Implementação de uma Solução Adaptativa para o Problema do Emparelhamento de Cadeias

(05 Janeiro 2009)

S. M. Melo, I. A. M. Rodrigues, A. A. de Castro Jr.

***Resumo*— O problema do emparelhamento de cadeias consiste em encontrar ocorrências de uma cadeia de caracteres (conhecida como padrão) dentro de outra cadeia de tamanho maior ou no corpo de um texto. Existe uma grande variedade de soluções propostas para este problema, uma delas baseada em autômatos finitos. Rodrigues *et al*[1] descreveu uma solução que utiliza os autômatos adaptativos para criar um conjunto de submáquinas que constroem o autômato finito que reconhece o padrão. Entretanto, a solução proposta não foi implementada completamente. Neste trabalho, pretende-se complementar os resultados da proposta adaptativa apresentada por Rodrigues, através da implementação completa do modelo proposto, da realização de um estudo sobre a complexidade de tempo e espaço do modelo apresentado e da comparação com outras técnicas existentes. Vale ressaltar que este trabalho apresenta os resultados parciais de um projeto de conclusão de curso com término previsto para junho de 2009. O trabalho está sendo realizado sob a orientação do Prof. Amaury Antônio de Castro Junior, do Curso de Sistemas de Informação do campus de Coxim (CPCX) da Universidade Federal de Mato Grosso do Sul (UFMS).**

Palavras-chave*— *autômatos adaptativos, emparelhamento de cadeias, Adaptools, complexidade de algoritmos.

I. INTRODUÇÃO

O problema do emparelhamento de cadeias consiste em encontrar ocorrências de uma cadeia de caracteres dentro de outra cadeia de tamanho maior ou no corpo de um texto. Este problema é também conhecido como *string-maching* ou *pattern-matching* [2,3].

O crescimento do uso de alinhamento de seqüências na busca de soluções para o problema de seqüenciamento genético, dentre outros tantos tipos de aplicações ligadas ou não à biologia molecular, torna o problema do emparelhamento de cadeias alvo de muitos trabalhos de pesquisa em Ciência da Computação.

Há, na literatura, uma grande variedade de algoritmos para o problema do emparelhamento de cadeias, entre eles destacam-se o de Força Bruta, o de KMP (Knuth, Morris e Pratt) e o de Boyer-Moore, considerados importantes e interessantes de serem estudados [3]. Uma outra solução bastante eficiente é baseada na utilização de autômatos finitos [2,5]. Esta última constrói um autômato finito que é responsável pela busca, na cadeia de texto, de todas as ocorrências de um determinado padrão. Para cada padrão, existe um autômato de emparelhamento de cadeias correspondente. Esse autômato é obtido a partir do padrão em uma etapa de pré-processamento, antes de ser utilizado para a busca do respectivo padrão na cadeia de texto. Portanto, para cada padrão a ser buscado no texto é necessária a construção de um novo autômato.

Rodrigues *et al* [1,6] apresentaram uma alternativa utilizando os autômatos adaptativos para o mesmo problema. Os autômatos adaptativos são dispositivos que possuem algumas transições especiais que quando acionadas, executam funções adaptativas que modificam a topologia do próprio autômato. Estas funções adaptativas acrescentam ou eliminam transições e estados, permitindo que a estrutura e o comportamento do autômato sejam dinamicamente modificados.

Através dos autômatos adaptativos é possível automatizar a realização do pré-processamento, possibilitando a adequação dos parâmetros de entrada e a eliminação de comparações desnecessárias, provendo uma solução alternativa baseada no modelo adaptativo. Entretanto, faz-se necessário um estudo sobre a complexidade de tempo e espaço desta proposta, bem como a sua implementação e a comparação com outras técnicas existentes.

II. AUTÔMATO DE EMPARELHAMENTO DE CADEIA

A utilização de autômatos de emparelhamento de cadeias exige a construção de um autômato para cada padrão *P* existente a partir de uma etapa de pré-processamento. Após sua construção o autômato é usado para pesquisar a ocorrência do padrão no corpo do texto.

As etapas para busca de um padrão no texto são representadas pela Figura 1. A busca é realizada em duas etapas: a primeira é a construção do autômato (pré-processamento); a segunda é o uso deste autômato para realizar a busca e encontrar todas as ocorrências do padrão no texto.

