



Compiladores

Linguagens Regulares, Expressões Regulares e Gramáticas Regulares

Artur Pereira <artur@ua.pt>,
Miguel Oliveira e Silva <mos@ua.pt>

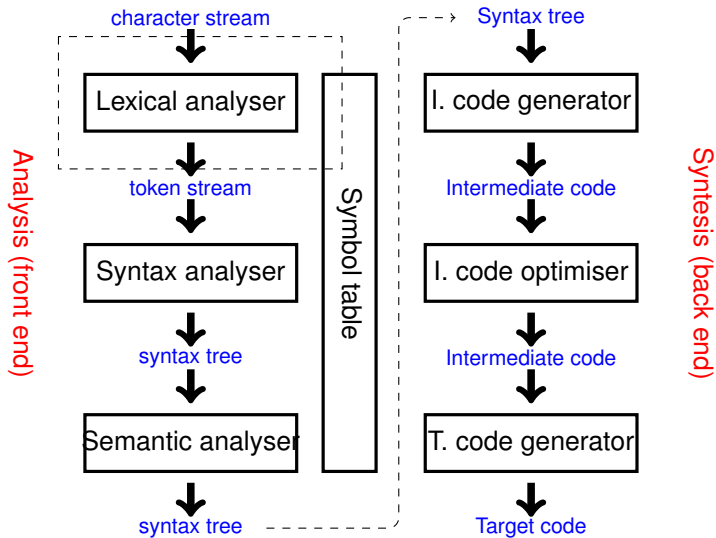
DETI, Universidade de Aveiro

Ano letivo de 2021-2022

Sumário

- 1 Análise lexical revisitada
- 2 Linguagens regulares
- 3 Expressões regulares
- 4 Gramáticas regulares
- 5 Equivalência entre expressões regulares e gramáticas regulares

Papel da análise lexical



Papel da análise lexical

- Converte a sequência de caracteres numa sequência de *tokens*
- Um *token* é um tuplo `<token-name, attribute-value>`
 - `token-name` é um símbolo (abstrato) representando um tipo de entrada
 - `attribute-value` representa o valor corrente desse símbolo

- Exemplo:

```
pos = pos + vel * 5;
```

é convertido em

```
<ID, "pos"> <=> <ID, "pos"> <+> <ID, "vel">  
<*> <INT, 5>
```

- Tipicamente, alguns símbolos são descartados pelo analisador lexical
- O conjunto dos *tokens* corresponde a uma linguagem regular
 - os *tokens* são descritos usando expressões regulares e/ou gramáticas regulares
 - são reconhecidos usando autómatos finitos

Linguagem regular

Definição

A classe das **linguagens regulares** sobre o alfabeto A define-se indutivamente da seguinte forma:

- 1 O conjunto vazio, \emptyset , é uma linguagem regular (LR).
- 2 Qualquer que seja o $a \in A$, o conjunto $\{a\}$ é uma LR.

Note que:

- em $a \in A$, a é uma letra do alfabeto
- em $\{a\}$, a é uma palavra com apenas uma letra
- Numa analogia Java, o primeiro é um 'a' e o segundo um "a"

Linguagem regular

Definição

A classe das **linguagens regulares** sobre o alfabeto A define-se indutivamente da seguinte forma:

- 1 O conjunto vazio, \emptyset , é uma linguagem regular (LR).
- 2 Qualquer que seja o $a \in A$, o conjunto $\{a\}$ é uma LR.
- 3 Se L_1 e L_2 são LR, então a sua reunião ($L_1 \cup L_2$) é uma LR.

Exemplo:

- Seja $L_1 = \{ab, c\}$, uma LR sobre o alfabeto $A = \{a, b, c\}$
- e $L_2 = \{bb, c\}$, outra LR sobre o mesmo alfabeto A
- então, $L_3 = L_1 \cup L_2 = \{ab, bb, c\}$ é uma LR sobre o mesmo alfabeto A

Linguagem regular

Definição

A classe das **linguagens regulares** sobre o alfabeto A define-se indutivamente da seguinte forma:

- 1 O conjunto vazio, \emptyset , é uma linguagem regular (LR).
- 2 Qualquer que seja o $a \in A$, o conjunto $\{a\}$ é uma LR.
- 3 Se L_1 e L_2 são LR, então a sua reunião ($L_1 \cup L_2$) é uma LR.
- 4 Se L_1 e L_2 são LR, então a sua concatenação ($L_1 \cdot L_2$) é uma LR.

