Grammatiche context free

Corso di Fondamenti di Informatica - modulo 1
Corso di Laurea in Informatica
Università di Roma "Tor Vergata"
a.a. 2020-2021

Giorgio Gambosi

Linguaggi CF

La derivazione di una stringa generata da una grammatica di tipo 2 può essere rappresentata mediante una struttura ad albero. Tali alberi vengono chiamati alberi di derivazione, o alberi sintattici.

In un albero sintattico, ad ogni nodo interno è associato un simbolo non-terminale e ad ogni foglia è associato un simbolo terminale. Per ogni produzione del tipo $S \longrightarrow aSbA$ che viene applicata nel processo di derivazione, il nodo interno etichettato con S avrà nell'albero quattro figli etichettati con a, S, b, A

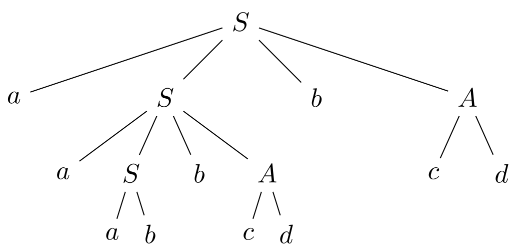
Data la grammatica $\mathcal{G}$ avente le produzioni

$$\begin{array}{lcl} S & \longrightarrow & aSbA \mid ab \\ A & \longrightarrow & cAd \mid cd \end{array}$$

la stringa $aaabbcdbcd \in L(\mathcal{G})$ può essere così derivata:

$$S \Longrightarrow aSbA \Longrightarrow aaSbAbA \Longrightarrow aaabbAbA$$
$$\Longrightarrow aaabbcdbA \Longrightarrow aaabbcdbcd.$$

Albero sintattico



Albero sintattico

In questa rappresentazione non si mantiene traccia dell'ordine con cui le produzioni sono state applicate. Ad un unico albero possono corrispondere diverse derivazioni.

Vantaggio: un albero di derivazione fornisce una descrizione sintetica della struttura sintattica della stringa, indipendentemente dall'ordine con cui le produzioni sono state applicate.

Forme ridotte e forme normali

Al fine di studiare alcune proprietà dei linguaggi generati da queste grammatiche, è utile considerare grammatiche "ristrette", comprendenti soltanto produzioni con struttura particolare.

È importante dimostrare che i linguaggi non contestuali possono essere generati mediante tali tipi di grammatiche.

Grammatica in forma ridotta

Una grammatica $\mathcal{G}$ è in forma ridotta se

1. non contiene ε-produzioni (se non, eventualmente, in corrispondenza all'assioma, ed in tal caso l'assioma non compare mai al lato destro di una produzione),

2. non contiene produzioni unitarie, cioè produzioni del tipo

$$A \longrightarrow B, \text{ con } A, B \in V_N,$$

3. non contiene simboli inutili, cioè simboli che non compaiono in nessuna derivazione di una stringa di soli terminali.

Trasformazione di una grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$ di tipo 2 in una grammatica equivalente in forma ridotta mediante sequenza di passi.

1. A partire da $\mathcal{G}$, derivazione di $\mathcal{G}_1$ di tipo 2 senza ε-produzioni tale che $L(\mathcal{G}_1) = L(\mathcal{G}) - \{\varepsilon\}$.

2. A partire da $\mathcal{G}_1$, derivazione di $\mathcal{G}_2$ di tipo 2 senza ε-produzioni e senza produzioni unitarie tale che $L(\mathcal{G}_2) = L(\mathcal{G}_1)$.

3. A partire da $\mathcal{G}_2$, derivazione di $\mathcal{G}_3$ di tipo 2 senza ε-produzioni, senza produzioni unitarie e senza simboli inutili tale che $L(\mathcal{G}_3) = L(\mathcal{G}_2)$.

4. La grammatica $\mathcal{G}_4$, di tipo 2, equivalente a $\mathcal{G}$ coincide con $\mathcal{G}_3$ se $\varepsilon \notin L(\mathcal{G})$; altrimenti, $\mathcal{G}_4$ è ottenuta da $\mathcal{G}_3$ introducendo un nuovo assioma ed un opportuno insieme di produzioni su tale simbolo.

Passo 1

Teorema 1. *Data una grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$ il cui insieme di produzioni P comprende soltanto produzioni di tipo non contestuale e produzioni vuote, esiste una grammatica non contestuale $\mathcal{G}'$ tale che $L(\mathcal{G}') = L(\mathcal{G}) - \{\varepsilon\}$.*

Passo 1

Determinazione dell'insieme $N \subseteq V_N$ dei simboli che si annullano, cioè i non terminali da cui è possibile derivare ε in $\mathcal{G}$.

