# Três Experimentos com Atualização Assíncrona por Prioridade de Vizinhança em Autômatos Celulares Elementares

Marcelo Vironda Rozanti
Felipe Stefanelli de Aguiar Silva
Pedro Paulo Balbi (Orientador)
Faculdade de Computação e Informática - Universidade Presbiteriana Mackenzie
São Paulo, SP – Brasil

***Abstract.*** *Cellular automata are discrete computational systems that have proved useful as general models of complexity and representation of systems' dynamics in a variety of scientific fields. These systems compute essentially by means of local functions and can even display universal computability. Here we report on the exploration of a fundamental subset of them, the elementary cellular automata, with a specific kind of asynchronous update scheme based on the update priority of the neighbourhood configurations that define the local function. We study the effect of this kind of update on those rules, in terms of their becoming number conserving, as well as solving the decision problems of density and parity classification.*

***Resumo.*** *Autômatos celulares são sistemas computacionais discretos que se têm provado úteis como modelos genéricos de complexidade e representação da dinâmica de sistemas em uma variedade de áreas. Estes sistemas computam essencialmente por meio de funções locais, e podem inclusive manifestar computabilidade universal. Relata-se aqui os resultados da exploração computacional de um conjunto fundamental deles, os autômatos celulares elementares, através de um tipo específico de atualização assíncrona, baseada em prioridades das configurações de vizinhança que definem a função local. Estuda-se o efeito desse esquema de atualização na capacidade desses autômatos se tornarem conservativos, e de resolver os problemas de decisão de determinação de maioria e de paridade.*

## 1. Introdução

Autômatos Celulares (ACs) são uma categoria de sistemas discretos. Por sua própria simplicidade, estes sistemas têm ocupado uma posição privilegiada no estudo de complexidade nos mais diversos campos da ciência. O desenvolvimento de ACs é tipicamente atribuído a John von Neumann através de suas tentativas de criar um modelo abstrato de autorreprodução biológica. Ao fim de 1950, foi notado que ACs poderiam ser vistos como computadores paralelos [Wolfram 2002, p. 876]. Apesar de falta de investigação científica até 1970, um exemplo de AC se tornou muito famoso por seu comportamento complexo e regras simplesmente descritas inventadas por John Conway,

*The Game of Life*, que se popularizou após sua aparição na revista *Scientific American* [Gardner 1970].

Desde então, múltiplos estudos foram realizados por cientistas ao redor do mundo, destacando os trabalhos feitos por Stephen Wolfram na década de 1980, culminando na publicação do livro A New Kind of Science em que Wolfram apresenta uma coleção de resultados a respeito de ACs.

Tratamos aqui de uma pesquisa exploratória, envolvendo Autômatos Celulares Elementares com um tipo especíifico de atualização assíincrona, conforme se apresentaría na Subseção 2.3. A seguir, são apresentados conceitos e experimentos relacionados a Autômatos Celulares Elementares assícronos.

### 1.1. Contextualização e Relevância

A motivação inicial para o estudo de ACs por parte de von Neumann não era matemática mas tinha como diretriz encontrar uma forma viável de tratar do problema de como fazer máquinas se reproduzirem. Hoje é sabido que esses sistemas são aplicáveis para resolver problemas relativos a classificação de densidade [Mitchell et al. 1994], simulação de partículas [Salem et al. 1985], compressão de dados [Lafe 1997], design estrutural [Abdalla and Gurdal 2002], representação de comportamentos dinâmicos como trânsito em cidades ou movimentação de seres vivos [Rosenblueth and Gershenson 2011], entre muitos outros.

Apesar de ACs serem executados tradicionalmente em modo síncrono, há múltiplos tipos de atualização de células, que produzem comportamentos diferentes dos síncronos. Comportamentos interessantes podem desaparecer em modo assíncrono mas outros podem emergir. Quando executados de forma assíncrona, ACs definem vários esquemas possíveis de atualização, como o esquema independente aleatório, o esquema de ordem aleatória, o esquema cíclico, entre outros.

