Universidade de São Paulo

PCS5730: Projeto e Técnicas de Construção de Compiladores

There and back again: Convertendo autômatos para gramáticas

Lucas Virgili

1º Semestre de 2014

Sumário

1 Resumo

Neste trabalho, propomos o desenvolvimento de um conjunto de ferramentas que será usado para obter uma gramática equivalente a um autômato, representado em forma de matriz.

2 Introdução

3 Gramáticas e autômatos

4 Ferramenta para obter a gramática a partir de um autômato

Neste trabalho, desenvolveremos duas ferramentas: a primeira, dada uma matriz representando um autômato, computa a representação do autômato em expressão regular; a segunda converte uma expressão regular em uma gramática na notação escolhida pelo usuário. Logo, utilizando as duas ferramentas em um *pipeline*, obtemos uma gramática equivalente ao autômato. A figura 1 representa o processo.

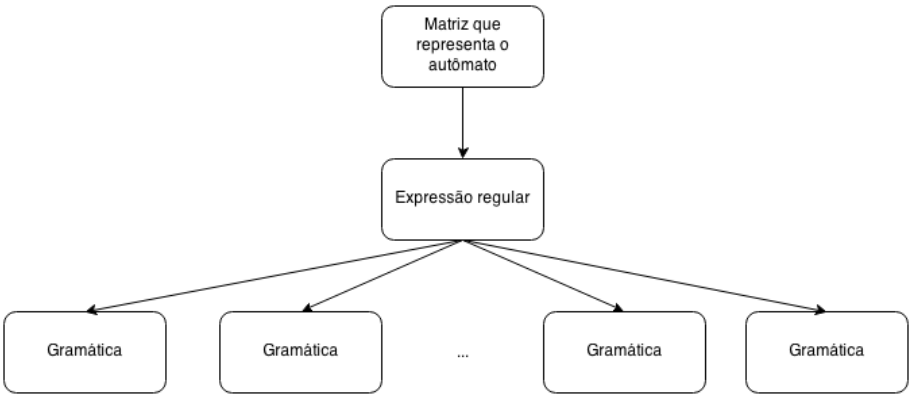

Figura 1: Otenção de gramáticas a partir do autômato

4.1 Obtendo a expressão regular

Sejam $\{q_1, q_2, \ldots, q_k\}$ os estados do autômato.

A ideia é usar um tipo de "indução": seja $R^k_{i,j}$ a expressão regular para as entradas indo do estado i ao estado j, usando somente os k primeiros estados do autômato.

Para cada par i, j, suponha que $R_{i,j}$ represente a expressão de q_i até q_j, mas sem usar o estado q_k. Agora, se pudermos usar o estado q_k, podemos montar o novo R:

$$R^k_{i,j} = R^{k-1}_{i,j} + R^{k-1}_{i,k}.R^{k-1*}_{k,k}.R^{k-1}_{k,j} \tag{1}$$

Isso quer dizer que, para irmos de i a j, podemos usar o que já sabíamos ou ir de i até o estado k, ficar em *loop* em k e depois irmos de k até o estado j.

O algoritmo, em "pseudo python":

```
# Automato Sigma:
# n estados
# R matriz

# Inicializacao:
for i in range(1, n):
    for j in range(1, n):
        if i == j:
            R[i][j][0] = epsilon
        else:
            R[i][j][0] = nada
        for terminal in Sigma:
            if transicao(i, terminal, j):
                R[i][j][0] = R[i][j][0] + terminal

# Inducao
for k in range(1, n):
    for i in range(1, n):
        for j in range(1, n):
            R[i][j][k] = R[i][j][k-1] + R[i][k][k-1]
            . estrela_kleene(R[k][k][k-1]) . R[k][j][k-1]

# Regex
inicio = find_inicio(Sigma)
for i in range(1, n):
    if is_final(i):
        regex = regex + R[inicio][i][n]
```

Note que o algoritmo não gera a expressão regular mais bonita.

5 Conclusão

6 Bibliografia

http://cs.stackexchange.com/questions/2016/how-to-convert-finite-automata-to-regular-expressions