Linguaggi Formali e Traduttori

3.1 Grammatiche libere dal contesto

Sommario

Motivazione

- Il linguaggio $L = \{a^n b^n \mid n \in \mathbb{N}\}$ **non** è regolare.
- Ponendo $a =$ (e $b =$), notiamo che L è il linguaggio delle parentesi bilanciate.
- Linguaggi come L sono di fondamentale importanza per descrivere la struttura di espressioni e blocchi nei linguaggi di programmazione.

In questa lezione

- Studiamo un approccio generativo – le **grammatiche libere** – per la descrizione di **linguaggi liberi**.
- Definiamo grammatiche per generare alcuni linguaggi liberi, tra cui L.
- Mostriamo che i linguaggi liberi includono tutti quelli regolari.

Grammatiche libere dal contesto

Definizione

Una **grammatica libera dal contesto** (o semplicemente **grammatica libera**) è una quadrupla $G = (V, T, P, S)$ dove:

- V è un insieme <u>finito</u> di **variabili** (o **simboli non terminali**, o **categorie sintattiche**).
- T è un alfabeto di simboli **terminali**.
- P è un insieme <u>finito</u> di **produzioni** della forma $A \to \alpha$, in cui:
  - $A \in V$ è detta **testa** della produzione;
  - $\alpha \in (V \cup T)^*$ è detta **corpo** della produzione.
- $S \in V$ è il **simbolo iniziale** della grammatica.

Convenzioni

- Le lettere a, b, c, ... denotano simboli terminali (elementi di T).
- Le lettere A, B, C, ... denotano variabili (elementi di V).
- Le lettere X, Y, Z, ... denotano simboli (elementi di $V \cup T$).
- Le lettere u, v, w, ... denotano stringhe di simboli terminali (elementi di T^*).
- Le lettere α, β, γ, ... denotano stringhe di simboli (elementi di $(V \cup T)^*$).
- Abbreviamo $A \to \alpha_1, \ldots, A \to \alpha_n$ con $A \to \alpha_1 \mid \cdots \mid \alpha_n$

Derivazioni

Definizione

Fissata una grammatica $G = (V, T, P, S)$, definiamo le derivazioni in un passo e in zero o più passi come segue:

- scriviamo $\alpha A \beta \Rightarrow_G \alpha\gamma\beta$ se $A \to \gamma \in P$.

  In tal caso diciamo che $\alpha\gamma\beta$ **deriva** <u>in un passo</u> da $\alpha A \beta$ in G.

- scriviamo $\Rightarrow_G^*$ per la chiusura riflessiva e transitiva di $\Rightarrow_G$, ovvero:

  - $\alpha \Rightarrow_G^* \alpha$
  - se $\alpha \Rightarrow_G \beta$ e $\beta \Rightarrow_G^* \gamma$, allora $\alpha \Rightarrow_G^* \gamma$

  Quando $\alpha \Rightarrow_G^* \beta$ diciamo che β **deriva** <u>in zero o più passi</u> da α in G.

Convenzione

Omettiamo G da $\Rightarrow_G$ e $\Rightarrow_G^*$ quando è chiaro a quale grammatica ci si riferisce.

Esempio

Data la grammatica

$$G = (\{A\}, \{0,1\}, \{A \to \varepsilon \mid 0 \mid 1 \mid 0A0 \mid 1A1\}, A)$$

abbiamo le seguenti derivazioni:

- $A \Rightarrow \varepsilon$
- $A \Rightarrow 0$
- $A \Rightarrow 1$
- $A \Rightarrow 0A0 \Rightarrow 00$
- $A \Rightarrow 1A1 \Rightarrow 101$
- $A \Rightarrow 0A0 \Rightarrow 01A10 \Rightarrow 011A110 \Rightarrow 0110110$

Note

- La **variabile** A indica una stringa palindroma arbitraria
- Le produzioni ci permettono di **riscrivere** A in una stringa palindroma più specifica
- Riscrivendo A ottengo tutte e sole le stringhe w palindrome ($w = w^R$)

Linguaggio generato da una grammatica

Definizione

Data una grammatica $G = (V, T, P, S)$, il **linguaggio generato** da G, denotato da $L(G)$ è definito come

$$L(G) \stackrel{\text{def}}{=} \{w \in T^* \mid S \Rightarrow_G^* w\}$$

Esempio

Per la grammatica G della slide precedente abbiamo

$$L(G) = \{w \in \{0,1\}^* \mid w = w^R\}$$

Definizione

Diciamo che L è un **linguaggio libero** se esiste una grammatica libera che lo genera.

Grammatiche e linguaggi regolari

Teorema

Per ogni linguaggio regolare L esiste una grammatica G tale che $L(G) = L$.

Dimostrazione

Sia $A = (Q, \Sigma, \delta, q_0, F)$ un DFA che riconosce L.

Definiamo la grammatica $G = (Q, \Sigma, P, q_0)$ dove P è così definito:

- se $q \in Q$ e $a \in \Sigma$, allora $q \to a\delta(q, a) \in P$
- se $q \in F$, allora $q \to \varepsilon \in P$

È facile vedere che $q_0 \Rightarrow^* w \Leftrightarrow \hat{\delta}(q_0, w) \in F$ da cui segue il risultato.