S. M. Melo e I. A. M. Rodrigues são alunas do curso de Sistemas de Informação da Universidade Federal de Mato Grosso do Sul (UFMS) no Campus de Coxim (CPCX), Av. Márcio de Lima Nantes, S/N – Vila da Barra – CEP: 79400-000 - Coxim/MS, Brasil (endereço eletrônico: silmorita@gmail.com; iranirv@gmail.com).

A. A. Castro Jr. é docente do Departamento de Ciências Exatas (DEX) do Campus de Coxim (CPCX) da Universidade Federal de Mato Grosso do Sul (UFMS), Av. Márcio de Lima Nantes, S/N – Vila da Barra – CEP: 79400-000 - Coxim/MS, Brasil (endereço eletrônico: amaury.ufms@gmail.com)

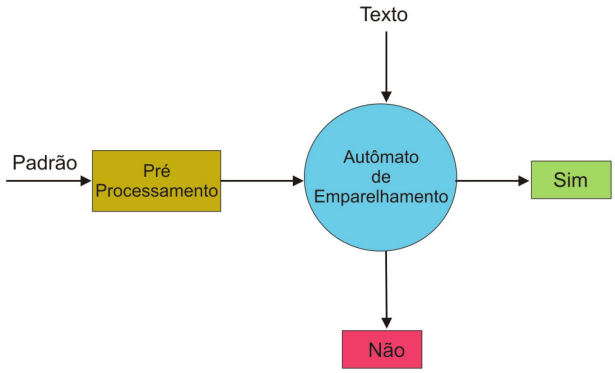

Fig. 1. Seqüência de etapas para criação e utilização de autômatos para o problema de emparelhamento de cadeias.

Inicialmente, é necessário definir uma função auxiliar σ, denominada função sufixo que mapeia de Σ* em {0,1,..*m*}, onde $\sigma(x)$ é o tamanho do maior prefixo de *P* que é o sufixo de *x*.

Por exemplo, se $p = ab$, $\sigma(e) = 0$, $\sigma(ccaca) = 1$, $\sigma(ccab) = 2$, o autômato de emparelhamento de cadeias que corresponde a um padrão $P[1..m]$ é definido da seguinte forma:

- Estado inicial q_0 é o estado 0,
- *P* é uma cadeia de padrão fixa,
- $Q = \{0,1,...,m\}$ é um conjunto finito de estados,
- *m* é o único estado aceitável,
- Σ é um alfabeto de entrada finito.
- A função de transição δ é descrita pela seguinte equação, para qualquer estado *q* e caractere *a*:

$$\delta(q, a) = \sigma(P_q a)$$

- O novo estado $\delta(q, a)$, corresponde ao prefixo de P com maior comprimento que é também sufixo de $P_q a$.

Para ilustrar a construção do autômato vamos considerar o padrão $P = ababaca$.

O autômato derivado do padrão *P* é mostrado abaixo.

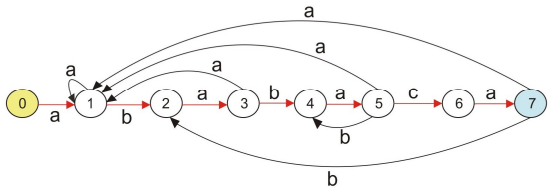

Fig. 2. Diagrama de transição de estados correspondente ao autômato de emparelhamento para o padrão $P = ababaca$.

O estado 0 é o estado inicial. O estado 7 é o único estado de aceitação. Uma aresta orientada do estado *i* para um estado *j* rotulado com a representação $\delta(i,a) = j$. As arestas direcionadas para direita, que formam a "espinha dorsal" do autômato, correspondem a seqüência de caracteres do padrão. As arestas que estão direcionadas a esquerda correspondem ao reconhecimento de prefixos durante o casamento dos caracteres do padrão e do texto. No diagrama, foram omitidas as arestas que retornam para o estado inicial.

A utilização do autômato no texto $T = abababacaba$ é mostrada abaixo. Neste exemplo, apenas uma ocorrência do padrão foi encontrada.

