decisões sobre a aceitação ou rejeição de valores de entradas de acordo com um conjunto de condições ou regras pré definidas. Pode ser definido como uma pergunta de sim ou não. Dada uma entrada específica, o objetivo é determinar se essa entrada satisfaz ou não certas condições.

3 METODOLOGIA

Para os experimentos computacionais desta pesquisa, foi utilizado o software Wolfram Mathematica, em particular sua Wolfram language e a biblioteca de funções de ACs e redes de autômatos CAMaT.nb (OLIVEIRA, P. P. B. de.).

Para construir as bacias de atração para o estudo primeiramente foram criadas duas funções para o auxílio deste estudo, pelo professor e orientador Eurico Luiz (BinaryNeighbourhoodListToRnum e BinaryUdsNCompositeRule), estas funções que utilizadas em conjunto, ao inserir um esquema de atualização origem é retornado a regra síncrona equivalente.

Nestes experimentos foram separadas 37 esquemas de atualização múltipla por prioridade de vizinhança, as mesmas que foram utilizadas para o estudo em Thiago De Mattos (2022). Estes esquemas são passados nas funções mencionadas anteriormente para extrair as suas regras equivalentes, que após isto são desenvolvidas suas Bacias de Atração.

Para a análise, as bacias de atração desenvolvidas são transformadas em uma lista de vértices, e aproveitando da característica apresentada anteriormente em que diversos ramos das árvores das bacias são equivalente por rotação, foi criada uma função que percorre a lista de vértices removendo os que são equivalentes por rotação.

Por exemplo ao aplicar esta função no atrator: ((011), (101), (110), (111)) é retornada por ela esta configuração reduzida e simplificada: ((110), (111)).

Aplicando-a em diversas bacias com diversos tamanhos foi possível reduzir em grande escala a complexidade das bacias, especialmente quando são executadas com tamanhos maiores, o que facilita a sua análise para encontrar os padrões nas configurações.

Após os vetores finais serem obtidos, estes foram um por um analisados a olho em diversos tamanhos para encontrar padrões em suas sequências que podem resolver problemas de decisão.

4 RESULTADOS E DISCUSSÃO

Ao analisar todas as 37 configurações uma por uma, aplicando diversos tamanhos e após passá-las pela função para remover configurações com equivalência por rotação, foram encontrados os padrões na tabela a seguir:

Tabela 1: Exemplos de sequências de configurações das bacias de atração de tamanho 5 para cada atualização inicial.

Atualização Inicial	Atrator para 1	Atrator para 0
(_, _, 2, 2, _, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 2, 2, _, 1, _, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, 3, 2, 3, 3, 1, 2, _)	((01111) (11111)) Sequências com no máximo um bit 0	((00000) (00001) (00011) (00101) (00111) (01011)) Sequências com mais de um bit 0
(_, _, 1, 3, 3, 2, 3, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências com mais de três bits 0
(_, _, 1, 3, _, 2, 3, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências com mais de três bits 0
(_, _, 1, 3, _, 2, _, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências com no mais de três bits 0

(_, _, 2, 3, _, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 2, _, 3, 1, 3, _)	((01111) (11111)) Sequências com no máximo um bit 0	((00000) (00001) (00011) (00101) (00111) (01011)) Sequências com mais de um bit 0
(_, _, 3, 2, 2, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 3, 2, _, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 3, 2, _, 1, _, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 3, 3, 1, 2, 3, _)	((00111) (01111) (11111)) Sequências com no máximo dois bits 0	((00000) (00001) (00011) (00101)) Sequências com mais de dois bits 0
(_, _, 3, _, 2, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, _, 1, 2, 3, 1, _)	((00001) (00011) (01011) (00111) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (01001)) Sequências sem bits 1 adjacentes
(_, _, _, 2, 3, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes

(_, 3, 2, 1, 1, 4, 1, _)	((00001) (00011) (00101) (0011) (01011) (01111) (11111)) Sequências com no máximo quatro bits 0	((00000)) Sequências com mais de quatro bits 0
(_, 4, 1, 3, 2, 5, 2, _)	((01011) (01111) (11111)) Sequências com no máximo dois bits 0	((00000) (00001) (00011) (00101)) Sequências com mais de dois bits 0
(_, 4, 3, 2, 2, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, 4, 3, 4, 4, 1, 2, _)	((00111) (01111) (11111)) Sequências com no máximo dois bits 0	((00000) (00001) (00011) (00101)) Sequências com mais de dois bits 0
(_, 4, 5, 4, 1, 2, 3, _)	((00111) (01111) (11111)) Sequências com no máximo dois bits 0	((00000) (00001) (00011) (01001)) Sequências com mais de dois bits 0
(_, 5, 2, 4, 3, 1, 3, _)	((01111) (11111)) Sequências com no máximo um bit 0	((00000) (00001) (00011) (00101) (00111) (01011)) Sequências com mais de dois bits 0
(_, 5, 4, 3, 1, 2, 1, _)	((01111) (11111)) Sequências com no máximo um bit 0	((00000) (00001) (00011) (00101) (00111) (01011)) Sequências com mais de um bit 0
(_, 5, 4, 5, 1, 2, 3, _)	((01111) (11111)) Sequências com no máximo um bit 0	((00000) (00001) (00011) (00101) (00111) (01011)) Sequências com mais de um bit 0
(_, _, 1, 3, 4, 2, 3, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências com mais de três bits 0

