Linguaggi Formali e Traduttori

4.3 Grammatiche fattorizzabili e ricorsive a sinistra

È proibito condividere e divulgare in qualsiasi forma i materiali didattici caricati sulla piattaforma e le lezioni svolte in videoconferenza: ogni azione che viola questa norma sarà denunciata agli organi di Ateneo e perseguita a termini di legge.

Sommario

Problema

- Molte grammatiche utili per descrivere linguaggi di programmazione non sono LL(1).
    1. Presenza di **produzioni fattorizzabili**

    $$A \to \alpha\beta_1 \mid \alpha\beta_2$$

    2. Presenza di **produzioni ricorsive a sinistra**

    $$A \to A\alpha \mid \beta$$

    3. Presenza di **ambiguità**

In questa lezione

- Studiamo alcune tecniche per modificare produzioni fattorizzabili e ricorsive a sinistra senza cambiare il linguaggio generato dalla grammatica in modo da renderla – spesso, ma <u>non sempre</u> – LL(1).

Fattorizzazione

Problema

Data una grammatica con le produzioni

$$A \to \alpha\beta_1 \mid \alpha\beta_2$$

abbiamo

$$\text{GUIDA}(A \to \alpha\beta_1) \supseteq \text{FIRST}(\alpha) \qquad \text{GUIDA}(A \to \alpha\beta_2) \supseteq \text{FIRST}(\alpha)$$

dunque

$$\text{GUIDA}(A \to \alpha\beta_1) \cap \text{GUIDA}(A \to \alpha\beta_2) \neq \emptyset$$

tranne nel caso degenere in cui α genera solo ε.

Soluzione

Fattorizzare il prefisso comune α introducendo una nuova variabile A':

$$A \to \alpha A' \qquad A' \to \beta_1 \mid \beta_2$$

Esempio di fattorizzazione

La grammatica

- $S \to \texttt{if}\ E\ \texttt{then}\ S\ \texttt{else}\ S\ \texttt{fi}$
- $S \to \texttt{if}\ E\ \texttt{then}\ S\ \texttt{fi}$
- $S \to a$
- $E \to b$

non è LL(1) infatti, dovendo espandere la variabile S quando il prossimo token nella stringa da riconoscere è **`if`**, il parser non saprebbe quale delle produzioni per S usare.

La grammatica è fattorizzabile nel modo seguente:

- $S \to \texttt{if}\ E\ \texttt{then}\ S\ S'$
- $S' \to \texttt{else}\ S\ \texttt{fi} \mid \texttt{fi}$
- $S \to a$
- $E \to b$

In particolare, gli insiemi guida delle produzioni della grammatica modificata sono ora disgiunti due a due.

Ricorsione immediata a sinistra

Problema

Una grammatica con le produzioni

$$A \to A\alpha \mid \beta$$

è detta **immediatamente ricorsiva a sinistra** in quanto la produzione $A \to A\alpha$ ha A sia in testa che come primo simbolo del suo corpo. La grammatica non è LL(1):

$$\textsc{guida}(A \to A\alpha) \supseteq \textsc{first}(A) \supseteq \textsc{first}(\beta) \qquad \textsc{guida}(A \to \beta) \supseteq \textsc{first}(\beta)$$

Osservazione

La grammatica genera stringhe $\beta\alpha\alpha\cdots\alpha$ composte da <u>una</u> β seguita da <u>zero o più</u> α.

Soluzione

Introdurre una <u>nuova variabile</u> per spostare la ricorsione da sinistra a destra:

$$A \to \beta A' \qquad A' \to \varepsilon \mid \alpha A'$$

Esempio di eliminazione della ricorsione

La grammatica

- $E \to E + T \mid T$
- $T \to T * F \mid F$
- $F \to n \mid (E)$

è immediatamente ricorsiva a sinistra nelle produzioni per E e per T. Ad esempio:

$$
\begin{aligned}
\textsc{guida}(E \to E + T) &= \textsc{first}(E) = \{n, (\} \\
\textsc{guida}(E \to T) &= \textsc{first}(T) = \{n, (\}
\end{aligned}
$$