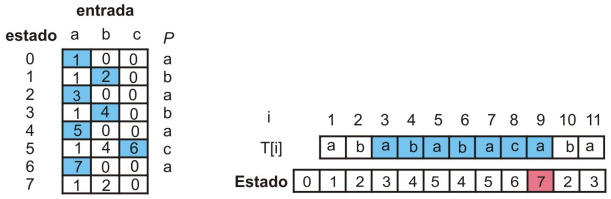

Fig. 3. Função de transição (esquerda) e operação do autômato sobre o texto T (direita).

III. AUTÔMATOS ADAPTATIVOS

Autômato adaptativo é uma instância de dispositivo adaptativo [7] que tem como camada subjacente um autômato de pilha estruturado (APE) [8].

De acordo com Neto [4], um autômato adaptativo *M* é definido por uma máquina de estados inicial E_0 a qual será submetida uma cadeia *W* de comprimento *n*. E_n é a máquina de estados final, depois de reconhecer a cadeia *W* inteira. E_i é uma máquina de estados intermediária que será submetida a uma cadeia W_i, que é uma sub-cadeia de *W*, sem os *i* primeiros elementos. Definindo *W* como sendo constituída pelos elementos $a_0a_1a_2...a_{n-1}$, pode-se dizer que a trajetória de reconhecimento do autômato adaptativo pela cadeia *W* é:

$$(E_0, a_0) \text{ -> } (E_1, a_1) \text{ -> } (E_2,a_2) \text{ ->...->} (E_{n-1},a_{n-1}).$$

No modo de operação do autômato adaptativo, o conceito de passo deve ser estendido para que se possa incorporar as ocasionais mudanças na topologia do dispositivo subjacente. Com isso, pode-se distinguir duas situações possíveis na operação dos autômatos adaptativos:

- Passo subjacente: corresponde à execução de um passo na operação do dispositivo subjacente, neste caso, o APE. Nesta situação, não ocorre nenhuma alteração no conjunto de regras que define o APE.
- Passo adaptativo: corresponde à execução de um passo de operação associado a alguma ação adaptativa não nula. Nesta situação, as componentes que definem o conjunto de regras do dispositivo subjacente são modificadas pelos efeitos da ação adaptativa correspondente e, conseqüentemente, a sua topologia deve refletir essas alterações.

Portanto, as transições que compõem o autômato adaptativo são representadas por regras de produção com a seguinte forma:

$$(\gamma g, s, \sigma\alpha), \mathcal{A} :\rightarrow (\gamma g', s', \sigma'\alpha), \mathcal{B},$$

onde:

- γ corresponde ao meta-símbolo que representa o conteúdo da pilha não considerado pelo autômato (não influencia na aplicação da produção);

- g e g' representam o estado armazenado no topo da pilha antes e depois da aplicação da transição, respectivamente;
- s e s' representam o estado corrente da transição e o próximo estado, respectivamente;
- σ é o símbolo do alfabeto que será consumido da cadeia e σ' é o símbolo do alfabeto que será empilhado na cadeia (ambos podem ser iguais a cadeia vazia, representada pelo símbolo e);
- α corresponde ao meta-símbolo que representa o restante da cadeia não considerada pelo autômato (não influencia na aplicação da produção);
- $\mathcal{A}$ e $\mathcal{B}$ são chamadas a funções adaptativas a serem executadas antes e depois da mudança de estados determinada pela produção (quando é executado um passo subjacente, $\mathcal{A} = \mathcal{B} = \emptyset$). As funções adaptativas podem ter *n* parâmetros e assumem a seguinte forma $\mathcal{A}(a_1, a_2, ...,a_n)$, em que a_i são argumentos que assumirão valores a serem usados no corpo da função. As funções adaptativas são declaradas conforme descrito em [8].

IV. MODELO ADAPTATIVO PARA O EMPARELHAMENTO DE CADEIAS

A utilização de autômato finito para solucionar o problema do emparelhamento de cadeias impõe a necessidade de um pré-processamento do padrão para a construção de um autômato de emparelhamento que reconheça aquele padrão. Dessa forma, para cada padrão a ser procurado no texto, a tecnologia adaptativa se coloca como uma alternativa para automatizar este pré-processamento. O modelo proposto por Rodrigues *et al* [1] utiliza os autômatos adaptativos na fase de construção do autômato usado na busca. Este modelo possui uma biblioteca de funções adaptativas que são usadas durante todo o processo de busca do padrão no texto. Neste modelo, para cada símbolo lido no padrão um novo estado é criado e novas transições são inseridas para todos os símbolos do alfabeto.