(_, _, 1, 4, 2, 3, 4, _)	((00101) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 1, 4, 3, 2, 3, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências sem bits 1 adjacentes
(_, _, 1, 4, _, 2, 3, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências com mais de três bits 0
(_, _, 2, 1, 3, 4, 5, _)	((00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo três bits 0	((00000) (00001)) Sequências com mais de três bits 0
(_, _, 2, 4, 1, 3, 4, _)	((00111) (01011) (01111) (11111)) Sequências com no máximo dois bits 0	((00000) (00001) (00011)) Sequências com mais de dois bits 0
(_, _, 3, 1, 2, 4, 1, _)	((00001) (00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo quatro bits 0	((00000)) Sequências com mais de quatro bits 0
(_, _, 3, 2, 4, 1, 2, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 4, 1, 1, 2, 3, _)	((00001) (00011) (00111) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (01001)) Sequências sem bits 1 adjacentes

(_, _, 4, 3, 1, 2, 3, _)	((00111) (01111) (11111)) Sequências com no máximo dois bits 0	((00000) (00001) (00011) (00101)) Sequências com mais de dois bits 0
(_, _, 5, 1, 2, 3, 4, _)	((00011) (00111) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001)(00101)) Sequências sem bits 1 adjacentes
(_, 4, 3, 1, 2, 5, 2, _)	((00001) (00011) (00101) (00111) (01011) (01111) (11111)) Sequências com no máximo quatro bits 0	((00000)) Sequências com mais de quatro bits 0
(_, 5, 4, 2, 3, 1, 3, _)	((00011) (00111) (01011) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes
(_, _, 5, 4, 2, 3, 1, _)	((00011) (00111) (01111) (11111)) Sequências com bits 1 adjacentes	((00000) (00001) (00101)) Sequências sem bits 1 adjacentes

Antes de comentar sobre a tabela, é importante ressaltar, que os atratores que vão para 1 e 0 são complementos um do outro, logo, os seus padrões tendem a ser opostos. Por exemplo, quando ocorre o padrão “Sequências com bits 1 adjacentes” no atrator para 1, sempre foi observado que o atrator para 0 possuía o padrão inverso: “Sequências sem bits 1 adjacentes”.

Observando os resultados da tabela e seus exemplos pode-se retirar dela alguns padrões, como o mencionado anteriormente que resolve um problema de adjacência: “Sequências com bits 1 adjacentes”, que ocorre no atrator para 1 no esquema de atualização inicial (_, _, 2, 2, _, 1, 2, _) e o oposto ocorre no atrator para 0.

Também apareceram configurações, com certas particularidades interessantes, por exemplo a regra com atualização de origem (_, _, 4, 1, 1, 2, 3, _), possui valores 1

intercalados entre zeros, com ou sem valores 1 adjacentes, como mostrado a seguir a configuração do atrator para 0 desta atualização com tamanho 10:

((0000000000), (0101010101),
(0101011011), (0101101011))

E de forma muito similar ao último padrão descrito, tiveram poucas atualizações iniciais em que existiam dois padrões explícitos: “Sequências sem bits 1 adjacentes e sempre com no mínimo 2 bits de distância”, resolvendo um problema de decisão sobre distância de bits entre valores de 1, que é o caso da atualização (_, _, 1, 3, 3, 2, 3, _), que é demonstrada a seguir com seu atrator para 0 com tamanho 10:

((0000000000), (0000000001), (0000001001),
(0000010001), (0000100001), (0001001001))

E também houve configurações em que o padrão resolve o problema de quantidade de bits 0, que é o caso da atualização (_, 3, 2, 3, 3, 1, 2, _), nos atratores para 1 só existem sequências com no máximo um bit 0 e nos atratores para 0 sequências com mais de um bit 0.

5 CONSIDERAÇÕES FINAIS

Os estudos neste trabalho mostram a variedade de comportamentos que surgem em sistemas dinâmicos complexos dos ACs, oferecendo as análises realizadas em várias bacias e configurações que foram estudadas.

