Aperfeiçoando a Representação de Autômatos Celulares Através de *Templates*

Zorandir Soares Jr.
zorandir@gmail.com

Universidade Presbiteriana Mackenzie
Programa de Pós-Graduação em Engenharia Elétrica e Computação

Orientador: Prof. Dr. Pedro Paulo Balbi de Oliveira

11 de agosto de 2015

Sumário

Objetivos e Motivação

Autômatos Celulares

Templates

Resultados Parciais

Problemas e Trabalhos Futuros

Cronograma

Objetivos e Motivação

- Aperfeiçoar a representação de autômatos celulares através de *templates*
- Desenvolver e melhorar algoritmos geradores de templates
- Apresentar exemplos da utilização dos templates em problemas típicos dos ACs
- Pesquisar e implementar soluções para os principais problemas em aberto

Autômatos Celulares

Figura: Tapetes expostos em uma exposição de arte realizada na "Maison Salvan", em Carjac, França, em junho de 2008. Eles foram criados com o simulador FiatLux CA.

Famílias de autômatos celulares

Uma família (ou espaço) de autômatos celulares é definida pelo raio r e pelo número de estados k.

O tamanho de uma família é definido pela expressão abaixo:

$$k^{k^{2r+1}} \tag{1}$$

Propriedades Estáticas

- Confinamento
- Simetria Interna Máxima
- Simetria Interna Arbitrária
- Totalidade e Semi-totalidade
- Conservabilidade da soma de estados
- Conservabilidade da soma modular de estados

Templates

Template é uma generalização de tabelas de transições de ACs.

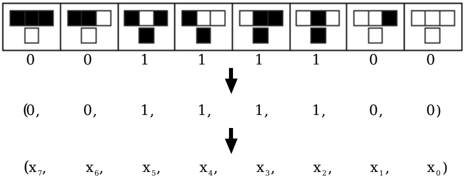


Figura: Exemplo de tabelas de transições

Exemplo - Template Base

$$\begin{aligned} &(k:2, r:1, \\ rawList:\ &(x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0), \\ expansionFunction:\ &RawExpansion) \end{aligned} \tag{2}$$

Exemplo - Template Modular

$$\begin{aligned}(k:2, r:1, N:2,\\ rawList:(x_7, x_6, x_5, x_4, x_3, x_2, x_1, x_0),\\ imprisonmentExpressions:(x_0 \in \{0,1\}),\\ expansionFunction: ModNExpansion)\end{aligned} \tag{3}$$

Expansão

Expansão é o processo no qual se obtêm todas as tabelas de transição R_k associadas a um template T.

$$E(T) = R_k \qquad (4)$$

Expansão

Expansão é o processo no qual se obtêm todas as tabelas de transição R_k associadas a um template T.

$$E(T) = R_k \tag{4}$$

- RawExpansion
- FilteredExpansion
- ModNExpansion

Intersecção

$$I(T_1, T_2) = T_3 \Leftrightarrow E(T_3) = E(T_1) \cap E(T_2) \quad (5)$$

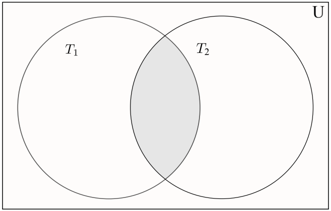


Figura: Os círculos T_1 e T_2 são templates que representam dois conjuntos de regras. Em cinza, T_3 é o template que representa o conjunto de regras de intersecção entre T_1 e T_2.

Intersecção

Intersecções resolvidas

- Intersecção de templates não modulares
- Intersecção de templates modulares com mesmo valor de N

Complemento

$$C(T_1) = \bar{T}_1 \Leftrightarrow E(\bar{T}_1) = U \setminus E(T_1) \quad (6)$$

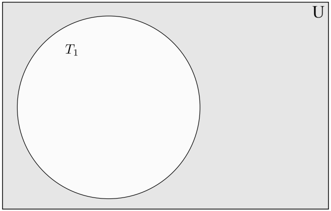


Figura: Em branco, T_1 é um template que representa um conjunto de regras. Em cinza, $\bar{T}_1$ é um conjunto de template que representa o complemento de T_1.

Geradoras de Templates e Número de Estados

Tabela: Compatibilidade entre algoritmos geradores de templates e número de estados

	Número de Estados	
Algoritmos Geradores de Templates	$k=2$	$k>2$
Totalidade e Semi-Totalidade	•	•
Confinamento	•	•
Color Blind	•	•[1]
Simetria Máxima	•	•
Simetria Arbitrária	•	
Conservabilidade da soma de estados	•	•
Conservabilidade da soma modular de estados	•[1]	

[1] Template Modular

Resultados Parciais

- Aplicação no Problema de Paridade
- Operação de Complemento

Problema de Paridade

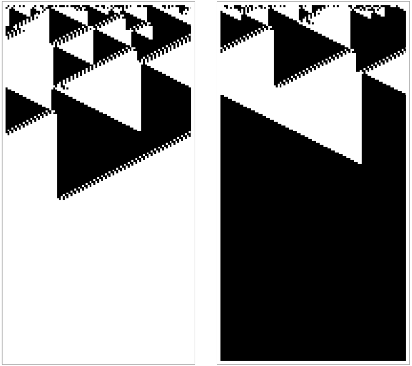


Figura: Exemplo de regra de paridade. A imagem a esquerda contém em sua entrada um número par de 1s. A da direita contém um número ímpar.