Na construção de um autômato é necessário saber o tamanho do padrão que será reconhecido, isto é feito através de uma submáquina inicialmente com apenas um estado e uma função adaptativa. Esta submáquina percorre o padrão e se automodifica, construindo o autômato auxiliar que representa o tamanho do padrão.

Outro autômato que deve ser construído é um autômato que reconhece exatamente uma vez a cadeia de entrada, que será usado para a busca no texto. Esse autômato possibilita que todas as ocorrências do padrão sejam encontradas consultando apenas uma vez cada símbolo do texto.

A inserção de novas transições e estados de destino no autômato deve considerar a possibilidade de sobreposição de símbolos de ocorrências sucessivas do padrão no texto. Por exemplo, se o padrão é *aba* e o texto é *ababa*, então o último caracter *a* da primeira ocorrência do padrão no texto, corresponde ao primeiro da segunda ocorrência. De acordo com as sobreposições possíveis é necessário determinar se a função cria um novo estado ou cria uma transição para um estado já existente. Para solucionar este problema, é necessário encontrar o maior prefixo de *P* que é um sufixo da subcadeia que já é reconhecida pelo autômato, concatenado com o símbolo para o qual será criado a transição. Este processo é realizado através de uma função adaptativa que cria um autômato adaptativo invertido com os símbolos do padrão, percorrendo-o de trás para frente.

Após percorrer todo o padrão e criar o autômato, o símbolo ├ (final do padrão) é encontrado. Com isso, uma função adaptativa é executada para remover as transições associadas com a função que criava os estados e acrescentava as transições no autômato de emparelhamento construído. Além disso, essa função insere uma ação adaptativa que será usada, posteriormente, para contagem do número de ocorrências do padrão no texto.

Agora com o autômato construído pode-se iniciar a busca no texto, que é iniciada no primeiro símbolo do texto e prossegue consumindo cada símbolo até o final. Cada símbolo lido faz com que o autômato mude de estado e caso alcance o estado final significa que o padrão foi encontrado no texto.

V. ADAPTOOLS

O AdapTools é um software livre, escrito em JAVA, que foi originalmente desenvolvido por Pistori [9] com o objetivo de executar autômatos adaptativos, bem como suas especializações: autômatos de pilha estruturados e autômatos de estados finitos. O estudo e a aplicação dessa ferramenta no desenvolvimento de soluções adaptativas para problemas reais podem dar suporte para o desenvolvimento de outras aplicações que façam uso de Tecnologia Adaptativa. A maior vantagem dos dispositivos baseados na tecnologia adaptativa é a sua facilidade de uso, sua relativa simplicidade e o fato de sua operação poder ser descrita de forma incremental, e seu comportamento, programado para se alterar dinamicamente em resposta aos estímulos de entrada recebidos.

O núcleo do Adaptools é composto por uma máquina virtual que executa uma versão levemente modificada de um autômato adaptativo. Essas pequenas modificações no formalismo original simplificam e uniformizam o formato de especificação de transições internas, transições externas e das ações adaptativas elementares. Com isso, todos os elementos do dispositivo podem ser apresentados através de uma única tabela [9,10].

VI. CONSIDERAÇÕES FINAIS

Este trabalho está sendo desenvolvido como projeto final de graduação do Curso de Sistemas de Informação do campus de Coxim (CPCX) da Universidade Federal de Mato Grosso do Sul (UFMS), sob orientação do Prof. Amaury Antônio de Castro Junior, com término previsto para junho de 2009.

Um dos objetivos deste trabalho em andamento é a implementação completa do modelo adaptativo proposto por Rodrigues *et al* [1], utilizando a ferramenta Adaptools. Com isso, será possível simular e analisar a solução proposta com todo o suporte gráfico da ferramenta.

Vale ressaltar que na solução adaptativa inicialmente proposta, quanto maior o tamanho do alfabeto, mais funções

adaptativas são necessárias. Dessa forma, pode-se verificar a possibilidade de criação de uma solução adaptativa para evitar essa manipulação das funções adaptativas. Além disso, pretende-se propor uma nova versão das funções adaptativas que possam reaproveitar os estados e transições de um autômato para a busca de novos padrões.

