Universidade de Aveiro

Departamento de Eletrónica, Telecomunicações e Informática

Compiladores

Exame teórico-prático, parte 2 Exame modelo

Curso: **Nome:** **NºMec:**

ALGORITMO do `predict`:

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

ALGORITMO do `first`:

```
first( α ) {
    if (α = ε) then
        return {ε}
    h = head (α)        # com |h| = 1
    ω = tail (α)        # tal que α = h ω
    if ( h ∈ T ) then
        return {h}
    else
        return  ⋃_{(h→β_i)∈P}  first(β_i ω)
    }
```

ALGORITMO do `follow`:

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

---

1. Sobre o alfabeto $T_1 = \{\texttt{t b z w a o v n}\}$ 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 $\texttt{a t w n v b z} \in L_1$.

[1,5] (b) Avalie a veracidade da afirmação: $\{\texttt{w}, \texttt{t}\} \subset \texttt{first}(\text{X I t P})$.
Apresente os passos intermédios e/ou o raciocínio adequados para suportar a sua resposta.

[1,5] (c) Avalie a veracidade da afirmação: $\texttt{t} \in \texttt{follow}(\text{T})$.
Apresente os passos intermédios e/ou o raciocínio adequados para suportar a sua resposta.

[2,0] (d) Calcule o conjunto $\texttt{predict}(\text{P} \to \text{X b P z P})$.
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 $A = \{\texttt{a}, \texttt{b}, \texttt{c}\}$ 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 $T_2 = A \cup \{\texttt{(}, \texttt{)}, \texttt{*}, \texttt{+}\}$, 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 $T_3 = \{\texttt{NUM}, \texttt{BOX}, \texttt{CIRCLE}, \texttt{THICKNESS}, \texttt{COLOR}, \texttt{'\{'}, \texttt{'\}'}\}$, considere a gramática G_3 dada a seguir e seja L_3 a linguagem por ela descrita.

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

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

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

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

	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.

continua na página seguinte

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 caraterizada 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 $\rightarrow \varepsilon$	
$seq_0 \rightarrow seq_1\ item$	
item $\rightarrow$ `COLOR NUM`	
item $\rightarrow$ `THICKNESS NUM`	
item $\rightarrow$ `CIRCLE` *point* `NUM`	`drawCircle(`*point*`.x, `*point*`.y, NUM.v, ...`
item $\rightarrow$ `BOX` *point* `{` *seq* `}`	
point $\rightarrow$ $\texttt{NUM}_1\ \texttt{NUM}_2$	