O tipo de assincronia usado nesta pesquisa, a atualização assíncrona por prioridade da vizinhança, é novo e ainda pouco estudado. Sua definição é aprofundada na Subseção 2.3.

### 1.2. Objetivos do Estudo

A motivação para o presente trabalho é estudar três importantes propriedades nos autômatos celulares elementares (ACEs) executados com o tipo de atualização em pauta:

- Conservabilidade numérica;
- Capacidade de classificação de densidade;
- Capacidade de resolver o problema da paridade.

O primeiro objetivo trata-se de verificar se a regra 184, que é conservativa em modo síncrono, se mantém conservativa em algum esquema de atualização assíncrono, conforme a definição em de conservabilidade detalhada na Subseção 2.4.

Considerando que é fato conhecido que nenhum AC síncrono é capaz de classificar densidade perfeitamente, o objetivo do segundo experimento foi verificar se algum ACE poderia adquirir essa capacidade em modo assíncrono.

É também um fato conhecido que o problema da paridade, detalhado na Subseção

2.6, pode ser resolvido por ACEs executados em modo síncrono. O objetivo do terceiro experimento foi verificar se ACEs executados com a estratégia descrita na Subseção 2.3 ainda seriam capazes de resolver este problema e em quais esquemas de atualização assíncrona por prioridade da vizinhança.

Três objetivos instrumentais fundamentais para a conclusão deste trabalho foram:

- Definir AC e ACE;
- Definir formalmente a estratégia de atualização;
- Definir e identificar a propriedade e problemas estudados;

## 2. Referencial Teórico

Em relação ao referencial teórico necessíario, foi de suma importância entender os seguintes tópicos:

### 2.1. Autômatos Celulares

De acordo com [Wolfram 1983], "sistemas matemáticos de simples construção são capazes de comportamento altamente complexo com muitas características". ACs são um exemplo desse tipo de sistemas. ACs são compostos por células, cada uma com um estado. Eles podem ter $d$ dimensões e $q$ estados finitos, os quais devem ser estipulados previamente e são necessariamente computados através de funções locais, isto é, que atuam considerando a vizinhança das células.

Wolfram afirma ainda que, autômatos celulares são simples idealizações matemíaticas de sistemas naturais. Eles consistem de um reticulado de com posições discretas idênticas, cada uma assumindo um conjunto finito de valores inteiros. Os valores dos locais evoluem em fases temporais, ou iterações, discretas de acordo com regras determiníisticas que especificam o valor de cada cíelula em termos dos valores das cíelulas adjacentes. Todo AC depende de uma *configuração inicial*, que íe um reticulado de necessariamente $d$-dimensções com cada cíelula com um estado pertencendo ao conjunto de $q$ estados.

### 2.2. Autômatos Celulares Elementares

ACEs formam a família de ACs com $d$=1, $q$=2 e $r$=1 (i.e., raio unitíario), ou seja, são unidimensionais, com células binárias e vizinhanças definidas como 3 células contíguas. O reticulado é tem comprimento $n$, isto é, tem $n$ células.

Assim como ACs, as regras de transição de um ACE definem o novo estado de uma determinada célula em função de sua vizinhança. O comportamento da aplicação das regras de transição nas bordas do reticulado pode ser especificado de várias formas, muitas vezes como conexo com a borda oposta, tornando o reticulado efetivamente circular e contínuo, que é o tipo estudado neste trabalho.

Uma definição direta das regras de transição consiste na enumeração de todos os estados possíveis da príopria cíelula, das cíelulas vizinhas e do príoximo estado, contendo a informação necessíaria para computar o estado futuro das cíelulas em função de sua vizinhança e, consequentemente, da evolução como um todo. A regra do AC corresponde a uma função local, formada por transições de estado. A ação dela sobre cada cíelula de uma configuração pode ser vista como resultado da ação de uma função global sobre o

reticulado, que íe distinta da função local.