Um outro resultado previsto deste trabalho é um estudo sobre a complexidade de tempo e espaço do modelo adaptativo para o problema de emparelhamento de cadeias, bem como uma comparação com outras técnicas existentes.

REFERÊNCIAS

[1] E. S. C. Rodrigues, F. A. Rodrigues, R. L. A. Rocha, "Autômatos Adaptativos para Emparelhamento de Cadeias," in II Workshop de Tecnologia Adaptativa (WTA2008), São Paulo/SP, LTA/PCS/EPUSP, 2008.

[2] Cormen, Thomaz H.; Leiserson, Charles E.; Stein, Clifford e Rivest, Ronald L. – Algoritmos: Teoria e prática. 1.ed. Rio de Janeiro, Campus, 2002.

[3] Szwarcfiter, Jayme Luiz e Markenzon, Lilian – Estrutura de dados e seus algoritmos. 2.ed. JC, 2004.

[4] J. J. Neto, "Contribuições à metodologia de construção de compiladores", Tese de Livre Docência, USP, São Paulo,1993.

[5] Hopcroft, John E.; Motwani, Rajeev e Ullman, Jeffrey D. – Introdução à Teoria de Autômatos, Linguagens e Computação. 2.ed. Rio de Janeiro, Campus, 2002.

[6] J. J. Neto - Um Levantamento da Evolução da Adaptatividade e da Tecnologia Adaptativa - IEEE Latin America Transactions , vol. 5, no. 7, nov. 2007.

[7] J. J. Neto, Adaptive Rule-Driven Devices - General Formulation and Case Study, Lecture Notes in Computer Science, Watson, B.W. and Wood, D. (Eds.): Implementation and Application of Automata 6th International Conference, CIAA 2001, Vol.2494, Pretoria, South Africa, July 23-25, Springer-Verlag, 2002, pp. 234-250.

[8] J. J. Neto, Adaptive automata for context-dependent languages, ACM SIGPLAN Notices, Vol 29, n. 9, pp 15 – 24, 1994.

[9] Pistori, H. e Neto, J. J. AdapTools: Aspectos de Implementação e Utilização Boletim Técnico PCS, Escola Politécnica, São Paulo, 2003.

[10] JESUS, L.; SANTOS, D. G.; CASTRO JR., A. A.; PISTORI, H. AdapTools 2.0: Aspectos de Implementação e Utilização. Revista IEEE América Latina. Vol. 5, Num. 7, ISSN: 1548-0992, Novembro 2007. (p. 527-532)

Silvana Morita Melo é aluna de graduação do quarto ano do curso de bacharelado em Sistemas de informação da Universidade Federal de Mato Grosso do Sul (UFMS) Campus de Coxim (CPCX), onde desenvolveu atividades na área de desenvolvimento web e manutenção de páginas (2007), foi aluna voluntária de iniciação científica no projeto: "Estudo e avaliação das técnicas e das ferramentas para o projeto de data warehouse" (2007-2008), participou também do projeto de extensão intitulado: Todo dia é dia de parque (2008), ambos na mesma instituição.

Irani Aparecida Moreira Rodrigues é aluna de graduação do quarto ano do curso de bacharelado em Sistemas de informação da Universidade Federal de Mato Grosso do Sul (UFMS) Campus de Coxim (CPCX).

Amaury Antônio de Castro Junior é graduado em Ciência da Computação pela Universidade Federal de Mato Grosso do Sul (1997) e mestre em Ciência da Computação pela mesma universidade (2003). Atualmente, é aluno de doutorado da Escola Politécnica da USP, sob a orientação do Prof. João José Neto. É membro do Laboratório de Linguagens e Técnicas Adaptativas e do Grupo de Pesquisa em Engenharia e Computação da UCDB e professor assistente da Universidade Federal de Mato Grosso do Sul (UFMS), Campus de Coxim (CPCX). Tem experiência na área de Ciência da Computação, com ênfase em Teoria da Computação, atuando nos seguintes temas: Tecnologias Adaptativas, Autômatos Adaptativos, Projeto de Linguagens de Programação e Modelos de Computação.