DINCON 2017
CONFERÊNCIA BRASILEIRA DE DINÂMICA, CONTROLE E APLICAÇÕES
30 de outubro a 01 de novembro de 2017 – São José do Rio Preto/SP

Representação de autômatos celulares unidimensionais balanceados, por meio de *templates*

William Barbosa dos Santos[1] and Pedro Paulo Balbi de Oliveira[2]

[1, 2]Pós-Graduação em Engenharia Elétrica e Computação, Universidade Presbiteriana Mackenzie, São Paulo, SP
[2]Faculdade de Computação e Informática, Universidade Presbiteriana Mackenzie, São Paulo, SP

Resumo. Discute-se a criação de uma representação para regras balanceadas de autômatos celulares unidimensionais, utilizando-se a noção de *templates* de regras, que é uma forma de se representar um conjunto de regras de determinada característica. Uma regra é balanceada quando sua representação k-ária apresenta todos os estados de um espaço de regras em quantidades iguais; tal propriedade é importante, por exemplo, no estudo de regras reversíveis. Implementou-se um algoritmo capaz de gerar todas as regras que o *template* representa, com posterior análise das mesmas quanto à presença da propriedade de balanceamento. Apesar de o método desenvolvido ser custoso computacionalmente, sua utilização em conjunto com outras propriedades de autômatos celulares faz como que a solução aqui proposta possa ser de extrema valia como ferramenta de redução do espaço de regras, em casos onde o balanceamento seja desejável.

Palavras-chave. Autômato celular unidimensional, balanceamento, regras balanceadas, templates, espaço de regras, reversibilidade, classificação de densidade.

1 Introdução

Discutem-se aqui regras balanceadas de autômatos celulares (ACs) unidimensionais, objetivando-se a criação de um algoritmo para geração automática de ACs de regras balanceadas em um determinado espaço.

Regra balanceada é aquela em que sua representação k-ária possui a mesma quantidade de cada um de seus k estados. Regras com essa propriedade são importantes pois auxiliam na redução do espaço de regras que implementem determinada tarefa computacional desejada; este é caso, por exemplo, do problema de classificação de densidade [1] – em que se deseja uma regra que determine o estado mais frequente na condição inicial – pois, mesmo não existindo uma regra que resolva o problema perfeitamente, o uso de balanceamento pode ser um dos caminhos a ser seguidos na busca de uma solução eficiente.

[1]wbs.developer@gmail.com
[2]pedrob@mackenzie.br

Da mesma forma, [3] mostra que todas as regras reversíveis são também balanceadas, de modo que uma generalização da representação das regras balanceadas pode auxiliar no estudo de regras com essa característica.

A investigação foi realizado computacionalmente, utilizando-se o software Mathematica [5] e a biblioteca CATemplates [6].

A próxima seção fornece mais detalhes sobre os autômatos celulares, bem como sobre a noção de regras balanceadas. Em seguida, a Seção 3 descreve a representação matemática das regras balanceadas implementada na biblioteca CATemplates. Na Seção 4 discutem-se os resultados obtidos.

2 Conceitos Básicos

2.1 Autômatos Celulares

Autômatos celulares são sistemas que podem representar comportamento global arbitrariamente complexo, mesmo a partir de operações locais extremamente simples [4]. Tratam-se de sistemas totalmente discretos (no tempo, espaço e nas varáveis de estado), em que cada unidade básica de processamento (a célula) age apenas localmente, com suas vizinhas mas, em conjunto, são capazes de gerar padrões emergentes complexos, mesmo em famílias de regras muito simples [4].

ACs são comumente representados como uma matriz, N-dimensional ($N > 0$), sendo que cada ponto (célula) da matriz assume um estado discreto no tempo t, representado por um de k valores inteiros possíveis, entre 0 e $k - 1$. A partir de uma matriz inicial, os estados das células em $t + 1$ são obtidos por meio de sua $r$egra, que age a partir das próprias células e suas vizinhas em t, isto é, por meio de um processamento totalmente local da regra.

O espaço de um autômato celular é definido pelos valores de k e r, este último o $r$aio da vizinhança, que indica sua abrangência espacial. O espaço elementar é definido por $k = 2$ e $r = 1$, significando que uma célula que esteja nesse espaço sempre estará em um de dois estados possíveis (0 ou 1), e sua vizinhança incluirá as células diretamente à esquerda e à direita (compondo 3 células na vizinhança).

Há ainda a possibilidade de se definir que apenas uma das células, à esquerda ou à direita, fará parte da vizinhança. Nesse caso diz-se que o AC possui $r = 0.5$, e sua vizinhança passa a ser composta por 2 células.