Exemplo:

- Seja $L_1 = \{ab, c\}$, uma LR sobre o alfabeto $A = \{a, b, c\}$
- e $L_2 = \{bb, c\}$, outra LR sobre o mesmo alfabeto A
- então, $L_3 = L_1 \cdot L_2 = \{abbb, abc, cbb, cc\}$ é uma LR sobre o mesmo alfabeto A

Linguagem regular

Definição

A classe das **linguagens regulares** sobre o alfabeto A define-se indutivamente da seguinte forma:

- 1 O conjunto vazio, \emptyset , é uma linguagem regular (LR).
- 2 Qualquer que seja o $a \in A$, o conjunto $\{a\}$ é uma LR.
- 3 Se L_1 e L_2 são LR, então a sua reunião ($L_1 \cup L_2$) é uma LR.
- 4 Se L_1 e L_2 são LR, então a sua concatenação ($L_1 \cdot L_2$) é uma LR.
- 5 Se L_1 é uma LR, então o seu fecho de Kleene (L_1)* é uma LR.

Exemplo:

- Seja $L_1 = \{ab, c\}$, uma LR sobre o alfabeto $A = \{a, b, c\}$
- então, $L_2 = L_1^* = \{\varepsilon, ab, c, abab, abc, cab, cc, \dots\}$ é uma LR sobre o mesmo alfabeto

Linguagem regular

Definição

A classe das **linguagens regulares** sobre o alfabeto A define-se indutivamente da seguinte forma:

- 1 O conjunto vazio, \emptyset , é uma linguagem regular (LR).
- 2 Qualquer que seja o $a \in A$, o conjunto $\{a\}$ é uma LR.
- 3 Se L_1 e L_2 são LR, então a sua reunião ($L_1 \cup L_2$) é uma LR.
- 4 Se L_1 e L_2 são LR, então a sua concatenação ($L_1 \cdot L_2$) é uma LR.
- 5 Se L_1 é uma LR, então o seu fecho de Kleene $(L_1)^*$ é uma LR.
- 6 Nada mais é LR.

Note que

- $\{\varepsilon\}$ é uma LR, uma vez que $\{\varepsilon\} = \emptyset^*$.

Definição de linguagem regular

exemplo #1

Q Mostre que a linguagem L , constituída pelo conjunto dos números binários começados em 1 e terminados em 0 é uma LR sobre o alfabeto $A = \{0, 1\}$

R

- pela regra 2 (elementos primitivos), $\{0\}$ e $\{1\}$ são LR
- pela regra 3 (união), $\{0, 1\} = \{0\} \cup \{1\}$ é uma LR
- pela regra 5 (fecho), $\{0, 1\}^*$ é uma LR
- pela regra 4 (concatenação), $\{1\} \cdot \{0, 1\}^*$ é uma LR
- pela regra 4, $(\{1\} \cdot \{0, 1\}^*) \cdot \{0\}$ é uma LR
- logo, $L = \{1\} \cdot \{0, 1\}^* \cdot \{0\}$ é uma LR

Expressões regulares

Definição

O conjunto das **expressões regulares** sobre o alfabeto A define-se indutivamente da seguinte forma:

- 1 \emptyset é uma expressão regular (ER) que representa a LR $\{\}$.
- 2 Qualquer que seja o $a \in A$, a é uma ER que representa a LR $\{a\}$.
- 3 Se e_1 e e_2 são ER representando respetivamente as LR L_1 e L_2 , então $(e_1|e_2)$ é uma ER representando a LR $L_1 \cup L_2$.
- 4 Se e_1 e e_2 são ER representando respetivamente as LR L_1 e L_2 , então (e_1e_2) é uma ER representando a LR $L_1.L_2$.
- 5 Se e_1 é uma ER representando a LR L_1 , então $(e_1)^*$ é uma ER representando a LR $(L_1)^*$.
- 6 Nada mais é expressão regular.

-
- É habitual representar-se por ε a ER \emptyset^* . Representa a linguagem $\{\varepsilon\}$.

Expressões regulares

Precedência dos operadores regulares

- Na escrita de expressões regulares assume-se a seguinte precedência dos operadores:
 - fecho (*)
 - concatenação
 - escolha (|).
- O uso destas precedências permite a queda de alguns parêntesis e consequentemente uma notação simplificada.

-
- Exemplo: a expressão regular

$e_1 | e_2 e_3^*$

recorre a esta precedência para representar a expressão regular

$(e_1)|(e_2((e_3)^*))$

Expressões regulares

Exemplos

Q Determine uma ER que represente o conjunto dos números binários começados em 1 e terminados em 0.

R $1(0|1)^*0$

Q Determine uma ER que represente as sequências definidas sobre o alfabeto $A = \{a, b, c\}$ que satisfazem o requisito de qualquer b ter um a imediatamente à sua esquerda e um c imediatamente à sua direita.