Costruzione di una sequenza $N_0, N_1, \ldots, N_k = N$ di sottoinsiemi di V_N, con $N_0 = \{A \in V_N \mid A \longrightarrow \varepsilon \in P\}$ e N_{i+1} derivato da N_i:

$$N_{i+1} = N_i \cup \left\{B \in V_N \mid (B \longrightarrow \beta \in P) \wedge (\beta \in N_i^+)\right\}.$$

La costruzione termina quando $N_{k+1} = N_k$, $k \geq 0$.

$\varepsilon \in L(\mathcal{G})$ se e solo se $S \in N$

Passo 1

input grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$;
output insieme $N \subseteq V_N$ dei simboli che si annullano;
N := $\{A \in V_N \mid A \longrightarrow \varepsilon \in P\}$;
repeat
  $\widehat{N}$:= N;
  N := $\widehat{N} \cup \left\{B \in V_N \mid (B \longrightarrow \beta \in P) \wedge (\beta \in \widehat{N}^+)\right\}$
until $N = \widehat{N}$

Passo 1

Costruzione dell'insieme P' delle produzioni di $\mathcal{G}'$:

- Si esamina ciascuna produzione $A \longrightarrow \alpha$ di P, con l'esclusione delle ε-produzioni

- Se nessun simbolo di α è annullabile: $A \longrightarrow \alpha$ è inserita in P'
- Altrimenti α contiene $k > 0$ simboli che si annullano: sono inserite in P' tutte le possibili produzioni ottenute da $A \longrightarrow \alpha$ eliminando da α uno dei sottoinsiemi di simboli che si annullano

Passo 1

input grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$, insieme $N \subseteq V_N$ dei
simboli che si annullano;
output insieme P' delle produzioni di $\mathcal{G}'$;
P' := $\emptyset$;
for each $A \longrightarrow \alpha \in P$ con $\alpha \neq \varepsilon$ **do**
  sia $\alpha = Z_1, \ldots, Z_t$;
  J := $\{i \mid Z_i \in N\}$;
  for each $J' \in 2^J$ **do**
    if $J' \neq \{1, \ldots, t\}$ **then**
      sia β la stringa ottenuta eliminando da α ogni Z_i
      con $i \in J'$;
      P' := $P' \cup \{A \longrightarrow \beta\}$

Passo 1

Nel caso in cui $L(\mathcal{G})$ contiene ε, si può ottenere da $\mathcal{G}'$ una grammatica equivalente a $\mathcal{G}$ tramite la semplice introduzione di una ε-produzione sull'assioma di $\mathcal{G}'$.

Esempio

Consideriamo la grammatica $\mathcal{G} = \langle\{a, b\}, \{S, A, B\}, P, S\rangle$, le cui produzioni P sono:

$$\begin{array}{lll} S & \longrightarrow & A \mid SSa \\ A & \longrightarrow & B \mid Ab \mid \varepsilon \\ B & \longrightarrow & S \mid ab \mid aA. \end{array}$$

Esempio

Sequenza di insiemi di simboli annullabili:

$$\begin{array}{l} N_0 = \{A\} \\ N_1 = \{S, A\} \\ N_2 = \{S, A, B\} \\ N_3 = \{S, A, B\} = N_2 = N \end{array}$$

Esempio

Produzioni P':

$$\begin{array}{lll} S & \longrightarrow & A \mid SSa \mid Sa \mid a \mid \varepsilon \\ A & \longrightarrow & B \mid Ab \mid b \\ B & \longrightarrow & S \mid ab \mid aA \mid a. \end{array}$$

Passo 2

Teorema 2. *Per ogni grammatica $\mathcal{G}$ di tipo 2 senza ε-produzioni, esiste sempre una grammatica $\mathcal{G}'$ di tipo 2 senza ε-produzioni, priva di produzioni unitarie ed equivalente a $\mathcal{G}$.*

Passo 2

Sia, per ogni $A \in V_N$, $U(A)$, il sottoinsieme di $V_N - \{A\}$ comprendente tutti i non terminali derivabili da A applicando una sequenza di produzioni unitarie,

$$U(A) = \{B \in V_N - \{A\} \mid A \overset{*}{\Longrightarrow} B\}$$

Passo 2

Data la grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$, P' è costruito:

- inserendo dapprima in P' tutte le produzioni non unitarie in P
- inserendo in P', per ogni non terminale A e per ogni $B \in U(A)$, la produzione $A \longrightarrow \beta$ se e solo se in P esiste una produzione non unitaria $B \longrightarrow \beta$

Passo 2

input Grammatica CF $\mathcal{G} = \langle V_T, V_N, P, S\rangle$ priva di ε-produzioni;
output Grammatica CF $\mathcal{G}' = \langle V_T, V_N, P', S\rangle$ priva di ε-produzioni e di produzioni unitarie equivalente a $\mathcal{G}$;
$P' := \{A \longrightarrow \alpha \in P \mid \alpha \notin V_N\}$;
for each $A \in V_N$ **do**
$P' := P' \cup \{A \longrightarrow \beta \mid B \longrightarrow \beta \in P \wedge B \in U(A) \wedge \beta \notin V_N\}$

Esercizio

Costruire un algoritmo che, data una grammatica $\mathcal{G}$ di tipo 2 senza ε-produzioni e dato un non terminale A della grammatica, determini l'insieme $U(A)$

Soluzione

Passo iniziale: inserisci in $U(A)$ tutti i simboli B tali che $A \longrightarrow B$

Passo iterativo: per ogni simbolo $B \in U(A)$, inserisci in $U(A)$ tutti i simboli C tali che $B \longrightarrow C$; termina se nessun nuovo simbolo è stato inserito in $U(A)$

Passo 3

Teorema 3. *Per ogni grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$ di tipo 2 senza ε-produzioni e senza produzioni unitarie, esiste sempre una grammatica $\mathcal{G}'$ di tipo 2 senza ε-produzioni, priva di produzioni unitarie e di simboli inutili ed equivalente a $\mathcal{G}$.*

Passo 3

Affinché un simbolo $A \in V_N$ non sia inutile, è necessario che nella grammatica $\mathcal{G}$ si abbia che:

- A sia un simbolo fecondo, vale a dire che da esso siano generabili stringhe di terminali, cioè $\exists w \in V_T^+$ tale che $A \overset{*}{\Longrightarrow} w$;
- A sia generabile dall'assioma in produzioni che non contengano simboli non fecondi, cioè $S \overset{*}{\Longrightarrow} \alpha A \beta$ con $\alpha, \beta \in (V_T \cup V_N)^*$ e, per ogni $B \in V_N$ in α o β, valga la proprietà precedente.

Equivalentemente, un simbolo $A \in V_N$ non è inutile se esiste una derivazione $S \overset{*}{\Longrightarrow} \alpha A \beta \overset{*}{\Longrightarrow} w \in V_T^+$.

Passo 3

Un non terminale A è fecondo se e solo se vale una delle due condizioni seguenti:

1. esiste $w \in V_T^+$ tale che $A \longrightarrow w \in P$;

2. esiste $\alpha \in (V_N \cup V_T)^*$ tale che $A \longrightarrow \alpha \in P$ e tutti i simboli non terminali in α sono fecondi.

Passo 3

input Grammatica non contestuale $\mathcal{G} = \langle V_T, V_N, P, S \rangle$,
priva di ε-produzioni e di produzioni unitarie;
output Grammatica non contestuale $\widehat{\mathcal{G}} = \left\langle V_T, \widehat{V}_N, \widehat{P}, S \right\rangle$,
priva di ε-produzioni, di produzioni unitarie e di simboli
non fecondi, equivalente a $\mathcal{G}$;
$F \ := \ \emptyset$;
while $\exists A \in V_N - F$ per cui $\exists A \longrightarrow \alpha \in P$, con $\alpha \in (F \cup V_T)^*$ **do**
$F \ := \ F \cup \{A\}$;
$\widehat{P} \ := \ \{A \longrightarrow \alpha \mid A \longrightarrow \alpha \in P, A \in F, \alpha \in (F \cup V_T)^*\}$

Passo 3

È necessario verificare che i simboli rimasti siano generabili a partire dall'assioma.

Ciò può essere effettuato in modo iterativo, osservando che A è generabile a partire da S se vale una delle due condizioni seguenti:

1. esistono α, $\beta \ \in (F \cup V_T)^*$ tali che $S \longrightarrow \alpha A \beta \in \widehat{P}$;

2. esistono α, $\beta \ \in (F \cup V_T)^*$ e $B \in F$, generabile a partire da S, tali che $B \longrightarrow \alpha A \beta \in \widehat{P}$.

