Las Gramáticas LL

Gramáticas con Parsing Eficiente

Universidad de Cantabria

Outline

El Problema

Hemos visto que es posible hallar un árbol de derivación dada una gramática libre de contexto. Para ello,

- Se transforma en una gramática en forma normal de Chomsky.
- Se aplica el algoritmo CYK con la función *gen*().

El Problema

Este algoritmo es ineficiente, nos gustaría que el algoritmo fuera lineal en el tamaño de la entrada. Nos decantaremos por la estrategia descendente, es decir asumiremos que la palabra esta en el lenguaje.

El Problema

Introduciremos varios conceptos que serán necesarios para definir los autómatas con pila deterministas que aceptaran estos lenguajes.

FIRST

Definición

Sea $G := (V, \Sigma, Q_0, P)$ *una gramática libre de contexto. Para cada forma sentencial* $\alpha \in (V \cup \Sigma)^*$ *y para cada* $k \in \mathbb{N}$ *definiremos la función*

$$FIRST_k^G(\alpha) := \{x \in \Sigma^* \ : \ \begin{array}{ll} |x| = k & \exists \beta \in \Sigma^*,\ \alpha \vdash_{lm}^G x\beta \\ |x| < k & \alpha \vdash_{lm}^G x \end{array} \}.$$

Omitiremos el superíndice ^{G} *siempre que su presencia sea innecesaria por el contexto.*

FIRST

El operador $FIRST_k$ asocia a cada forma sentencial los primeros k símbolos de cualquier forma terminal alcanzable desde α mediante derivaciones “más a la izquierda”.

FIRST

Investigaremos el caso de $k = 1$ y denotaremos $\mathit{FIRST}(\alpha) := \mathit{FIRST}_1(\alpha)$.
La primera pregunta es responder como calcular el *FIRST* de cada elemento.

FIRST

Definición

Sean $L_1, \ldots, L_n \subseteq (V \cup \Sigma)^*$ *lenguajes. Definiremos el lenguaje* $L_1 \oplus_1 \cdots \oplus_1 L_n \subseteq (V \cup \Sigma)^*$ *mediante la siguiente igualdad: Sea* $j \in \{1, \ldots, n\}$ *tal que* $\lambda \in L_i$ *para* $1 \leq i \leq j-1$ *y* $\lambda \notin L_j$. *Entonces,*

$$L_1 \oplus_1 \cdots \oplus_1 L_n := \bigcup_{i=1}^{j} L_i.$$

FIRST

Teorema

Con las anteriores notaciones, se tienen las siguientes propiedades.

1. *Si $X = a$ y la gramática no contiene símbolos inútiles, entonces $FIRST(X) = \{a\}$.*
2. *Si $\alpha := X_1 \cdots X_n$ donde $X_i \in (V \cup \Sigma)^*$, entonces*

$$FIRST(\alpha) = FIRST(X_1) \oplus_1 \cdots \oplus_1 FIRST(X_n).$$

3. *Si V_λ son los símbolos no terminales que alcanzan la palabra vacía, entonces $\lambda \in FIRST(X)$ si y solamente si $X \in V_\lambda$.*

Notación

Para simplificar la notación, supongamos dada una aplicación

$$F : (V \cup \Sigma) \longrightarrow \mathcal{P}((V \cup \Sigma))^*,$$

escribiremos $\oplus_1^F \alpha$ para cada forma terminal α queriendo denotar

$$\oplus_1^F \alpha := F(X_1) \oplus_1 \cdots \oplus_1 F(X_n),$$

cuando $\alpha = X_1 \cdots X_n$.

Algoritmo

Hallar $V_\lambda := \{A \in V \ : \ A \vdash \lambda\}$.
Si $A \in \Sigma$, **entonces** $F(A) := \{A\}$
en otro caso
$G(A) := \emptyset$
$$F(A) := \begin{cases} \{A\} & \text{si } A \notin V_\lambda \\ \{A, \lambda\} & \text{si } A \in V_\lambda \end{cases}$$
mientras $F(A) \neq G(A)$ para algún $A \in V$ **hacer**
$G(A) := F(A)$
$F(A) := \{\oplus_1^F \alpha \ : \ X \mapsto \alpha, \ X \in F(A)\} \cup \{F(A)\}$
fin mientras

FOLLOW

Definition (FOLLOW)

Para cada forma sentencial $\alpha \in (V \cup \Sigma)^*$ definiremos la función $FOLLOW_k^G(\alpha)$ del modo siguiente.

- Si existe una forma sentencial $\omega\alpha$ (i.e. si $Q_0 \vdash_G \omega\alpha$), con $\omega \in (V \cup \Sigma)^*$, entonces $\lambda \in FOLLOW_k^G(\alpha)$.
- Adicionalmente, definamos

$$FOLLOW_k^G(\alpha) := \{x \in \Sigma^* \ : \ Q_0 \vdash \omega\alpha\gamma, \ x \in FIRST_k^G(\gamma)\}.$$

De nuevo, omitiremos el super–índice ^{G} cuando no genere confusión.

FOLLOW

De nuevo nos ocuparemos solamente de $FOLLOW := FOLLOW_1$. Obsérvese que $FOLLOW_k(\alpha) \subseteq \Sigma^*$ y que para cada $x \in FOLLOW_k(\alpha)$, $|x| \leq k$.

Algoritmo

Hallar $FIRST(X)$, para cada $X \in (V \cup \Sigma)$.
$G(X) := \emptyset$, para cada $X \in V$
$F(Q_0) := \{\lambda\}$
$F(A) := \emptyset$, para cada $A \neq Q_0$.
mientras $F(A) \neq G(A)$ para algún $A \in V$, **hacer**
$G(A) = F(A)$ para cada $A \in V$

$$F(A) := \left[\bigcup_{B \mapsto \omega A \omega'} \left(FIRST(\omega') \setminus \{\lambda\} \right) \right]$$

$$\bigcup \left[\bigcup_{B \mapsto \omega A \omega',\ \lambda \in FIRST(\omega')} F(B) \right] \bigcup F(A)$$