R O a pode aparecer sozinho; o c também; o b , se aparecer, tem de ter um a à sua esquerda e um c à sua direita. Ou seja, pode considerar-se que as palavras da linguagem são sequências de 0 ou mais a , c ou abc .

$(a|abc|c)^*$

Q Determine uma ER que represente as sequências binárias com um número par de zeros.

R $(1^*01^*01^*)^*|1^* = 1^*(01^*01^*)^*$

Expressões regulares

Propriedades da operação de escolha

- A operação de escolha goza das propriedades:
 - comutativa: $e_1 \mid e_2 = e_2 \mid e_1$
 - associativa: $e_1 \mid (e_2 \mid e_3) = (e_1 \mid e_2) \mid e_3 = e_1 \mid e_2 \mid e_3$
 - idempotência: $e_1 \mid e_1 = e_1$
 - existência de elemento neutro: $e_1 \mid () = () \mid e_1 = e_1$

-
- Exemplo:
 - comutativa: $a \mid ab = ab \mid a$
 - associativa: $a \mid (b \mid ca) = (a \mid b) \mid ca = a \mid b \mid ca$
 - idempotência: $ab \mid ab = ab$
 - não há interesse prático em fazer uma união com o conjunto vazio
 - assim, em algumas ferramentas (por exemplo o ANTLR), $()$ representa a palavra vazia, não o conjunto vazio

Expressões regulares

Propriedades da operação de concatenação

- A operação de concatenação goza das propriedades:
 - associativa: $e_1(e_2e_3) = (e_1e_2)e_3 = e_1e_2e_3$
 - existência de elemento neutro: $e_1\varepsilon = \varepsilon e_1 = e_1$
 - existência de elemento absorvente: $e_1() = ()e_1 = ()$
 - **não goza da propriedade comutativa**

-
- Exemplo: seja $e_1 = a$, $e_2 = bc$, e $e_3 = c$
 - associativa: $a(bc\ c) = (a\ bc)c = a\ bc\ c$

Expressões regulares

Propriedades distributivas

- A combinação das operações de concatenação e escolha gozam das propriedades:

- distributiva à esquerda da concatenação em relação à escolha:

$$e_1(e_2 \mid e_3) = e_1e_2 \mid e_1e_3$$

- distributiva à direita da concatenação em relação à escolha:

$$(e_1 \mid e_2)e_3 = e_1e_3 \mid e_2e_3$$

-
- Exemplo:

- distributiva à esquerda da concatenação em relação à escolha:

$$ab(a \mid cc) = aba \mid abcc$$

- distributiva à direita da concatenação em relação à escolha:

$$(ab \mid a)cc = abcc \mid acc$$

Expressões regulares

Propriedades da operação de fecho de Kleene

- A operação de fecho goza das propriedades:

- $(e^*)^* = e^*$
- $(e_1^* \mid e_2^*)^* = (e_1 \mid e_2)^*$
- $(e_1 \mid e_2^*)^* = (e_1 \mid e_2)^*$
- $(e_1^* \mid e_2)^* = (e_1 \mid e_2)^*$

- Mas atenção:

- $(e_1 \mid e_2)^* \neq e_1^* \mid e_2^*$
- $(e_1 e_2)^* \neq e_1^* e_2^*$

-
- Exemplo:

- $b(a^*)^* = ba^*$
- $(a^* \mid b^*)^* = (a \mid b)^*$
- $(a \mid b^*)^* = (a \mid b)^*$
- $(a^* \mid b)^* = (a \mid b)^*$
- $(a \mid b)^* \neq a^* \mid b^*$
- $(ab)^* \neq a^* b^*$

Expressões regulares

Exemplos

Q Sobre o alfabeto $A = \{0, 1\}$ construa uma expressão regular que represente a linguagem

$$L = \{\omega \in A^* : \#(0, \omega) = 2\}$$

R $1^*01^*01^*$

Q Sobre o alfabeto $A = \{a, b, \dots, z\}$ construa uma expressão regular que represente a linguagem

$$L = \{\omega \in A^* : \#(a, \omega) = 3\}$$

R $(b|c|\dots|z)^*a(b|c|\dots|z)^*a(b|c|\dots|z)^*a(b|c|\dots|z)^*$

-
- Na última resposta, onde estão as reticências (...) deveriam estar todas as letras entre d e y. Parece claro que faz falta uma forma de simplificar este tipo de expressões