Esempio di eliminazione della ricorsione

La grammatica

- $E \to E + T \mid T$
- $T \to T * F \mid F$
- $F \to n \mid (E)$

è immediatamente ricorsiva a sinistra nelle produzioni per E e per T. Ad esempio:

$$\begin{aligned} \textsc{guida}(E \to E + T) &= \textsc{first}(E) = \{n, (\} \\ \textsc{guida}(E \to T) &= \textsc{first}(T) = \{n, (\} \end{aligned}$$

Eliminando la ricorsione immediata a sinistra otteniamo la grammatica:

- $E \to TE'$
- $E' \to +TE' \mid \varepsilon$
- $T \to FT'$
- $T' \to *FT' \mid \varepsilon$
- $F \to n \mid (E)$

Ricorsione a sinistra: caso generale

Una grammatica con le produzioni

$$A \to A\alpha_1 \mid A\alpha_2 \mid \cdots \mid A\alpha_m \mid \beta_1 \mid \beta_2 \mid \cdots \mid \beta_n$$

in cui nessun β_i inizia con A, genera stringhe della forma

$$\beta_i \alpha_{k_1} \alpha_{k_2} \cdots \alpha_{k_l}$$

L'eliminazione della ricorsione immediata a sinistra porta alla grammatica

$$\begin{aligned} A &\to \beta_1 A' \mid \beta_2 A' \mid \cdots \mid \beta_n A' \\ A' &\to \varepsilon \mid \alpha_1 A' \mid \alpha_2 A' \mid \cdots \mid \alpha_m A' \end{aligned}$$

Osservazioni

- In generale, l'eliminazione della ricorsione a sinistra <u>non garantisce</u> che la grammatica risultante sia LL(1).
- Ad esempio, nella grammatica qui sopra basta che uno degli α_i sia annullabile per avere insiemi guida delle produzioni per A' non disgiunti.

Ricorsione indiretta a sinistra

In alcune grammatiche alcune ricorsioni a sinistra sono "indirette":

$$
\begin{array}{rcl}
S & \rightarrow & Aa \mid b \\
A & \rightarrow & Ac \mid Sd \mid \varepsilon
\end{array}
$$

Tentando di eliminare la ricorsione a sinistra per le produzioni di A otteniamo

$$
\begin{array}{rcl}
S & \rightarrow & Aa \mid b \\
A & \rightarrow & SdA' \mid A' \\
A' & \rightarrow & \varepsilon \mid cA'
\end{array}
$$

ma la grammatica non è LL(1), infatti:

- $\text{GUIDA}(A \rightarrow SdA') \supseteq \text{FIRST}(S) \supseteq \text{FIRST}(A) \supseteq \text{FIRST}(A') \ni c$
- $\text{GUIDA}(A \rightarrow A') \supseteq \text{FIRST}(A') \ni c$

C'è una **ricorsione indiretta** a sinistra che riguarda la variabile A

$$A \Rightarrow Sd \Rightarrow Aad$$

Eliminazione della ricorsione indiretta

Idea

Si può esporre la ricorsione indiretta facendo opportune riscritture di variabili.

Algoritmo

1. Si impone un ordine (arbitrario) alle variabili della grammatica.
2. Considerando ogni variabile secondo l'ordine imposto, si elimina la ricorsione immediata per quella variabile e si riscrivono le occorrenze di quella variabile che compaiono nei corpi delle produzioni delle variabili seguenti.

Esempio

$$
\begin{array}{lcl}
S & \to & Aa \mid b \\
A & \to & Ac \mid Sd \mid \varepsilon
\end{array}
\Rightarrow
\begin{array}{lcl}
S & \to & Aa \mid b \\
A & \to & Ac \mid Aad \mid bd \mid \varepsilon
\end{array}
\Rightarrow
\begin{array}{lcl}
S & \to & Aa \mid b \\
A & \to & bdA' \mid A' \\
A' & \to & \varepsilon \mid cA' \mid adA'
\end{array}
$$

Esercizi

1. Applicare le trasformazioni studiate in questa lezione alla grammatica non ambigua delle formule booleane per farla diventare LL(1).
2. Implementare il parser top-down per la grammatica ottenuta nell'esercizio precedente così ottenuta. Scegliere caratteri ASCII "normali" per rappresentare i connettivi logici, ad esempio & per $\wedge$, | per $\vee$ e ~ per $\neg$.