A transição de uma célula obedece à regra de transição desse autômato celular, que pode ser expressa por uma operação matemática, ou simplesmente, por uma tabela de transições de estado [4]. O número da regra é definido segundo a ordem lexicográfica das vizinhanças (com a homogênea em 1 à esquerda). Dessa forma, pode-se resumir a tabela de transição pela sua forma k-ária, no caso, 00011110, que corresponde ao decimal 30.

Como em qualquer espaço (ou, família) de regras existem $k^{k^{2r+1}}$ regras, o custo de exploração do espaço cresce exponencialmente, ao se tentar identificar regras específicas, ou que possuam determinada propriedade. O presente artigo vem, portanto, para facilitar a identificação das regras balanceadas.

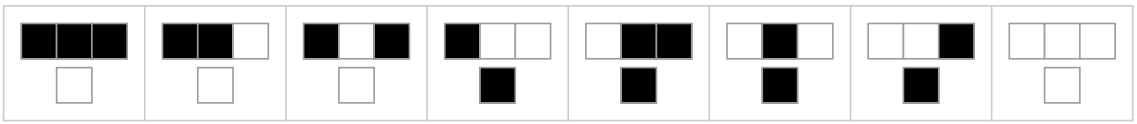

Figura 1: Tabela de transição da regra 30 do espaço elementar.

2.2 Templates para Representação de Regras para Autômatos Celulares

Por definição, uma tabela de transição representa uma única regra em determinado espaço. A noção de *templates* vem como uma forma de representar um conjunto de regras [2]. Assuma por exemplo a regra 00011110, na representação por *templates* poderíamos utiliziar variáveis livres ao invés do valor exato, por exemplo $000111x_10$. Sendo x_1 uma variável, pode conter qualquer em k. Assim sendo, o *template* $000111x_10$ representa, para o espaço elementar, as regras 00011100 e 00011110.

Com um *template* definido pode-se realizar determinadas operações. Entre elas a intersecção, que é o processo de gerar um novo *template* à partir de dois *templates* existentes, onde o novo representa somente as regras existentes em ambos os *templates* interseccionados. E a expansão, que é a geração de todas as regras que um *template* representa.

Em sua definição original, *templates* podem possuir propriedades adicionais, como a definição do espaço em si em que atuam e funções a serem executadas após a operação de expansão [2]. Ressalte-se, entretanto, que a notação aqui utilizada para apoiar a generalização das regras balanceadas é uma versão simplificada da representação original de *templates*.

3 Regras Balanceadas

Autômatos celulares balanceados são aqueles que possuem em sua matriz de transição quantidades iguais de todos os valores possíveis em k. Para um AC binário isso significa número igual de 1s e 0s. Devido a essa característica pode-se calcular a quantidade de regras balanceadas dado um determinado k e r calculando-se as permutações possíveis na tabela de transição. Em especial para o espaço elementar tem-se $8!/(4! \times 4!) = 70$ regras [3]. A função f a seguir generaliza o cômputo, para um espaço qualquer:

$$g(k, r) = k^{\lceil 2r \rceil + 1} \tag{1}$$

$$f(k, r) = \frac{\mathrm{g}(k, r)!}{\left(\frac{\mathrm{g}(k,r)}{k}!\right)^k} \tag{2}$$

A Figura 1 exibe um exemplo de regra balanceada. Observa-se que a tabela de transição possui quantidade igual de 0s e 1s. Para uma representação genérica da tabela de transição, utilizaremos uma simplificação da representação por *templates* [2]. No caso, teremos todas as posições da tabela de transição preenchidas com variáveis livres, exceto pela última, onde será utilizada uma função matemática para definição do balanceamento. Tal definição

tem por objetivo representar um conjunto de regras onde a tabela de transição respeite os resultados da função de balanceamento. Quanto às variáveis livres, essas podem assumir qualquer valor dentro do espaço restrito do AC.

Dessa forma, um *template* que represente as regras balanceadas deve possuir todas as suas posições com variáveis livres exceto uma. Isto porque, não fosse assim e tivéssemos dois campos calculados, o *template* não seria capaz de representar a regra balanceada oriunda da permuta de seus campos calculados.

Uma maneira de se calcular o último campo do *template* para que representem regras balanceadas seria assumir o resultado da soma das demais posições do *template* e ponderar o resultado de modo a se chegar ao valor faltante. Para um *template* com $r = 0.5$ e $k = 2$, regras balanceadas precisam ter duas posições com valor 0 e duas posições com valor 1. A soma dos valores então resulta em $\sum_{i=0}^{N-1} x_i = 2$, sendo N como definido pela Expressão (3).