Problema de Paridade

$$T_{paridade} = (T_{conservaparidade} \cap T_{confinado}) \cap \bar{T}_{conservaestados} \quad (7)$$

A regra que solucione o problema de paridade:

- deve ser **confinada**
- deve **conservar a paridade**
- não deve **number conserving**

Problema de Paridade

Tamanho do espaço:

$$2^{2^{2\times 3+1}} = 3,4 \times 10^{38} \tag{8}$$

Número máximo de regras conservativas de paridade:

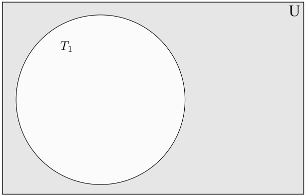
$$2^{63} \approx 9.2 \times 10^{18} \tag{9}$$

Complemento

$$rawList : (x_7, x_6, x_5, 1{-}x_1, x_3, x_2, x_1, 0) \qquad (10)$$

$$\begin{cases} x_7 & = & x_7 \\ x_6 & = & x_6 \\ x_5 & = & x_5 \\ x_4 & = & 1 - x_1 \\ x_3 & = & x_3 \\ x_2 & = & x_2 \\ x_1 & = & x_1 \\ x_0 & = & 0 \end{cases} \qquad (11)$$

$$x_4 = 1 - x_1 \wedge x_0 = 0 \qquad (12)$$

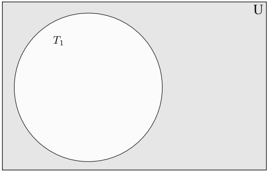


Complemento

$$x_4 = 1 - x_1 \vee x_0 = 0 \quad (13)$$

$$x_4 = 1 - (1 - x_1) \vee x_0 = 1 - 0 \quad (14)$$

$$S = \{\{x_4 \to x_1\}, \{x_0 \to 1\}\} \quad (15)$$



$$\{(x_7, x_6, x_5, x_4, x_3, x_2, x_1, 1), (x_7, x_6, x_5, x_1, x_3, x_2, x_1, x_0)\} \quad (16)$$

Intersecção de Templates Modulares

Intersecções não resolvidas

- Intersecção de templates com Mods diferentes
- Intersecção de um template modular com um template não modular

Solução do problema permite:

- Realizar o complemento de Templates Modulares
- Criar a operação de diferença entre Templates

Intersecção de Templates Modulares primos entre si

Teorema Chinês dos restos

Sejam $m_1, m_2, \ldots, m_r$, r inteiros positivos que são primos entre si, dois a dois, e sejam $a_1, a_2, \ldots, a_r$, r inteiros quaisquer. Então, o sistema de congruências:

$$\begin{cases} x \equiv a_1 \pmod{m_1} \\ x \equiv a_2 \pmod{m_2} \\ \quad\vdots \\ x \equiv a_r \pmod{m_r} \end{cases}$$

admite um solução x única módulo $m = m_1 m_2 \ldots m_r$

Negação de variável

No caso binário, uma variável pode ser negada por meio da função $f(x) = 1 - x$. No template abaixo a terceira posição ilustra a variável x_1 negada.

$$\begin{aligned} &(k = 2, r = 0.5, \\ rawList &= (1, 1 - x_1, x_1, 0), \\ expansionFunction &= RawExpansion) \end{aligned} \tag{17}$$

Negação de variável

A questão: como negar variáveis em templates não binários?

Solução do problema permite:

- Abstração do algoritmo gerador de templates de simetria interna.
- Abstração da operação de complemento.

Operação de Diferença entre Conjuntos

$$C(T_1, T_2) = T_3 \Leftrightarrow E(T_3) = E(T_2) \setminus E(T_1) \tag{18}$$

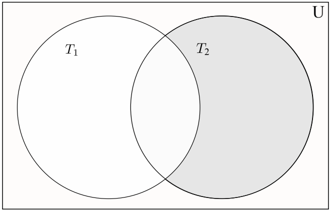


Figura: Os círculos T_1 e T_2 são os templates que representam dois conjuntos de regras. Em cinza, T_3 é um conjunto de template que representa o conjunto de regras de $T_2 - T_1$.

Cronograma

- Fase 0: participação nas disciplinas necessárias ao cumprimento dos créditos para a obtenção do título de mestre;
- Fase 1: pesquisa bibliográfica de autômatos celulares e de templates;
- Fase 2: programação e testes da operação de complemento de templates;
- Fase 3: proposição de propriedades estáticas para se gerar templates;
- Fase 4: pesquisa e implementação dos problemas em abertos e de novas operações de geração de templates;
- Fase 5: submissão de artigo;
- Fase 6: escrita da dissertação.

Cronograma

Tabela: Cronograma de desenvolvimento do projeto

	2014				2015			
Atividades	Jan-Mar	Abr-Jun	Jul-Set	Out-Dez	Jan-Mar	Abr-Jun	Jul-Set	Out-Dez
Fase 0	•	•	•	•		•		
Fase 1				•	•			
Fase 2					•	•		
Fase 3							•	
Fase 4							•	•
Fase 5							•	
Fase 6						•	•	•

Agradecimentos

À Capes, CNPq, MackPesquisa e ao Laboratório de Computação Natural (LCoN).