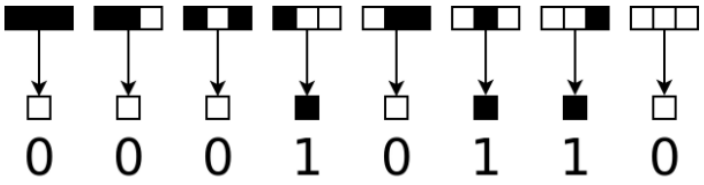


**Figura 1. Ilustração das regras de transição para a regra 22 no código Wolfram.**

Os ACEs são enumerados por suas regras de transição, de 0 a 255, o que é conhecido como número de Wolfram. Segundo [Wolfram 2002, p. 24], "A linha superior em cada caixa dá uma das possíveis combinações de cores para uma célula e sua vizinhança imediata. A linha inferior então especifica qual será a cor da célula central no próximo passo em cada um desses casos."

Uma definição direta de regra de transição consiste na enumeração de todas as combinações de estado possíveis da vizinhança de uma célula, e do próximo estado associado a elas, o que define a função local a ser usada na evolução temporal do reticulado.

### 2.3. Atualização Assíncrona por Prioridade da Vizinhança

Parte imprescindível da definição de um AC é a ordem de atualização das células no reticulado. Apesar de os ACs serem fundamentalmente concebidos com o comportamento de atualização síncrona, é possível utilizar outras estratégias.

No presente trabalho foi empregada uma nova estratégia de atualização: por "prioridade da vizinhança", em que os esquemas associados contém a prioridade da atualização de cada uma das 8 possíveis vizinhanças do espaço elementar, as quais deverão ser seguidas deterministicamente durante as evoluções temporais.

Esse tipo de atualização requer uma distinção das iterações globais e locais (microiterações): as prioridades de vizinhanças são iteradas sequencialmente da menor à maior e, para cada prioridade, o reticulado é varrido e atualizam-se apenas as células que tenham vizinhança com a prioridade atual, mantendo-se o estado das vizinhanças que não estejam na prioridade atual da iteração. Ao fim da varredura do reticulado, o processo se repete para a próxima prioridade, agora iterando sobre o reticulado gerado no processo descrito anteriormente, com uma prioridade menor. O reticulado resultante após iterar sobre todas as prioridades é chamado de iteração global (ou macroiteração) e é incluído nas renderizações, o que não acontece com as microiterações.

Esse modo de atualização inclui também a atualização síncrona, que pode ser simulada quando as prioridades do esquema têm mesmo valor (11111111, 22222222, ..., etc). As evoluções podem ser identificadas como um par $\langle R, E \rangle$, onde *R* é a regra e *E* é um esquema de prioridade.

É importante notar que diferentes esquemas de prioridade podem ser equivalentes: quando esquemas de prioridade equivalentes são aplicados a uma mesma regra, são geradas evoluções iguais. Um exemplo entre esquemas equivalentes pode ser entre os esquemas 11111112 e 11111113, em que as prioridades da última transição são nominalmente

diferentes, mas efetivamente iguais. Outro caso é a equivalência entre os oito esquemas síncronos: 11111111, 22222222, 33333333, etc. Para experimentos feitos neste trabalho, foram filtrados esses dois casos, de modo que não houvesse redundância nas execuções, reduzindo o custo computacional.

Além desses dois tipos de equivalência entre esquemas, é possível haver equivalência entre esquemas apenas quando são pareados a uma determinada regra, em função de suas transições não-ativas, como é o caso do que acontece no primeiro experimento, discutido na seção 4.

