Linguagens Formais e Autômatos (LFA)

Aula de 11/09/2013

Conjuntos Regulares, Expressões Regulares, Gramáticas Regulares e Autômatos Finitos

Conjuntos Regulares

Conjuntos regulares sobre um alfabeto finito Σ são **LINGUAGENS** definidas pela aplicação de 3 operações:

- UNIÃO
- CONCATENAÇÃO
- FECHAMENTO REFLEXIVO E TRANSITIVO

Postulados – Dado um alfabeto finito Σ:

1. $\varnothing$ é um conjunto regular sobre Σ.
2. $\{\varepsilon\}$ é um conjunto regular sobre Σ.
3. $\forall\ \sigma \in \Sigma$, $\{\sigma\}$ é um conjunto regular sobre Σ.

Dados dois conjuntos regulares X e Y, são também conjuntos regulares:

O fechamento reflexivo e transitivo de cada um; e

A união ou concatenação de um com o outro.

Notação de Kleene para Expressões Regulares

$\varnothing$ é uma expressão regular	denota o conjunto regular $\varnothing$
ε é uma expressão regular	denota o conjunto regular $\{\varepsilon\}$
Qualquer σ, se $\sigma \in \Sigma$, é uma expressão regular	denota o conjunto regular $\{\sigma\}$

Observação:
É comum expressar x.x* (ou xx*) pela notação x^+.

Se *x* e *y* são expressões regulares, então também são expressões regulares:

(x)	denota o conjunto regular X
x | y (ou x+y)	denota $X \cup Y$
x . y (ou xy)	denota X.Y
x*	denota X*

Exercícios

Qual o conjunto regular denotado pelas seguintes expressões sobre $\Sigma = \{a,b,c\}$?

a) (ab | c*)
b) a(b | c)*
c) (ab | c)*

Exemplos (Ramos, 2009 p. 148)

Exemplo 3.6 Considerem-se o alfabeto $\Sigma = \{a,b,c,d\}$ e os dois subconjuntos $A = \{a\}, B = \{b,c\}$. A seguir são apresentadas diferentes linguagens sobre Σ, definidas através da notação dos conjuntos e das expresões regulares:

- Sentenças que possuem no mínimo um símbolo a:

  $\Sigma^* A \Sigma^*$ ou $(a \mid b \mid c \mid d)^* a (a \mid b \mid c \mid d)^*$
- Sentenças que possuem exatamente dois símbolos a:

  $(\Sigma - A)^* A (\Sigma - A)^* A (\Sigma - A)^*$ ou $(b \mid c \mid d)^* a (b \mid c \mid d)^* a (b \mid c \mid d)^*$
- Sentenças que possuem um número par de símbolos a:

  $((\Sigma - A)^* A (\Sigma - A)^* A (\Sigma - A)^*)^*$ ou $((b \mid c \mid d)^* a (b \mid c \mid d)^* a (b \mid c \mid d)^*)^*$
- Sentenças que são iniciadas com o símbolo a e terminam com o símbolo b ou c:

  $A \Sigma^* B$ ou $a(a \mid b \mid c \mid d)^* (b \mid c)$
- Sentenças contendo apenas os símbolos a, b, c, com no mínimo um símbolo:

  $(A \cup B)^+$ ou $(a \mid b \mid c)^+$
- Sentenças formadas por símbolos do alfabeto $\{a,b,c,d\}$ contendo uma (e somente uma) subcadeia constituída por um símbolo do conjunto A e dois (e somente dois) do conjunto B, nesta ordem:

  $((\Sigma - A) - B)^* ABB ((\Sigma - A) - B)^*$ ou $d^* a (b \mid c)(b \mid c) d^*$

Exercícios (Conjuntos, Linguagens sobre Σ ={a,b})

Construa EXPRESSÕES REGULARES que definam corretamente uma linguagem $L \subset \Sigma^*$ definida pelas seguintes alternativas:

1. as cadeias de L possuem comprimento par
2. as cadeias de L possuem comprimento ímpar
3. as cadeias de L terminam por bbb
4. as cadeias de L não terminam com bbb

Exercício

Construa RECONHECEDORES para as cadeias da linguagem L $\subset \Sigma^*$ definida pelas seguintes alternativas:

1. as cadeias de L possuem comprimento par
2. as cadeias de L possuem comprimento ímpar
3. as cadeias de L terminam por bbb
4. as cadeias de L não terminam com bbb

Que algoritmo você seguiu?