Assumindo a última posição do *template* como a posição a ser calculada, a expressão para o cálculo pode ser vista em (4).

$$N = k^{\lceil r*2 \rceil + 1} \tag{3}$$

$$b(k,r) = -((\sum_{i=1}^{N-1} x_i) - ((\sum_{i=0}^{k-1} i)\frac{N}{k})) \tag{4}$$

$$x_0 = -\left(\sum_{i=1}^{N-1} x_i - \left(\frac{(k-1)k}{2}\frac{N}{k}\right)\right) \tag{5}$$

A Expressão (5) mostra a Equação (4) pouco mais resumida. Assim, tendo-se um *template* de características quaisquer, basta a aplicação desse cálculo na última posição para que o mesmo represente as regras balanceadas. No entanto, é necessário retirar as regras inválidas geradas, isto é, regras que possuem campos com valores não existentes em k. Isto porque a Expressão (5) força o balanceamento do *template* com base na soma que deveria ser o resultado de todos os campos da tabela de transição, uma vez que todas as regras balanceadas de um espaço possuem o mesmo valor de soma.

Essa característica do cálculo também ocasiona a precisão apenas para autômatos celulares binários. Portanto a Expresão apresentada não funciona para ACs com $k > 2$.

O *template* (8) demonstra o uso da Expressão (5) para representar regras balanceadas no espaço $k = 2, r = 0.5$, e o *template* (11) para o espaço de $k = 2$ e $r = 1$.

$$T_0 = (x_3, x_2, x_1, -((x_3 + x_2 + x_1) - [\tfrac{2(2-1)}{2}\tfrac{4}{2}])) \tag{6}$$

$$T_0 = (x_3, x_2, x_1, -((x_3 + x_2 + x_1) - 2)) \tag{7}$$

$$T_0 = (x_3, x_2, x_1, 2 - x_3 - x_2 - x_1) \tag{8}$$

$$T_1 = (x_7, x_6, x_5, x_4, x_3, x_2, x_1, - ((x_7 + x_6 + x_5 + x_4 + x_3 + x_2 + x_1) - [\tfrac{(2-1)*2}{2}\tfrac{8}{2}])) \tag{9}$$

$$T_1 = (x_7, x_6, x_5, x_4, x_3, x_2, x_1, \\ -((x_7 + x_6 + x_5 + x_4 + x_3 + x_2 + x_1) - 4)) \quad (10)$$

$$T_1 = (x_7, x_6, x_5, x_4, x_3, x_2, x_1, \\ (4 - x_7 - x_6 - x_5 - x_4 - x_3 - x_2 - x_1)) \quad (11)$$

A Tabela 1 ilustra o resultado dos cálculos em (8) para todos os valores possíveis em x_0, x_1 e x_3. Os resultados onde $x_0 = 2$ e $x_0 = -1$ são descartados, e com isso temos exatamente as regras que possuem números iguais de 1s e 0s. O processo é o mesmo para qualquer raio, alterando-se apenas o tamanho da tabela verdade, dado que a quantidade de variáveis livres (x_i) aumenta exponecialmente (como pode ser visto pela função g em (2)).

Tabela 1: Tabela verdade para T_0 (Equação (8)).

x_3	x_2	x_1	x_0
0	0	0	2
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	-1

O problema na lógica do cálculo apresentado está em considerar os valores em cada posição do *template*. A propriedade de balancemaneto não é dependente dos valores que estão sendo representados. Uma regra pode ser dita balanceada tão logo possua quantidades iguais de todos os estados possíveis no espaço que representa. Na prática isso quer dizer que pode-se ter uma AC específico cujo estados possíveis sequer estejam no domínio numérico e, portanto, a utilização da somatória dos estados passa a não ser aplicável. Ainda que possamos atribuir um valor numérico aos estados do AC por conveniência, para regras com mais de dois estados possíveis a Expressão (5) perde sua acurácia. Isso porque, diferentemente das regras com $k = 2$, pode haver diferentes configurações onde a soma dos elementos equivale à soma dos elementos das regras balanceadas. Para $k = 4, r = 0.5$, uma regra balanceada teria: 0123, a soma dos elementos nesse é $\sum_{i=0}^{3} x_i = 6$. No entanto a regra 0033 também tem a soma de seus elementos resultando em 6, e pela Expressão (5) ela seria considerada balanceada.