### 2.4. Conservabilidade Numérica

Segundo [Boccara and Fukś 2002], "Uma regra de AC $f$ de $q$ estados, unidimensional, de comprimento $n$ é conservativa se, para todas as configurações cíclicas de tamanho $L \geq n$ satisfaz":

$$f(x_1, x_2, ..., x_{n-1}, x_n) + f(x_2, x_3, ..., x_n, x_{n+1}) + ...$$
$$+ f(x_L, x_1, ..., x_{n-2}, x_{n-1} = x_1 + x_2 + ... + x_L$$

Ou seja, uma evolução é dita conservativa caso, para qualquer configuração inicial, a soma dos estados presentes em qualquer configuração é sempre a mesma para todas as iterações. Como é mencionado na Subseção 1.1, ACs conservativos podem ser usados para modelar ambientes em que há conservabilidade, como trânsito automobilístico, simulação de partículas, entre outros. A regra 184 é uma regra que, em execução síncrona, gera evoluções conservativas pela definição anterior de conservabilidade.

### 2.5. Classificação de Densidade

Segundo [Capcarrere et al. 1996, p. 1], "um exemplo dessa computação emergente é utilizar um AC para determinar a densidade global de bits em uma configuração de estado inicial. Conhecido como o problema de classificação de densidade, o AC unidimensional de dois estados é apresentado com uma configuração inicial arbitrária e deve convergir no tempo para um estado de apenas 1s se a configuração inicial contiver uma densidade de 1s > 0,5, e de apenas 0s se essa densidade < 0,5; para uma densidade inicial de 0,5, o comportamento do AC é indefinido". Em outras palavras, capacidade de classificação de densidade se dá quando o reticulado cíclico de um ACE torna-se totalmente preenchido pelo estado predominante na configuração inicial.

### 2.6. Problema da Paridade

Segundo [Betel et al. 2013, p. 1], "Consideramos o problema de paridade em autômatos celulares unidimensionais, binários e circulares: se a configuração inicial contém um número ímpar de 1s, a reticulado inteira deve convergir para 1s; caso contrário, deve convergir para 0s". Assim como o problema da maioria, "é fácil ver que o problema é mal-definido para reticulados de tamanho par (que, por definição, nunca seriam capazes de convergir para 1)". É possível entender, então, que a solução do problema da paridade se dá quando o reticulado cíclico de um ACE torna-se totalmente preenchido pelo estado de quantidade ímpar na configuração inicial.

## 3. Metodologia da Pesquisa

A pesquisa se limita a ACEs com o tipo de atualização evidenciado na Subseção 2.3, com comprimento de reticulados testados até $n$=15 e com $r$=1.

Para o primeiro experimento, foram testados esquemas de atualização *independentes* para o ACE 184, i.e., os esquemas cujas dinâmicas são definidas pelas transições *ativas* de estado da regra, as que mudam o estado da célula central.

Os três experimentos foram de natureza computacional. Foram testados comprimentos de reticulado ($n$) sucessivamente maiores, de $n$=5 em diante. Para o primeiro experimento, foram testados esquemas de atualização *independentes* para o ACE 184, i.e., os esquemas cujas dinâmicas são definidas pelas transições *ativas* de estado da regra, as que mudam o estado da célula central.

Para o segundo e o terceiro experimento, foram escolhidas regras *balanceadas* (as que têm quatro transições de estado para cada um dos estados, 0 e 1); o balanceamento se deve a uma condição necessária do problema e a restrição em 75 esquemas para diminuir as demandas computacionais em uma primeira fase. Em uma segunda fase, foram testadas as regras restantes, com exceção da regra 51, que contém a máxima quantidade possível de esquemas de atualização independentes, totalizando 545.835.

A capacidade de classificação de densidade e a capacidade de resolução do problema da paridade foram medidas através da contagem de configurações iniciais em que cada par $\langle R, E \rangle$ resolve o problema em questão.

## 4. Resultados dos Experimentos

### Experimento 1

É sabido que há cinco regras que apresentam conservabilidade numérica no caso síncrono: 170, 184, 204, 226 e 240. As regras 170 e 240 são dinamicamente equivalentes e produzem um comportamento de bit-shifting (para esquerda e direita, respectivamente), que é muito simples. A regra 204, por sua vez, é mais simples ainda, chamada de regra identidade, em que não há mudança no reticulado ao longo de sua evolução. A motivação para o primeiro experimento foi descobrir se, no esquema de atualização usado, essa propriedade é mantida para a regra 184 (que é equivalente à regra 226), que foi escolhida por ser a mais rica do ponto de vista dinâmico dentre as cinco, empatada com a 226.