Linguagens e Gramáticas Regulares e sua equivalência com Autômatos Finitos

Uma GR e um AF são equivalentes entre si se

- Todas as sentenças que PODEM ser aceitas pelo AF pertencem à linguagem gerada por GR e
- Todas as sentenças que pertencem à linguagem gerada por GR PODEM ser aceitas por AF.

Guia de Demonstração da Equivalência

1. Demonstrar que todas as cadeias não nulas de L(GR) – supondo que L(GR) tenha ao menos uma cadeia x tal que |x|>0 – têm derivações que correspondem diretamente a transições de AF.
2. Demonstrar que todas as cadeias não nulas aceitas por AF – supondo que haja tais cadeias – passam por transições entre o estado inicial e o final as quais estão, cada uma delas, em correspondência com regras de produção de GR.

Algoritmo para converter GR→AF

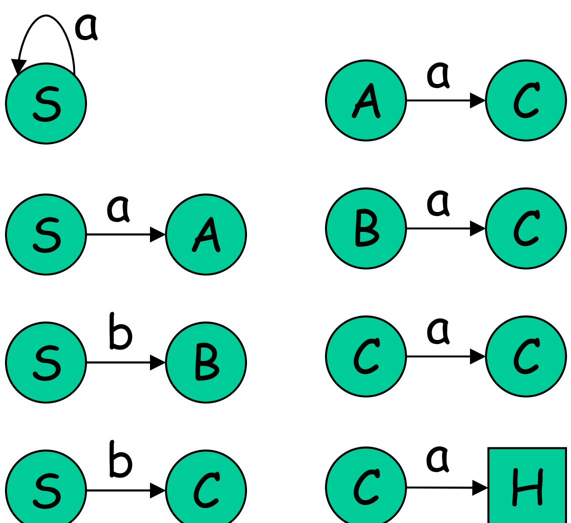
1. Para cada regra do tipo X -> yY (para X ≠ Y) gere nó-aresta-nó [X]-y->[Y]
2. Para cada regra do tipo X -> yX gere nó-aresta-nó [X]-y->[X]
3. Para cada regra do tipo X -> y gere nó-aresta-LABEL do tipo [X]-y->H
4. Combine todos os nós iguais em um só, todos os labels iguais em um só label e complete todas as arestas entre eles geradas pelos passos 1, 2 e 3
5. O nó [S] torna-se o nó inicial do AF
6. O label H gerado no passo 3 torna-se estado final do AF

Parkes, Alan P. (2009-06-29) A Concise Introduction to Languages and Machines (Undergraduate Topics in Computer Science) Springer London. Kindle Edition.

Exemplo de Conversão GR → AF

Gramática Regular (G5 do Parkes)

S -> aS | aA | bB | bC
A -> aC
B -> aC
C -> a
C -> aC

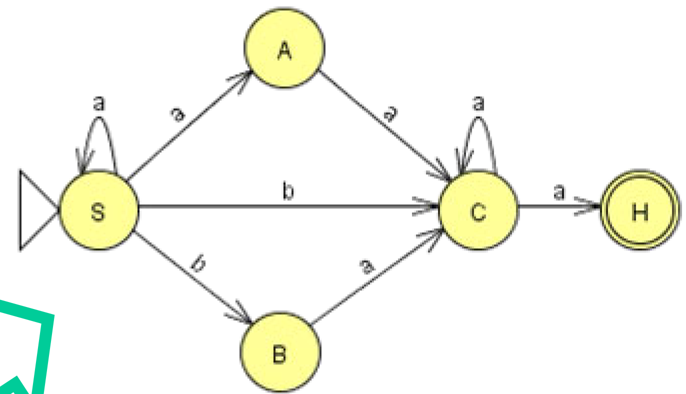


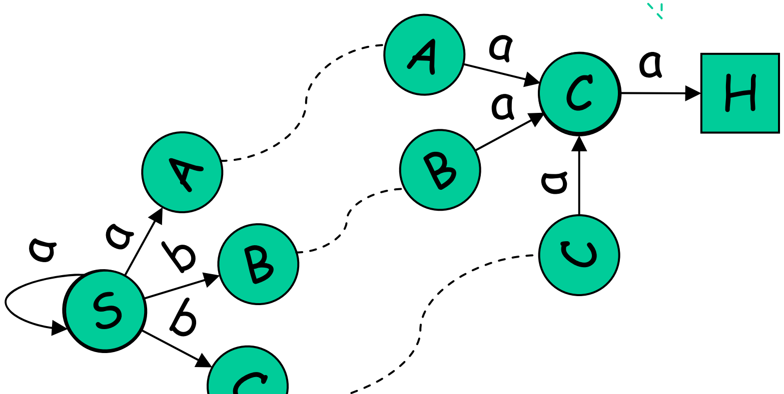
Exemplo de Conversão GR → AF

Gramática Regular (G5 do Parkes)

S -> aS | aA | bB | bC
A -> aC
B -> aC
C -> a
C -> aC

AF gerado pelo algoritmo de Parkes





Note-se que este AF é não-determinístico e pode ser minimizado (A e B são claramente redundantes, por exemplo).

