



Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática
Linguagens Formais e Autómatos + Compiladores

Exame teórico-prático, parte 2

(Ano Letivo de 2018/2019)

modelo

1. Sobre o alfabeto $T_1 = \{t, b, z, w, a, v, n\}$ considere a gramática G_1 dada a seguir e seja L_1 a linguagem por ela descrita.

$$\begin{array}{lcl} P & \rightarrow & \varepsilon \mid X I t P \mid X b P z P \\ X & \rightarrow & \varepsilon \mid w C \\ I & \rightarrow & \varepsilon \mid a \\ C & \rightarrow & T \mid C o T \\ T & \rightarrow & v \mid n T \end{array}$$

- [1,5] (a) Mostre que a palavra $a t w n v b z$ pertence a L_1 .
- [1,5] (b) Avalie a veracidade da afirmação: $\{w, t\} \subset \text{first}(X I t P)$.
Apresente os passos intermédios e/ou o raciocínio adequados para suportar a sua resposta.
- [1,5] (c) Avalie a veracidade da afirmação: $t \in \text{follow}(T)$.
Apresente os passos intermédios e/ou o raciocínio adequados para suportar a sua resposta.
- [2,0] (d) Calcule o conjunto $\text{predict}(P \rightarrow X I t P)$.
Apresente os passos intermédios e/ou o raciocínio adequados para suportar a sua resposta.
- [2,0] (e) As produções começadas por P e C tornam a gramática G_1 inadequada à implementação de um reconhecedor descendente com *lookahead* de 1. Altere-a de forma a obter uma equivalente que o permita.

-
2. Considere o alfabeto $A = \{a, b, c\}$ e seja L_2 o conjunto de todas as expressões regulares definíveis sobre o alfabeto A . L_2 é uma linguagem independente do contexto definida sobre o alfabeto $T_2 = A \cup \{ (,), *, | \}$, em que $*$ é o operador de fecho, $|$ o operador de escolha e em que o operador de concatenação é implícito. Em termos de precedência, e da mais alta para a mais baixa, estão as operações de fecho, concatenação e escolha. Os parêntesis podem ser usados para alterar a precedência por defeito.
- [3,0] (.) Construa uma gramática independente do contexto que represente a linguagem L_2 .

Continua no verso

3. Sobre o alfabeto $T_3 = \{\text{NUM}, \text{BOX}, \text{CIRCLE}, \text{THICKNESS}, \text{COLOR}, \text{'{'}, \text{'}}'\}$, considere a gramática G_3 dada a seguir e seja L_3 a linguagem por ela descrita.

$$\begin{array}{ll}
 \text{draw} & \rightarrow \text{seq} \\
 \text{seq} & \rightarrow \varepsilon \\
 & | \text{seq item} \\
 \text{item} & \rightarrow \text{COLOR NUM} \\
 & | \text{THICKNESS NUM} \\
 & | \text{CIRCLE point NUM} \\
 & | \text{BOX point '{' seq '}}' \\
 \text{point} & \rightarrow \text{NUM NUM}
 \end{array}$$

Considere ainda a coleção de conjunto de itens usada na construção de um reconhecedor ascendente parcialmente apresentada a seguir, onde $\delta(Z_i, a)$ representa a função de transição de estado.

$$\begin{aligned}
 Z_0 &= \{\text{draw} \rightarrow \bullet \text{seq}, \text{seq} \rightarrow \bullet, \text{seq} \rightarrow \bullet \text{seq item}\} \\
 Z_1 &= \delta(Z_0, \text{seq}) = \{\text{draw} \rightarrow \text{seq} \bullet, \text{seq} \rightarrow \text{seq} \bullet \text{item}, \text{item} \rightarrow \bullet \text{COLOR NUM}, \text{item} \rightarrow \bullet \text{THICKNESS NUM}, \\
 &\quad \text{item} \rightarrow \bullet \text{CIRCLE point NUM}, \text{item} \rightarrow \bullet \text{BOX point '{' seq '}}'\} \\
 Z_2 &= \delta(Z_1, \text{item}) = \{\text{seq} \rightarrow \text{seq item} \bullet\} \\
 Z_3 &= \delta(Z_1, \text{COLOR}) = \{\text{item} \rightarrow \text{COLOR} \bullet \text{NUM}\} \\
 Z_4 &= \delta(Z_1, \text{THICKNESS}) = \{\text{item} \rightarrow \text{THICKNESS} \bullet \text{NUM}\} \\
 Z_5 &= \delta(Z_1, \text{CIRCLE}) = \{\dots\} \\
 Z_6 &= \delta(Z_1, \text{BOX}) = \{\dots\} \\
 Z_7 &= \delta(Z_3, \text{NUM}) = \{\text{item} \rightarrow \text{COLOR NUM} \bullet\} \\
 Z_8 &= \delta(Z_4, \text{NUM}) = \{\text{item} \rightarrow \text{THICKNESS NUM} \bullet\}
 \end{aligned}$$