Em relação à conservabilidade, a execução da regra 184 com todas as configurações iniciais de comprimento até $n$=15, e com todos os seus esquemas de prioridade independentes, resultou num conjunto de esquemas assíncronos em que a regra permanece conservativa, como é mostrado na Figura 2. Nota-se, entretanto, que todos correspondem ao modo síncrono, já que eles só diferem entre si nas prioridades das transições inativas, e as prioridades das ativas são as mesmas em cada esquema.

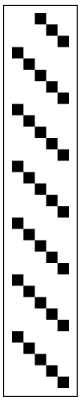

**Figura 2. Todos os esquemas assíncronos conservativos da regra 184 são equivalentes.**

| Regra | Esquema |
|---|---|
| 184 | 3222<u>2</u>2<u>2</u>1 |
| | 3<u>1</u>111<u>1</u>12 |
| | <u>2</u>333<u>3</u>3<u>1</u>1 |
| | <u>2</u>1111<u>1</u>13 |
| | <u>1</u>333<u>3</u>332 |
| | <u>1</u>222<u>2</u>223 |
| | <u>2</u>222<u>2</u>221 |
| | <u>2</u>1111<u>1</u>12 |
| | <u>2</u>1111<u>1</u>11 |
| | <u>1</u>222<u>2</u>222 |
| | <u>1</u>222<u>2</u>221 |
| | <u>1</u>1111<u>1</u>12 |
| | <u>1</u>1111<u>1</u>11 |

As prioridades com underscore significam que qualquer prioridade pode ser colocada em seu lugar. Isto se deve a, nesses índices, as transições de estado serem não-ativas e, portanto, serem irrelevantes em que ordem são atualizadas. No entanto, é possível ver que as prioridades das transições de estado ativas são iguais entre si, que são todas representações equivalentes da execução síncrona.

### Experimentos 2 e 3

A motivação para os dois últimos experimentos a respeito do problema da maioria e problema da paridade foi estudar se os ACEs com atualização assíncrona por prioridade eram capazes de resolver esses problemas de decisão emblemáticos no contexto de ACs.

A execução das regras e esquemas de prioridade independentes selecionadas até comprimento $n$=7 bastou para mostrar que nenhuma das regras elementares testadas foi capaz de classificar a densidade nem resolver o problema da paridade totalmente, qualquer

que fosse o esquema de atualização independente utilizado.

A seguir são apresentadas as regras a quantidade de esquemas testados para elas no presente trabalho.

**Tabela 1. Primeiro conjunto de regras e quantidade de esquemas testados**

| Regra | Qtd. Esquemas |
|---|---|
| 29 | 75 |
| 30 | 75 |
| 45 | 75 |
| 46 | 75 |
| 60 | 75 |
| 71 | 75 |
| 75 | 75 |
| 77 | 3 |
| 78 | 3 |
| 85 | 75 |
| 86 | 75 |
| 89 | 75 |
| 90 | 75 |
| 92 | 3 |
| 101 | 75 |
| 102 | 75 |
| 105 | 75 |
| 106 | 75 |
| 108 | 3 |
| 116 | 75 |
| 120 | 75 |
| 135 | 75 |
| 139 | 75 |
| 141 | 3 |
| 142 | 3 |
| 149 | 75 |

| Regra | Qtd. Esquemas |
|---|---|
| 150 | 75 |
| 153 | 75 |
| 154 | 75 |
| 156 | 3 |
| 165 | 75 |
| 166 | 75 |
| 169 | 75 |
| 170 | 75 |
| 172 | 3 |
| 180 | 75 |
| 184 | 75 |
| 195 | 75 |
| 197 | 3 |
| 198 | 3 |
| 201 | 3 |
| 202 | 3 |
| 204 | 1 |
| 209 | 75 |
| 210 | 75 |
| 212 | 3 |
| 216 | 3 |
| 225 | 75 |
| 226 | 75 |
| 228 | 3 |
| 232 | 3 |