Expressões regulares

Extensões notacionais comuns

- uma ou mais ocorrências:

$$e^+ = e.e^*$$

- uma ou nenhuma ocorrência:

$$e? = (e|\varepsilon)$$

- um símbolo do sub-alfabeto dado:

$$[a_1 a_2 a_3 \cdots a_n] = (a_1 \mid a_2 \mid a_3 \mid \cdots \mid a_n)$$

- um símbolo do sub-alfabeto dado:

$$[a_1 - a_n] = (a_1 \mid \cdots \mid a_n)$$

- um símbolo do alfabeto fora do conjunto dado:

$$[\hat{a}_1 a_2 a_3 \cdots a_n], \quad [\hat{a}_1 - a_n]$$

Em ANTLR:

- $x..y$ é equivalente a $[x-y]$
- $\sim[abc]$ é equivalente a $[\hat{a}bc]$

Expressões regulares

Outras extensões notacionais

- n ocorrências de:

$$e\{n\} = \underbrace{e.e.\cdots.e}_n$$

- de n_1 a n_2 ocorrências:

$$e\{n_1, n_2\} = \underbrace{e.e.\cdots.e}_{n_1, n_2}$$

- n ou mais ocorrências:

$$e\{n, \} = \underbrace{e.e.\cdots.e}_{n,}$$

- $.$ representa um símbolo qualquer
- \wedge representa palavra vazia no início de linha
- $\$$ representa palavra vazia no fim de linha
- $\backslash <$ representa palavra vazia no início de palavra
- $\backslash >$ representa palavra vazia no fim de palavra

Em ANTLR:

- Pode ser feito através de predicados semânticos

Expressões regulares

Exemplos de extensões notacionais

Q Sobre o alfabeto $A = \{0, 1\}$ construa uma expressão regular que reconheça a linguagem

$$L = \{\omega \in A^* : \#(0, \omega) = 2\}$$

$$\mathcal{R} \quad 1^*01^*01^* = (1^*0)(1^*0)1^* = (1^*0)\{2\}1^*$$

Q Sobre o alfabeto $A = \{a, b, \dots, z\}$ construa uma expressão regular que reconheça a linguagem

$$L = \{\omega \in A^* : \#(a, \omega) = 3\}$$

$$\begin{aligned}\mathcal{R} \quad & (b|c|\dots|z)^*a(b|c|\dots|z)^*a(b|c|\dots|z)^*a(b|c|\dots|z)^* \\ &= ([b-z]^*a)([b-z]^*a)([b-z]^*a)[b-z]^* \\ &= ([b-z]^*a)\{3\}[b-z]^*\end{aligned}$$

Gramáticas regulares

Introdução

- Exemplo de gramática regular

$$\begin{array}{l} S \rightarrow a X \\ X \rightarrow a X \\ \quad | b X \\ \quad | \epsilon \end{array}$$

- Exemplo de gramática **não** regular

$$\begin{array}{l} S \rightarrow a S a \\ \quad | b S b \\ \quad | a \end{array}$$

-
- Letras minúsculas representam símbolos terminais e letras maiúsculas representam símbolos não terminais (o contrário do ANTLR)
 - Nas gramáticas regulares os símbolos não terminais apenas podem aparecer no fim

Gramáticas regulares

Definição

Uma **gramática regular** é um quádruplo $G = (T, N, P, S)$, onde

- T é um conjunto finito não vazio de símbolos terminais;
- N , sendo $N \cap T = \emptyset$, é um conjunto finito não vazio de símbolos não terminais;
- P é um conjunto de produções (ou regras de rescrita), cada uma da forma $\alpha \rightarrow \beta$, onde
 - $\alpha \in N$
 - $\beta \in T^* \cup T^* N$
- $S \in N$ é o símbolo inicial.

-
- A linguagem gerada por uma gramática regular é regular
 - Logo, é possível converter-se uma gramática regular numa expressão regular que represente a mesma linguagem e vice-versa

Gramáticas regulares

Operações sobre gramáticas regulares

- As gramáticas regulares são fechadas sob as operações de
 - reunião
 - concatenação
 - fecho
 - intersecção
 - complementação
- As operações de intersecção e complementação serão abordadas mais adiante através de autómatos finitos

Reunião de gramáticas regulares

Exemplo

- Q Sobre o conjunto de terminais $T = \{a, b, c\}$, determine uma gramática regular que represente a linguagem

$$L = L_1 \cup L_2$$

sabendo que

$$L_1 = \{\omega a : \omega \in T^*\} \quad L_2 = \{a\omega : \omega \in T^*\}$$

R

$S_1 \rightarrow a S_1$	$S_2 \rightarrow a X_2$
$\quad \quad b S_1$	$X_2 \rightarrow a X_2$
$\quad \quad c S_1$	$\quad \quad b X_2$
$\quad \quad a$	$\quad \quad c X_2$
	$\quad \quad \epsilon$

- Comece-se por obter as gramáticas regulares que representam L_1 e L_2 .