- [2,0] (a) Preencha as linhas da tabela de reconhecimento (*parsing*) para um reconhecedor ascendente relativamente aos estados Z_0 a Z_4 .
- [2,0] (b) Determine os conjuntos de itens definidores dos estados Z_5 , Z_6 e de mais três, além dos apresentados.

4. Considere novamente a gramática G_3 dada no exercício anterior. Uma palavra na linguagem dada por G_3 descreve um desenho definido por uma sequência das seguintes operações gráficas (*item*):

- **COLOR NUM**, que permite mudar a cor da caneta de desenho para a dada por NUM.
- **THICKNESS NUM**, que permite mudar a espessura da caneta de desenho para a dada por NUM.
- **CIRCLE point NUM**, que desenha um circunferência centrada no ponto dado por *point* e com raio dado por NUM, usando a caneta de desenho ativa.
- **BOX point '{' seq '}}'**, que cria um sub-desenho com um *offset* dado por *point* em relação ao desenho dentro do qual fica. O ponto (0,0) do sub-desenho é o ponto *point* do desenho onde está incluído.

Apenas o símbolo terminal **NUM** tem um atributo associado, designado v e que representa um número. O símbolo não terminal *point* representa as coordenadas X e Y de um ponto. A configuração inicial do sistema é caracterizada por cor 0, espessura 1 e *offset* (0,0). Finalmente, considere que dispõe da função **drawCircle(x, y, r, c, t)** que desenha uma circunferência centrada no ponto (x,y), com raio r, usando uma caneta de desenho com cor c e espessura t.

- [1,5] (a) Trace a árvore de derivação da palavra
 COLOR NUM CIRCLE NUM NUM NUM BOX NUM NUM '{' THICKNESS NUM CIRCLE NUM NUM NUM '}}'
- Se quiser, ao traçar a árvore, pode abreviar a designação dos símbolos, desde que isso não afete a interpretação da sua resposta.
- [3,0] (b) Construa uma gramática de atributos que permita invocar a função **drawCircle** de forma adequada para cada circunferência incluída num desenho.

CÁBULA OFICIAL

ALGORITMO do first:

```
first( $\alpha$ ) {
  if ( $\alpha == \lambda$ ) then
    return { $\lambda$ }
  else if ( $\alpha == a$  and  $a \in T$ ) then
    return { $a$ }
  else if ( $\alpha == B$  and  $B \in N$ ) then
     $M = \{\}$ 
    foreach ( $B \rightarrow \gamma \in P$ )
       $M = M \cup \text{first}(\gamma)$ 
    return  $M$ 
  else /*  $|\alpha| > 1$  */
     $x = \text{head}(\alpha)$  /* the first symbol */
     $\beta = \text{tail}(\alpha)$  /* all but the first symbol */
     $M = \text{first}(x)$ 
    if  $\lambda \notin M$  then
      return  $M$ 
    else
      return  $(M - \{\lambda\}) \cup \text{first}(\beta)$ 
}
```

ALGORITMO do follow:

1. $\$ \in \text{follow}(S)$.
 2. se $(A \rightarrow \alpha B) \in P$, então $\text{follow}(B) \supseteq \text{follow}(A)$.
 3. se $(A \rightarrow \alpha B \beta) \in P$ e $\lambda \notin \text{first}(\beta)$, então $\text{follow}(B) \supseteq \text{first}(\beta)$.
 4. se $(A \rightarrow \alpha B \beta) \in P$ e $\lambda \in \text{first}(\beta)$, então $\text{follow}(B) \supseteq ((\text{first}(\beta) - \{\lambda\}) \cup \text{follow}(A))$.
-

ALGORITMO do predict:

$$\text{predict}(A \rightarrow \alpha) = \begin{cases} \text{first}(\alpha) & \lambda \notin \text{first}(\alpha) \\ (\text{first}(\alpha) - \{\lambda\}) \cup \text{follow}(A) & \lambda \in \text{first}(\alpha) \end{cases}$$
