



Compiladores

Exame teórico-prático, parte 2

(Ano Letivo de 2022-2023)

Exame modelo

Curso:

Nome:

NºMec:

ALGORITMO do predict:

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

ALGORITMO do first:

ALGORITMO do follow:

```

first( $\alpha$ ) {
  if ( $\alpha = \varepsilon$ ) then
    return { $\varepsilon$ }
   $h = \text{head}(\alpha)$  # com  $|h| = 1$ 
   $\omega = \text{tail}(\alpha)$  # tal que  $\alpha = h\omega$ 
  if ( $h \in T$ ) then
    return { $h$ }
  else
    return  $\bigcup_{(h \rightarrow \beta_i) \in P} \text{first}(\beta_i \omega)$ 
}

```

1. $\$ \in \text{follow}(S)$
2. if ($A \rightarrow \alpha B \in P$) then
 $\text{follow}(B) \supseteq \text{follow}(A)$
3. if ($A \rightarrow \alpha B \beta \in P$) \wedge ($\varepsilon \notin \text{first}(\beta)$) then
 $\text{follow}(B) \supseteq \text{first}(\beta)$
4. if ($A \rightarrow \alpha B \beta \in P$) \wedge ($\varepsilon \in \text{first}(\beta)$) then
 $\text{follow}(B) \supseteq ((\text{first}(\beta) - \{\varepsilon\}) \cup \text{follow}(A))$

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

```

P → ε | X I t P | X b P z P
X → ε | w C
I → ε | a
C → T | C o T
T → v | n T

```

- [1,5] (a) Mostre que $\mathbf{a t w n v b z} \in L_1$.
- [1,5] (b) Avalie a veracidade da afirmação: $\{\mathbf{w, t}\} \subset \text{first}(X I t P)$.
Apresente os passos intermédios e/ou o raciocínio adequados para suportar a sua resposta.
- [1,5] (c) Avalie a veracidade da afirmação: $\mathbf{t} \in \text{follow}(T)$.
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 $A = \{\mathbf{a, b, c}\}$ 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 $*$ representa o operador de fecho e $+$ o operador de escolha; operação de concatenação tem o operador implícito. Em termos de precedência, 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 na página seguinte

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

```

draw  →  seq
seq   →  ε
      |  seq item
item  →  COLOR NUM
      |  THICKNESS NUM
      |  CIRCLE point NUM
      |  BOX point '{' seq '}'
point →  NUM NUM

```

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

$$Z_0 = \{draw \rightarrow \bullet seq, seq \rightarrow \bullet, seq \rightarrow \bullet seq\ item\}$$

$$Z_1 = \delta(Z_0, seq) = \{draw \rightarrow seq \bullet, seq \rightarrow seq \bullet item, item \rightarrow \bullet COLOR\ NUM, item \rightarrow \bullet THICKNESS\ NUM, item \rightarrow \bullet CIRCLE\ point\ NUM, item \rightarrow \bullet BOX\ point\ '\{'\ seq\ '\}'\}$$

$$Z_2 = \delta(Z_1, item) = \{seq \rightarrow seq\ item \bullet\}$$

$$Z_3 = \delta(Z_1, COLOR) = \{item \rightarrow COLOR \bullet\ NUM\}$$

$$Z_4 = \delta(Z_1, THICKNESS) = \{item \rightarrow THICKNESS \bullet\ NUM\}$$

$$Z_5 = \delta(Z_1, CIRCLE) = \{\dots\}$$

$$Z_6 = \delta(Z_1, BOX) = \{\dots\}$$

$$Z_7 = \delta(Z_3, NUM) = \{item \rightarrow COLOR\ NUM \bullet\}$$

$$Z_8 = \delta(Z_4, NUM) = \{item \rightarrow THICKNESS\ NUM \bullet\}$$

[2,0] (a) Preencha as linhas da tabela de reconhecimento (*parsing*) para um reconhecedor ascendente relativamente aos estados Z_0 a Z_4 .

	NUM	BOX	CIRCLE	THICKNESS	COLOR	{	}	\$	draw	seq	item	point
Z_0												
Z_1												
Z_2												
Z_3												
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, usando N, CI, CO, T, B, s, i e p em vez de NUM, CIRCLE, COLOR, THICKNESS, BOX, seq, item e point, respetivamente.

[3,0] (b) Complete a gramática de atributos abaixo tal que ela adequadamente invoque a função `drawCircle` para cada circunferência incluída numa descrição em L_3 .

Production	Semantic rule
$draw \rightarrow seq$	
$seq \rightarrow \epsilon$	
$seq \rightarrow seq \textit{ item}$	
$item \rightarrow \text{COLOR NUM}$	
$item \rightarrow \text{THICKNESS NUM}$	
$item \rightarrow \text{CIRCLE point NUM}$	<code>drawCircle(point.x, point.y, NUM.v, ...</code>
$item \rightarrow \text{BOX point } \{ seq \}$	
$point \rightarrow \text{NUM NUM}$	