Foi possível identificar padrões distintos em diferentes bacias de atração, destacando alguns padrões comuns: configurações que sempre possuem sequências com bits 1 adjacentes e aquelas que nunca os possuem, distância entre bits 1, e sequências que são definidas pela quantidade de bits 0.

Além disso, é percebido a influência do tamanho das configurações, desde que em certas configurações, os padrões só se tornaram evidentes a partir de tamanhos específicos. Portanto, é importante considerar o tamanho de uma configuração ao realizar análises.

Ainda existem diversas possibilidades de exploração das bacias de atração dos ACs, vale ressaltar que foi utilizado apenas esquemas de atualização por prioridade de vizinhança múltiplas, seria de interesse o estudo das bacias com esquemas de atualização única.

Por fim, pode-se destacar o desafio na análise e busca por padrões das bacias de atração. Mesmo com funções para remover configurações com rotações equivalentes, identificar padrões em algumas delas continuou sendo um desafio. O que mostra a complexidade dos comportamentos dos ACs e o estudo de suas bacias de atração.

6 REFERÊNCIAS

ABURAS, M. M. The simulation and prediction of spatio-temporal urban growth trends using cellular automata models: A review. International Journal of Applied Earth Observation and Geoinformation, Elsevier, v. 52, p. 380–389, 2016.

ARACENA, J. On the number of update digraphs and its relation with the feedback arc sets and tournaments. Discrete Applied Mathematics, Elsevier, v. 161, n. 10, p. 1345–1355, 2013.

ARACENA, J. Combinatorics on update digraphs in Boolean networks. Discrete Applied Mathematics, v. 159, n. 6, p. 401–409, 2011.BERSINI, H.; DETOURS, V.

COOK, M. Universality in elementary cellular automata. Complex Systems, v. 15, n. 1, p. 1–40, 2004.

GARDNER, M. Mathematical games: The fantastic combinations of John Conway's new solitaire game "Life". Scientific American, v. 223, n. 4, p. 120–123, 1970.

HOEKSTRA, A.; KROC, J.; SLOOT, P. Introduction to modeling of complex systems using cellular automata. Simulating Complex Systems by Cellular Automata, Springer, p. 1–16, 2010.

KARI, J. Theory of cellular automata: A survey. Theoretical Computer Science, Elsevier, v. 334, n. 1–3, p. 3–33, 2005.

LANGTON, C. Computation at the edge of chaos: Phase transitions and emergent computation. Physica D: Nonlinear Phenomena, Elsevier, v. 42, n. 1-3, p. 12–37, 1990.

LI, W.; PACKARD, N. The structure of the elementary cellular automata rule space. Structure, v. 4, p. 281–297, 1990.

MELOTTI, G. Aplicação de Autômatos Celulares em Sistemas Complexos: Um Estudo de Caso em Espalhamento de Epidemias, p. 43-67, 2009.

MIKLER, A. R.; VENKATACHALAM, S.; ABBAS, K. Modeling infectious diseases using global stochastic cellular automata. Journal of Biological Systems, World Scientific, v. 13, n. 04, p. 421–439, 2005.

MITCHELL, M. Computation in cellular automata: A selected review lattice. NonStandard Computation, p. 1–42, 1998.

NEUMANN, J. V. Theory of self–reproducing automata. IEEE Transactions on Neural Networks, v. 21, n. 1, p. 3–14, 1966.

NOGUEIRA, M. A. Classificação automática do comportamento dinâmico de autômatos celulares binários. Tese (Doutorado) — Universidade Presbiteriana Mackenzie, 2019.

OLIVEIRA, P. P. B. de. Cellular Automata Mathematica Toolbox – CAMaT, Universidade Presbiteriana Mackenzie, p. 87-110 , 2019.

DE MATTOS, T. Análise do espaço elementar de autômatos celulares com atualização por prioridade de vizinhança.Universidade Presbiteriana Mackenzie, 2022.

WOLF-GLADROW, D. A. Lattice-Gas Cellular Automata and Lattice Boltzmann Models. [S.l.]: Springer, 2000.

WOLFRAM, S. Computation theory of cellular automata. Communications in Mathematical Physics, v. 96, n. 1, p. 15–57, 1984.

WOLFRAM, S. Twenty problems in the theory of cellular automata. Physica Scripta, Institute of Physics Publishing, T9, n. 3, p. 170–183, 1985.

WOLFRAM, S. A New Kind of Science. [S.l.]: Wolfram Media, 2002.

WUENSCHE, A.; LESSER, M. The Global Dynamics of Cellular Automata: An Atlas of Basin of Attraction Fields of One-Dimensional Cellular Automata. [S.l.]: Addison-Wesley, 1992.