Equivalência entre GR e AF

GR

$S \rightarrow aA$

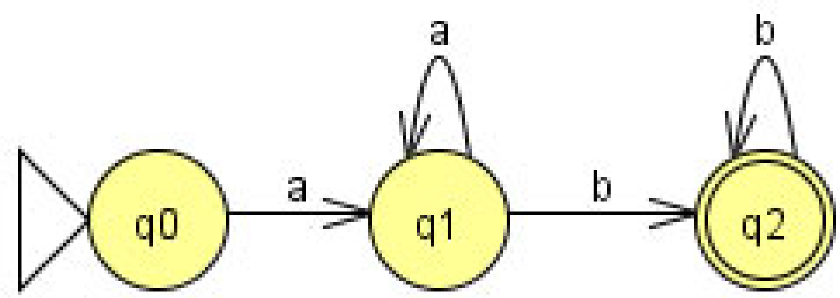
$B \rightarrow \varepsilon$

$A \rightarrow bB$

$A \rightarrow aA$

$B \rightarrow bB$

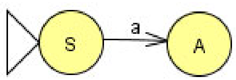
AF



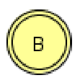
Guia de Demonstração de GR ⇔ AF

Parte 1 : Produções de GR

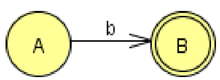
$S \rightarrow aA$



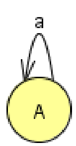
$B \rightarrow \varepsilon$



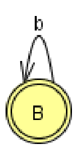
$A \rightarrow bB$



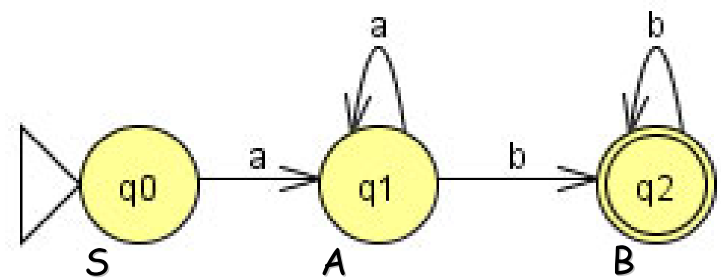
$A \rightarrow aA$



$B \rightarrow bB$

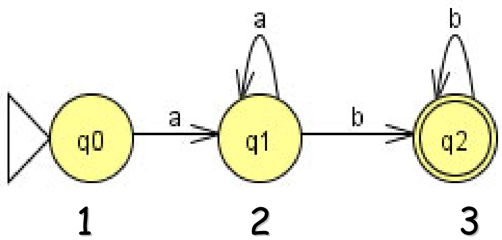


AF



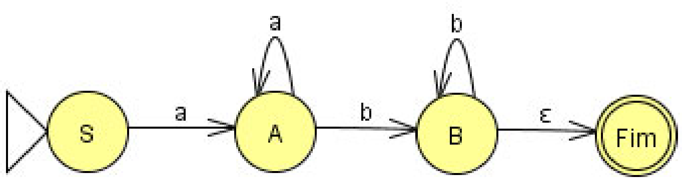
Guia de Demonstração de AF ⇔ GR

AF



Na realidade, AF é uma versão minima de um atômato equivalente AF', onde há uma última transição com a cadeia vazia, marcando uma expansão explícita de B $\rightarrow \varepsilon$.

AF'



Parte 2 : Transições de AF

q0,a,q1 ⇨ 1 → a2 (S → aA)

q1,a,q1 ⇨ 2 → a2 (A → aA)

q1,b,q2 ⇨ 2 → b3 (A → bB)

q2,b,q2 ⇨ 3 → b3 (B → bB)

$\left\{ \begin{array}{l} \text{q2},\varepsilon,\# \\ \text{q2}, \varepsilon,\text{Fim} \end{array} \right.$ ⇨ 3 → ε (B → ε)