Reunião de gramáticas regulares

Exemplo

- Q Sobre o conjunto de terminais $T = \{a, b, c\}$, determine uma gramática regular que represente a linguagem

$$L = L_1 \cup L_2$$

sabendo que

$$L_1 = \{\omega a : \omega \in T^*\} \quad L_2 = \{a\omega : \omega \in T^*\}$$

R

$$\begin{array}{l} S_1 \rightarrow a S_1 \\ \quad | \quad b S_1 \\ \quad | \quad c S_1 \\ \quad | \quad a \end{array}$$

$$\begin{array}{l} S_2 \rightarrow a X_2 \\ X_2 \rightarrow a X_2 \\ \quad | \quad b X_2 \\ \quad | \quad c X_2 \\ \quad | \quad \varepsilon \end{array}$$

$$\begin{array}{l} S \rightarrow S_1 \mid S_2 \\ S_1 \rightarrow a S_1 \mid b S_1 \mid c S_1 \\ \quad | \quad a \\ S_2 \rightarrow a X_2 \\ X_2 \rightarrow a X_2 \mid b X_2 \mid c X_2 \\ \quad | \quad \varepsilon \end{array}$$

- E acrescentam-se as transições $S \rightarrow S_1$ e $S \rightarrow S_2$ que permitem escolher as palavras de L_1 e de L_2 , sendo S o novo símbolo inicial.

Reunião de gramáticas regulares

Algoritmo

\mathcal{D} Sejam $G_1 = (T_1, N_1, P_1, S_1)$ e $G_2 = (T_2, N_2, P_2, S_2)$ duas gramáticas regulares quaisquer, com $N_1 \cap N_2 = \emptyset$. A gramática $G = (T, N, P, S)$ onde

$$T = T_1 \cup T_2$$

$$N = N_1 \cup N_2 \cup \{S\} \quad \text{com} \quad S \notin (N_1 \cup N_2)$$

$$P = \{S \rightarrow S_1, S \rightarrow S_2\} \cup P_1 \cup P_2$$

é regular e gera a linguagem $L = L(G_1) \cup L(G_2)$.

- Para $i = 1, 2$, a nova produção $S \rightarrow S_i$ permite que G gere a linguagem $L(G_i)$

Concatenação de gramáticas regulares

Exemplo

- Q Sobre o conjunto de terminais $T = \{a, b, c\}$, determine uma gramática regular que represente a linguagem

$$L = L_1 \cdot L_2$$

sabendo que

$$L_1 = \{\omega a : \omega \in T^*\} \quad L_2 = \{a\omega : \omega \in T^*\}$$

R

$S_1 \rightarrow a S_1$	$S_2 \rightarrow a X_2$
$b S_1$	$X_2 \rightarrow a X_2$
$c S_1$	$b X_2$
a	$c X_2$
	ϵ

- Comece-se por obter as gramáticas regulares que representam L_1 e L_2 .

Concatenação de gramáticas regulares

Exemplo

- Q Sobre o conjunto de terminais $T = \{a, b, c\}$, determine uma gramática regular que represente a linguagem

$$L = L_1 \cdot L_2$$

sabendo que

$$L_1 = \{\omega a : \omega \in T^*\} \quad L_2 = \{a\omega : \omega \in T^*\}$$

R

$S_1 \rightarrow a S_1$	$S_2 \rightarrow a X_2$	$S_1 \rightarrow a S_1 \mid b S_1 \mid c S_1$
$\mid b S_1$	$X_2 \rightarrow a X_2$	$\mid a S_2$
$\mid c S_1$	$\mid b X_2$	$S_2 \rightarrow a X_2$
$\mid a$	$\mid c X_2$	$X_2 \rightarrow a X_2 \mid b X_2 \mid c X_2$
	$\mid \epsilon$	

- A seguir substitui-se $S_1 \rightarrow a$ por $S_1 \rightarrow a S_2$, de modo a impor que a segunda parte das palavras têm de pertencer a L_2

Concatenação de gramáticas regulares

Algoritmo

Sejam $G_1 = (T_1, N_1, P_1, S_1)$ e $G_2 = (T_2, N_2, P_2, S_2)$ duas gramáticas regulares quaisquer, com $N_1 \cap N_2 = \emptyset$. A gramática $G = (T, N, P, S)$ onde

$$T = T_1 \cup T_2$$

$$N = N_1 \cup N_2$$

$$P = \{A \rightarrow \omega S_2 : (A \rightarrow \omega) \in P_1 \wedge \omega \in T_1^*\} \\ \cup \{A \rightarrow \omega : (A \rightarrow \omega) \in P_1 \wedge \omega \in T_1^* N_1\} \\ \cup P_2$$

$$S = S_1$$

é regular e gera a linguagem $L = L(G_1) \cdot L(G_2)$.