**Tabela 2. Segundo conjunto de regras e quantidade de esquemas testados**

| Regra | Qtd. Esquemas |
|---|---|
| 23 | 4683 |
| 27 | 4683 |
| 39 | 4683 |
| 43 | 4683 |
| 53 | 4683 |
| 54 | 4683 |
| 57 | 4683 |
| 58 | 4683 |
| 83 | 4683 |
| 99 | 4683 |
| 113 | 4683 |
| 114 | 4683 |
| 147 | 4683 |
| 163 | 4683 |
| 177 | 4683 |
| 178 | 4683 |

## 5. Considerações Finais

Estudamos três importantes propriedades nos ACEs executados com a estratégia de atualização assíncrona por prioridade da vizinhança: conservabilidade numérica, capacidade de classificação de densidade e capacidade de resolver o problema da maioria. Nosso objetivo foi dar um passo no estudo sobre ACEs com o esquema de atualização assíncrono utilizado.

O primeiro experimento mostrou que a regra 184 permanece conservativa apenas em esquemas de atualização equivalentes ao modo síncrono. Já o segundo experimento revelou que, para os esquemas de atualização testados, não havia pares $\langle R, E \rangle$ que resolvessem o problema da maioria. Por fim, o terceiro experimento demonstrou que não havia pares $\langle R, E \rangle$ que resolvessem o problema da paridade. O código-fonte desses experimentos está disponível em: `github.com/mvrozanti/AsyncElementaryCAUdsN-python-experiments`.

A única regra que não foi analisada no contexto dos problemas de decisão discutidos foi a regra 51 (chamada regra inversora, que muda 0s para 1s e vice-versa), e que contém a máxima quantidade possível de esquemas de atualização independentes (545.835 esquemas), em função do alto custo computacional. É importante notar que esta estratégia de atualização é recente e ainda há muitos testes a serem feitos, seja para identificar propriedades estáticas ou verificar a capacidade de resolver problemas de decisão.

## Referências


Abdalla, M. and Gurdal, Z. (2002). Structural design using optimality based cellular automata. In *43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1676. DOI: 10.2514/6.2002-1676.

Betel, H., de Oliveira, P. P., and Flocchini, P. (2013). On the parity problem in one-dimensional cellular automata. *Natural Computing,* 12(3):323-337.

Boccara, N. and Fukś, H. (2002). Number-conserving cellular automaton rules. *Fundamenta Informaticae*, 52(1-3):1–13.

Capcarrere, M. S., Sipper, M., and Tomassini, M. (1996). Two-state, r= 1 cellular automaton that classifies density. *Physical review letters*, 77(24):4969.

Gardner, M. (1970). Mathematical games: The fantastic combinations of john conway's new solitaire game "life". *Scientific American*, 223(4):120–123.

Lafe, O. (1997). Data compression and encryption using cellular automata transforms. *Engineering Applications of Artificial Intelligence*, 10(6):581–591.

Mitchell, M., Crutchfield, J. P., and Hraber, P. T. (1994). Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D: Nonlinear Phenomena*, 75(1-3):361–391.

Rosenblueth, D. A. and Gershenson, C. (2011). A model of city traffic based on elementary cellular automata. *Complex Systems*, 19(4):305.

Salem, J. B., Wolfram, S., et al. (1985). Thermodynamics and hydrodynamics with cellular automata. *Thinking Machines Corporation technical report*, 24.

Wolfram, S. (1983). Cellular automata. *Los Alamos Science*, 9.

Wolfram, S. (2002). *A new kind of science*. Wolfram media Champaign, IL.