Observando o problema pode-se notar que os valores dos estados se cruzam, sendo que apenas 2 posições com o valor máximo já é o suficiente para balancear o *template* inteiro. O erro nesse caso está em balancear o *template* com base nos valores de suas células. Ao invés disso, podemos criar faixas de valores e atribuir pesos a cada valor possível do AC. O importante para se evitar o mesmo problema da Expressão (5) é que as faixas de valores não se cruzem. A título de exemplo, para $k = 2$ e $r = 0.5$, basta atribuir para as posições

onde $x = 0$ o peso de 0.1, e para as posições onde $x = 1$ o peso de 0.5. Com as faixas de valores definidas dessa maneira, a soma dos valores de qualquer regra dita balanceada será de 1.2. Utilizando essa lógica evitamos o problema de cruzamento dos valores, pois mesmo que todas as posições da regra sejam 0, a soma dos pesos será menor do que um único elemento 1 na regra. E, caso tenhamos 3 posições com valor 1, a soma do total dos pesos já seria 1.5, tornando a regra inválida. De forma análoga ao cálculo para balanceamento utilizado na Expressão (5), tem-se que eliminar os resultados inválidos da expansão, exceto que desta vez os resultados inválidos contemplam também números reais, dado que pesos reais são utilizados. A Expressão (15) ilustra a lógica aqui descrita, utilizando (13) para calcular o peso dos estados.

$$N = k^{\lceil r*2 \rceil + 1} \tag{12}$$

$$p(k, r, x) = (x + 1)^{N/k} \tag{13}$$

$$s(k, r, x_1 ... x_{N-1}) = (\sum_{i=0}^{N-1} p(k, r, i \bmod k)) - (\sum_{i=1}^{N-1} p(k, r, x_i)) \tag{14}$$

$$x_0 = (s(k, r, x_1 ... x_{N-1})^{1/(N/k)}) - 1 \tag{15}$$

O *template* (16) ($k = 4, r = 0.5$) ilustra o uso da Expressão (15), e a expansão pode ser observada na Tabela 2.

$$T_0 = (x_3, x_2, x_1, \\ ((p(2, 0.5, 1) * 2 + p(2, 0.5, 0) * 2) - (p(2, 0.5, x_3) + p(2, 0.5, x_2) + p(2, 0.5, x_1)))^{0.5} - 1) \tag{16}$$

Tabela 2: Tabela verdade para T_0 (Equação (16)).

x_3	x_2	x_1	x_0
0	0	0	1.64575
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	-1+1.41421i

4 Considerações Finais

O método aqui descrito foi testado para os espaços $\{k = 2, r = 1.0\}$, $\{k = 3, r = 0.5\}$ e $\{k = 2, r = 1.5\}$. A quantidade de regras geradas nesses espaços foi comparada com o resultado da equação que define o total de regras balanceadas em um espaço. Além disso as regras geradas foram verificadas computacionalmente para garantia de que são de fato balanceadas.

A simples permuta de valores pode ser considerado um meio alternativo de se gerar todas as regras balanceadas em um espaço. O método utilizado neste trabalho possui custo computacional relativamente elevado, dado que na verdade todas as regras do espaço necessitam ser calculadas para que então se descarte as que não se adequem. Isso no entanto gera um benefício colateral, pois permite que regras não completamente balanceadas, mas com algum grau (a ser definido) de balanceamento permaneçam no conjunto de regras a ser analisado.

Em adição, ainda é possívei utilizar a generalização aqui proposta para se realizar a intersecção com outros *templates*. Ao se realizar essa operação une-se a propriedade de balanceamento com uma nova definição utilizada no *template* a ser interseccionado. Isso permite uma redução do espaço de regras por torná-lo mais específico, o que é muito importante ao se analisar o espaço.

Faz-se necessário estudar mais aprofundadamente as características das regras não completamente balanceadas, bem como definir um grau de balanceamento, o que deverá ser considerado em estudos futuros.

5 Agradecimentos

Pedro P.B. de Oliveira agradece o apoio do MackPesquisa (Fundo Mackenzie de Pesquisa) e do CNPq.

Referências

[1] P.P.B. de Oliveira, On density determination with cellular automata: results, constructions and directions, Journal of Cellular Automata, vol. 9, 357–385, (2014).

[2] M. Verardo and P.P.B. de Oliveira, Representing Families of Cellular Automata Rules, The Mathematica Journal, vol.16, (2014)

[3] G. Kronemberger e P.P.B. de Oliveira, A hipótese das regras primitivas e derivadas, na busca construtiva por autômatos celulares reversíveis, Simpósio Brasileiro de Automação Inteligente, vol. X, (2011)

[4] S. Wolfram, A new kind of science, Wolfram Media Inc, vol. 1, (2002)

[5] Wolfram Research, Wolfram Mathematica, http://www.wolfram.com/mathematica/

[6] M. Verardo e P.P.B. de Oliveira, CATemplates, https://github.com/mverardo/CATemplates