- As produções da primeira gramática do tipo $\beta \in T^*$ ganham o símbolo inicial da segunda gramática no fim
- As produções da primeira gramática do tipo $\beta \in T^* N$ mantêm-se inalteradas
- As produções da segunda gramática mantêm-se inalteradas

Fecho de gramáticas regulares

Exemplo

- Q Sobre o conjunto de terminais $T = \{a, b, c\}$, determine uma gramática regular que represente a linguagem

$$L = L_1^*$$

sabendo que

$$L_1 = \{\omega a : \omega \in T^*\}$$

R

$$\begin{array}{l} S_1 \rightarrow a S_1 \\ \quad | b S_1 \\ \quad | c S_1 \\ \quad | a \end{array}$$

-
- Começa-se pela obtenção da gramática regular que representa L_1 .

Fecho de gramáticas regulares

Exemplo

- Q Sobre o conjunto de terminais $T = \{a, b, c\}$, determine uma gramática regular que represente a linguagem

$$L = L_1^*$$

sabendo que

$$L_1 = \{\omega a : \omega \in T^*\}$$

R

$$\begin{array}{l} S_1 \rightarrow a S_1 \\ \quad | \quad b S_1 \\ \quad | \quad c S_1 \\ \quad | \quad a \end{array}$$

$$\begin{array}{l} S \rightarrow \varepsilon \quad | \quad S_1 \\ S_1 \rightarrow a S_1 \quad | \quad b S_1 \quad | \quad c S_1 \\ \quad \quad | \quad a S \end{array}$$

- Acrescentando-se a transição $S \rightarrow S_1$ e substituindo-se $S_1 \rightarrow a$ por $S_1 \rightarrow a S$, permite-se iterações sobre S_1
- Acrescentando-se $S \rightarrow \varepsilon$, permite-se 0 ou mais iterações

Fecho de gramáticas regulares

Algoritmo

\mathcal{D} Seja $G_1 = (T_1, N_1, P_1, S_1)$ uma gramática regular qualquer. A gramática $G = (T, N, P, S)$ onde

$$T = T_1$$

$$N = N_1 \cup \{S\} \quad \text{com} \quad S \notin N_1$$

$$P = \{S \rightarrow \varepsilon, S \rightarrow S_1\} \\ \cup \{A \rightarrow \omega S : (A \rightarrow \omega) \in P_1 \wedge \omega \in T_1^*\} \\ \cup \{A \rightarrow \omega : (A \rightarrow \omega) \in P_1 \wedge \omega \in T_1^* N_1\}$$

é regular e gera a linguagem $L = (L(G_1))^*$.

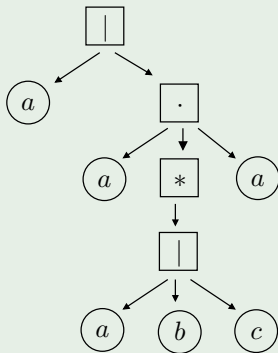
-
- As novas produções $S \rightarrow \varepsilon$ e $S \rightarrow S_1$ garantem que $(L(G_1))^n \subseteq L(G)$, para qualquer $n \geq 0$
 - As produções que só têm terminais ganham o novo símbolo inicial no fim
 - As produções que terminam num não terminal mantêm-se inalteradas

Conversão de uma ER em uma GR

exemplo

\mathcal{Q} Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

\mathcal{R}



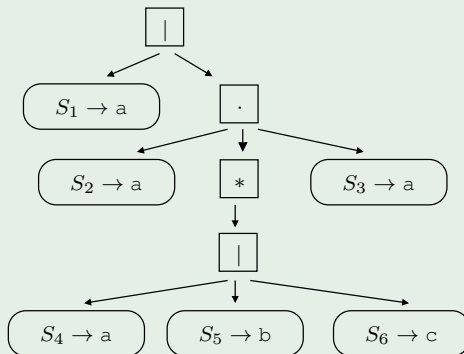
- Coloque-se de forma arbórea

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R



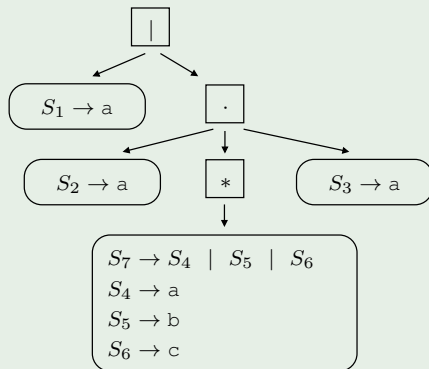
- Após converter as folhas (elementos primitivos) em GR