Osservazioni

- Il teorema mostra che le grammatiche possono generare **tutti** i linguaggi regolari.
- Sappiamo che le grammatiche possono generare linguaggi non regolari (come quello delle stringhe palindrome).
- I linguaggi liberi includono propriamente i linguaggi regolari.

Esempio: stringhe della forma $a^n b^n$

Si consideri la grammatica

$$(\{S\}, \{a, b\}, P, S)$$

in cui P è l'insieme di produzioni

- $S \to \varepsilon$
- $S \to aSb$

Alcune derivazioni

- $S \Rightarrow \varepsilon$
- $S \Rightarrow aSb \Rightarrow ab$
- $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$
- $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbb$

In generale

- $S \Rightarrow^* a^n b^n$ per ogni $n \in \mathbb{N}$

Esempio: stringhe della forma $a^m b^n$

Si consideri la grammatica

$$(\{S, A, B\}, \{a, b\}, P, S)$$

in cui P è l'insieme di produzioni:

- $S \to AB$
- $A \to \varepsilon \mid aA$
- $B \to \varepsilon \mid bB$

Alcune derivazioni

- $S \Rightarrow AB \Rightarrow A \Rightarrow \varepsilon$
- $S \Rightarrow AB \Rightarrow aAB \Rightarrow aAbB \Rightarrow abB \Rightarrow ab$
- $S \Rightarrow AB \Rightarrow aAB \Rightarrow aaAB \Rightarrow aaaAB \Rightarrow aaaB \Rightarrow aaabB \Rightarrow aaab$

In generale

- $S \Rightarrow^* a^m b^n$ per ogni $m, n \in \mathbb{N}$

Esempio: espressioni aritmetiche

Si consideri la grammatica

$$(\{E\}, \{x, y, +, *, (,)\}, P, E)$$

in cui P è l'insieme di produzioni

- $E \rightarrow x$
- $E \rightarrow y$
- $E \rightarrow E + E$
- $E \rightarrow E * E$
- $E \rightarrow (E)$

Alcune derivazioni

- $E \Rightarrow E + E \Rightarrow x + E \Rightarrow x + y$
- $E \Rightarrow E + E \Rightarrow x + E \Rightarrow x + E * E \Rightarrow x + y * E \Rightarrow x + y * y$
- $E \Rightarrow E * E \Rightarrow (E) * E \Rightarrow (E + E) * E$
  $\Rightarrow (x + E) * E \Rightarrow (x + y) * E \Rightarrow (x + y) * y$

Esempio: comando di assegnamento in Java

Si consideri la grammatica

$$(\{S, R, E\}, \{\texttt{=}, \texttt{[}, \texttt{]}, c, x\}, P, S)$$

in cui P è l'insieme di produzioni

- $S \to R \texttt{ = } E$
- $R \to x \mid R\texttt{[}E\texttt{]}$
- $E \to c \mid R$

Alcune derivazioni

- $S \Rightarrow R \texttt{ = } E \Rightarrow x \texttt{ = } E \Rightarrow x \texttt{ = } c$
- $S \Rightarrow R \texttt{ = } E \Rightarrow R \texttt{ = } R \Rightarrow x \texttt{ = } R \Rightarrow x \texttt{ = } x$
- $S \Rightarrow R \texttt{ = } E \Rightarrow R\texttt{[}E\texttt{]} \texttt{ = } E \Rightarrow x\texttt{[}E\texttt{]} \texttt{ = } E \Rightarrow x\texttt{[}c\texttt{]} \texttt{ = } E \Rightarrow x\texttt{[}c\texttt{]} \texttt{ = } c$

Alcune stringhe non derivabili

- $S \not\Rightarrow^* x$
- $S \not\Rightarrow^* c \texttt{ = } x$

Esercizi

Sulla definizione di grammatiche

Definire grammatiche per generare i seguenti linguaggi:

1. Il linguaggio generato da $(ab)^*$.
2. Le stringhe di parentesi quadre bilanciate (es. `[[][[]]]`).
3. Le stringhe di 0 e 1 della forma ww^R.
4. Stringhe di 0 e 1 in cui c'è lo stesso numero di 0 e 1 (es. `00100111`).
5. Espressioni booleane composte dalle costanti t (true), f (false) e i connettivi $\wedge$ (congiunzione), $\vee$ (disgiunzione) e $\neg$ (negazione). Ammettere la possibilità di usare parentesi (es. $t \wedge (f \vee \neg t)$ e $\neg(t \vee f)$).

Sul linguaggio generato da una grammatica

Descrivere il linguaggio generato dalle seguenti grammatiche:

1. $(\{S\}, \{a, b\}, \{S \to \varepsilon \mid aaSb\}, S)$
2. $(\{S, A\}, \{a, b\}, \{S \to \varepsilon \mid ASb, A \to \varepsilon \mid a\}, S)$
3. $(\{S, X, C\}, \{a, b, c\}, \{S \to XC, X \to \varepsilon \mid aXb, C \to \varepsilon \mid cC\}, S)$
4. $(\{S\}, \{a, b\}, \{S \to \varepsilon \mid aSb \mid bSa\}, S)$