Passo 3

input Grammatica non contestuale $\widehat{\mathcal{G}} = \left\langle V_T, F, \widehat{P}, S \right\rangle$ priva
di ε-produzioni, di produzioni unitarie e di simboli
non fecondi;
output Grammatica non contestuale $\mathcal{G}' = \langle V_T, V_N', P', S \rangle$ priva
di ε-produzioni, produzioni unitarie, simboli non fecondi,
simboli non generabili da S, equivalente a $\widehat{\mathcal{G}}$;
$NG \ := \ F - \{S\}$;
for each $A \in F - NG$ **do**
for each $\alpha \in (V_T \cup F)^*$ tale che $A \longrightarrow \alpha \in \widehat{P}$ **do**
for each $B \in NG$ che appare in α **do**
$NG \ := \ NG - \{B\}$;
$V_N' \ := \ F - NG$;
$P' \ := \ \{A \longrightarrow \alpha \mid A \longrightarrow \alpha \in \widehat{P}, A \in V_N', \alpha \in (V_N' \cup V_T)^*\}$

Passo 3

Al fine di eliminare i simboli inutili (non fecondi e non generabili da S) è necessario applicare i due algoritmi nell'ordine dato: eliminare prima i simboli non generabili e poi quelli non fecondi può far sì che non tutti i simboli inutili vengano rimossi dalla grammatica.

Infatti, si consideri la grammatica

$$
\begin{array}{lll}
S & \longrightarrow & AB \mid a \\
A & \longrightarrow & a.
\end{array}
$$

Passo 3

Procedendo prima all'eliminazione dei simboli non derivabili dall'assioma e poi all'eliminazione di quelli non fecondi, otterremmo le seguenti grammatiche:

$$\begin{array}{lcl} S & \longrightarrow & AB \mid a \\ A & \longrightarrow & a \end{array} \quad \text{e successivamente} \quad \begin{array}{lcl} S & \longrightarrow & a \\ A & \longrightarrow & a. \end{array}$$

che non è in forma ridotta.

Passo 3

Se invece si procede come indicato sopra si ottengono le due grammatiche

$$\begin{array}{lcl} S & \longrightarrow & a \\ A & \longrightarrow & a \end{array} \quad \text{e successivamente} \quad S \longrightarrow a.$$

Passo 4

Una grammatica $\mathcal{G} = \langle V_T, V_N, P, S\rangle$ può essere estesa in una grammatica $\mathcal{G}' = \langle V_T, V_N', P', S'\rangle$ che generi anche la stringa vuota ε nel modo seguente:

1. $V_N' = V_N \cup \{T\}$, dove $T \notin V_N$;
2. $P' = P \cup \{T \longrightarrow \varepsilon\} \cup \{T \longrightarrow \alpha \mid S \longrightarrow \alpha \in P\}$;
3. $S' = T$.

Esempio

$$\begin{array}{llcl} 1 & S & \longrightarrow & aUVb \mid TZ \\ 2 & Z & \longrightarrow & aZ \\ 3 & U & \longrightarrow & bU \mid b \\ 4 & V & \longrightarrow & W \\ 5 & V & \longrightarrow & aY \\ 6 & Y & \longrightarrow & bY \mid b \\ 7 & W & \longrightarrow & cWd \mid cd \\ 8 & T & \longrightarrow & tT \mid tz. \end{array}$$

Esempio

- L'eliminazione delle produzioni unitarie porta ad escludere la produzione 4 e ad aggiungere una terza produzione alla 1.
- L'eliminazione di simboli non fecondi porta ad escludere la produzione 2 e la seconda produzione della 1.
- L'eliminazione dei simboli non raggiungibili porta infine ad escludere la produzione 8.

Esempio

Si ottiene quindi la grammatica

$$\begin{array}{lcl} S & \longrightarrow & aUVb \mid aUWb \\ U & \longrightarrow & bU \mid b \\ V & \longrightarrow & aY \\ Y & \longrightarrow & bY \mid b \\ W & \longrightarrow & cWd \mid cd. \end{array}$$

Esercizio

Trasformare la grammatica seguente in una grammatica equivalente in forma ridotta.

$$
\begin{array}{rcl}
S & \longrightarrow & H \mid Z \\
H & \longrightarrow & A \mid \varepsilon \\
Z & \longrightarrow & bZb \\
A & \longrightarrow & bbABa \mid a \\
B & \longrightarrow & cB \mid BZY \mid \varepsilon \\
Y & \longrightarrow & Yb \mid b.
\end{array}
$$