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R



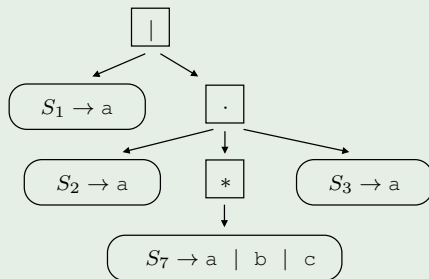
- Após aplicar a escolha (reunião) de baixo

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R



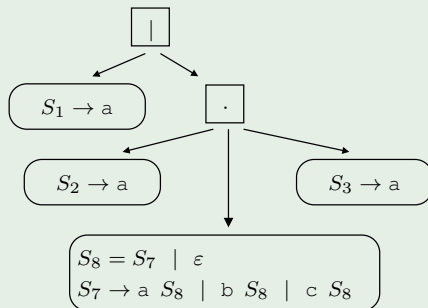
- Simplificando

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R



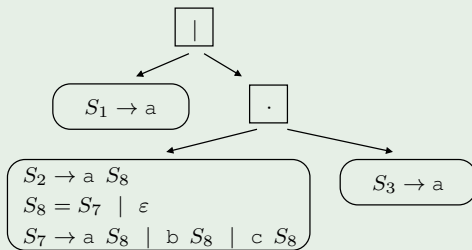
- Após aplicar o fecho

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R



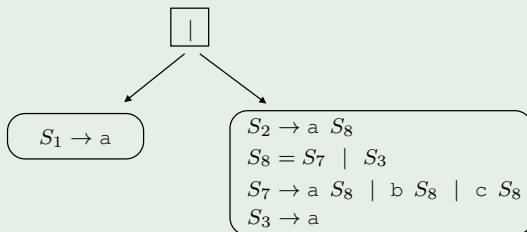
- Após aplicar a concatenação da esquerda

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R



- Após aplicar a concatenação da direita

Conversão de uma ER em uma GR

exemplo

Q Construa uma GR equivalente à ER $e = a|a(a|b|c)^*a$.

R

$$S \rightarrow S_1 \mid S_2$$

$$S_1 \rightarrow a$$

$$S_2 \rightarrow a S_8$$

$$S_8 \rightarrow S_7 \mid S_3$$

$$S_7 \rightarrow a S_8 \mid b S_8 \mid c S_8$$

$$S_3 \rightarrow a$$

e simplificando

$$S \rightarrow a \mid a S_8$$

$$S_8 \rightarrow a S_8 \mid b S_8 \mid c S_8 \mid a$$

- Finalmente após aplicar escolha (reunião) de cima

Conversão de uma ER em uma GR

Abordagem

- Dada uma expressão regular qualquer ela é:
 - ou um elemento primitivo;
 - ou uma expressão do tipo e^* , sendo e uma expressão regular qualquer;
 - ou uma expressão do tipo $e_1.e_2$, sendo e_1 e e_2 duas expressões regulares quaisquer;
 - ou uma expressão do tipo $e_1|e_2$, sendo e_1 e e_2 duas expressões regulares quaisquer;
- Identificando-se as GR equivalentes às ER primitivas, tem-se o problema resolvido, visto que se sabe como fazer a reunião, a concatenação e o fecho de GR.

expressão regular	gramática regular
ε	$S \rightarrow \varepsilon$
a	$S \rightarrow a$

Conversão de uma ER em uma GR

Algoritmo de conversão

- 1 Se a ER é do tipo primitivo, a GR correspondente pode ser obtido da tabela anterior.
- 2 Se é do tipo e^* , aplica-se este mesmo algoritmo na obtenção de uma GR equivalente à expressão regular e e, de seguida, aplica-se o fecho de GR.
- 3 Se é do tipo $e_1.e_2$, aplica-se este mesmo algoritmo na obtenção de GR para as expressões e_1 e e_2 e, de seguida, aplica-se a concatenação de GR.
- 4 Finalmente, se é do tipo $e_1|e_2$, aplica-se este mesmo algoritmo na obtenção de GR para as expressões e_1 e e_2 e, de seguida, aplica-se a reunião de GR.

-
- Na realidade, o algoritmo corresponde a um processo de decomposição arbórea a partir da raiz seguido de um processo de construção arbórea a partir das folhas.

Conversão de uma GR em uma ER

Exemplo

Q Obtenha uma ER equivalente à gramática regular seguinte

$$S \rightarrow a S \mid c S \mid aba X$$

$$X \rightarrow a X \mid c X \mid \varepsilon$$

R Abordagem admitindo expressões regulares nas produções das gramáticas

$$E \rightarrow \varepsilon S$$

$$S \rightarrow a S \mid c S \mid (aba) X$$

$$X \rightarrow a X \mid c X \mid \varepsilon \varepsilon$$

- acrescentou-se um novo símbolo inicial de forma a garantir que não aparece do lado direito

$$E \rightarrow \varepsilon S$$

$$S \rightarrow (a|c) S \mid (aba) X$$

$$X \rightarrow (a|c) X \mid \varepsilon \varepsilon$$

- transformou-se $S \rightarrow a S$ e $S \rightarrow c S$ em $S \rightarrow (a|c) S$
- fez-se algo similar com o X

$$E \rightarrow \varepsilon (a|c)^* (aba) X$$

$$X \rightarrow (a|c) X \mid \varepsilon \varepsilon$$

- transformaram-se as produções $E \rightarrow \varepsilon S$, $S \rightarrow (a|c) S$ e $S \rightarrow aba X$ em $E \rightarrow (a|c)^* aba X$
- Note que o $(a|c)$ passou a $(a|c)^*$

$$E \rightarrow \varepsilon (a|c)^* (aba) (a|c)^* \varepsilon$$

- repetiu-se com o X , obtendo-se a ER desejada: $(a|c)^* aba(a|c)^*$

Conversão de uma GR em uma ER

Exemplo

Q Obtenha uma ER equivalente à gramática regular seguinte

$$S \rightarrow a S \mid c S \mid aba X$$

$$X \rightarrow a X \mid c X \mid \varepsilon$$

R Abordagem transformando a gramática num conjunto e triplos

$\{(E, \varepsilon, S),$
 $(S, a, S), (S, c, S), (S, aba, X),$
 $(X, a, X), (X, c, X), (X, \varepsilon, \varepsilon)\}$

- *converte-se a gramática num conjunto de triplos, acrescentando um inicial*

$\{(E, \varepsilon, S), (S, (a|c), S), (S, aba, X),$
 $(X, (a|c), X), (X, \varepsilon, \varepsilon)\}$

- *transformou-se $(S, a, S), (S, c, S)$ em $(S, (a|c), S)$*
- *fez-se algo similar com o X*

$\{(E, (a|c)^* aba, X),$
 $(X, (a|c), X), (X, \varepsilon, \varepsilon)\}$

- *transformou-se o triplo de triplos $(E, \varepsilon, S), (S, (a|c), S), (S, aba, X)$ em $(E, (a|c)^* aba, X)$*
- *Note que o $(a|c)$ passou a $(a|c)^*$*

$\{(E, (a|c)^* aba(a|c)^*, \varepsilon)\}$

- *repetiu-se com o X , obtendo-se a ER desejada: $(a|c)^* aba(a|c)^*$*

Conversão de uma GR em uma ER

Algoritmo

- Uma expressão regular e que represente a mesma linguagem que a gramática regular G pode ser obtida por um processo de transformações de equivalência.
- Primeiro, converte-se a gramática $G = (T, N, P, S)$ no conjunto de triplos seguinte:

$$\begin{aligned}\mathcal{E} &= \{(E, \varepsilon, S)\} \\ &\cup \{(A, \omega, B) : (A \rightarrow \omega B) \in P \wedge B \in N\} \\ &\cup \{(A, \omega, \varepsilon) : (A \rightarrow \omega) \in P \wedge \omega \in T^*\}\end{aligned}$$

com $E \notin N$.

- A seguir, removem-se, por transformações de equivalência, um a um, todos os símbolos de N , até se obter um único triplo da forma (E, e, ε) .
- O valor de e é a expressão regular pretendida.

Conversão de uma GR em uma ER

Algoritmo de remoção dos símbolos de N

- 1 Substituir todos os triplos da forma (A, α_i, A) , com $A \in N$, por um único (A, ω_2, A) , onde $\omega_2 = \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_m$
- 2 Substituir todos os triplos da forma (A, β_i, B) , com $A, B \in N$, por um único (A, ω_1, B) , onde $\omega_1 = \beta_1 \mid \beta_2 \mid \cdots \mid \beta_n$
- 3 Substituir cada triplo de triplos da forma $(A, \omega_1, B), (B, \omega_2, B), (B, \omega_3, C)$, com $A, B, C \in N$, pelo triplo $(A, \omega_1 \omega_2^* \omega_3, C)$
- 4 Repetir os passos anteriores enquanto houver símbolos intermédios

-
- Note que, se não existir qualquer triplo do tipo (A, α_i, A) , ω_2 representa o conjunto vazio e consequentemente $\omega_2^* = \varepsilon$