Indice

1 Grammatiche di tipo 2

Una grammatica è formata da quattro elementi:

$$G = \langle V, \Sigma, P, S \rangle$$

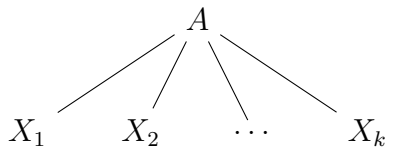
e nello specifico, in quelle di tipo 2 le produzioni hanno la forma

$$A \to \alpha \qquad A \in V, \alpha \in (V \cup \Sigma)^*$$

Una rappresentazione utile per le derivazioni di linguaggi CF sono gli alberi, ad esempio data $w \in L(G)$, allora $S \overset{*}{\Rightarrow} w$. Questa derivazione io la possono rappresentare come un albero di derivazione, o albero di parsing, o ancora parse tree. Questo è un albero

- con radice etichettata con il simbolo iniziale della grammatica
- le foglie da sinistra a destra sono w
- i nodi possono essere di tre tipi:
  - variabili, per i nodi interni
  - terminali, per le foglie
  - ϵ la parola vuota, in casi speciali per le foglie

Dato un nodo



rappresenta l'applicazione della regola di produzione

$$A \to X_1 X_2 \dots X_k \in P \qquad A \in V, \forall i \in 1, \dots, k \mid X_i \in V \cup \Sigma$$

All'ultimo livello possiamo avere nodi

solo se $A \to \epsilon \in P$.

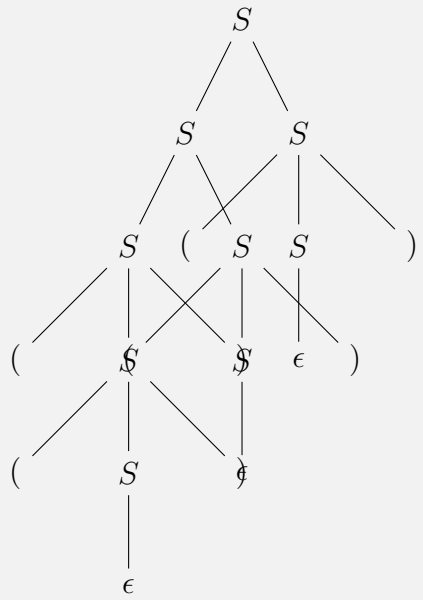
Abbiamo detto che gli automi a pila riconoscono linguaggi con ricorsione, dove questa nell'automa si esprime nella memoria a pila, nelle grammatiche si esprime nella struttura ad albero.

Definiamo la grammatica per le parentesi correttamente bilanciate

$$\begin{aligned} S &\to \epsilon \\ S &\to (S) \\ S &\to SS \end{aligned}$$

prendiamo ora la stringa $w = (())()()$ e scriviamone la derivazione

$$\begin{aligned} S &\Rightarrow SS \\ &\Rightarrow SSS \\ &\Rightarrow (S)SS \\ &\Rightarrow (S)(S)S \\ &\Rightarrow (S)(S)(S) \\ &\Rightarrow ((S))(S)(S) \\ &\Rightarrow ((S))(S)() \\ &\Rightarrow ((S))()() \\ &\Rightarrow (())()() \end{aligned}$$



Però possiamo notare che per la stessa stringa potevamo fare anche la derivazione

$$\begin{aligned} S &\Rightarrow SS \\ &\Rightarrow (S)S \\ &\Rightarrow ((S))S \\ &\Rightarrow (())S \\ &\Rightarrow (())SS \end{aligned}$$

Per evitare derivazioni multiple si utilizza un criterio detto di *derivazione leftmost*: una derivazione è lefmost se ogni volta che si fa una sostituzione sostituisco sempre la variabile più a sinistra della forma sentenziale. Si può dimostrare che esiste una corrispondenza uno a uno tra derivazioni leftmost e alberi di derivazione. La seconda e la terza derivazioni dell'esempio sono due derivazioni leftmost diverse.

Diciamo che una grammatica è ambigua se c'è una stringa che ammette almeno due alberi di derivazione – o derivazioni leftmost – diversi.

Nell'esempio di sopra si può vedere anche che ogni sottoalbero è una sequenza bilanciate di parentesi.

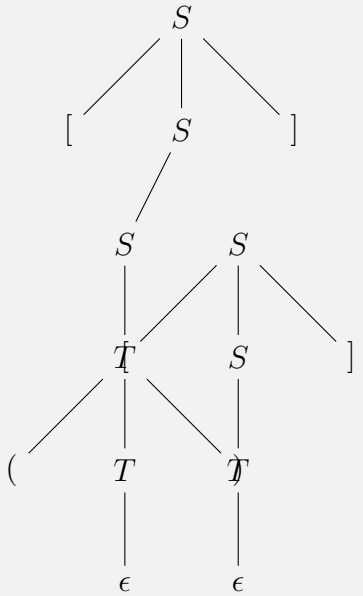
Se nella grammatica di sopra vorremmo anche le quadre, senza precedenze, questa è facilmente

$$
\begin{aligned}
S &\to \epsilon \\
S &\to (S) \\
S &\to [S] \\
S &\to SS
\end{aligned}
$$

Ma se si chiede che le quadre non possano stare all'interno delle tonde

$$
\begin{aligned}
S &\to T \\
S &\to [S] \\
S &\to SS \\
T &\to \epsilon \\
T &\to (T) \\
T &\to TT
\end{aligned}
$$

e vediamo un albero di derivazione di esempio



2 Equivalenza tra grammatiche di tipo 2 ad automi a pila

2.1 Da una grammatica che genera un linguaggio generiamo un automa che riconosce lo stesso

Data una grammatica

$$G = \langle V, \Sigma, P, S \rangle$$

di tipo 2, vogliamo costruire

$$M = \langle Q, \Sigma, \Gamma, \delta, q, Z_0, \varnothing \rangle$$

che accetta per pila vuota, con

- Q formato da un solo stato $\{q\}$
- $\Gamma = \Sigma \cup V$
- $Z_0 = S$

e δ definito come

- se $A \to \alpha \in P$ allora $(q, \alpha) \in \delta(q, \epsilon, A)$
- $\forall \sigma \in \Sigma$, $\delta(q, \sigma, \sigma) = \{(q, \epsilon)\}$, quindi si consuma il simbolo in cima alla pila

Si può dimostrare che il linguaggio generato dalla grammatica $L(G)$ è uguale al linguaggio accettato dall'automa per pila vuota $N(M)$.

Prendiamo

$$G = \langle \{S, T, U\}, \{a, b\}, P, S \rangle$$

con P definito

$$\begin{aligned} S &\to TU \\ T &\to aTb \mid \epsilon \\ U &\to bUa \mid \epsilon \end{aligned}$$

questo genera

$$L = \{a^n b^{n+m} a^m \mid n \geq 0, m \geq 0\}$$

Scriviamo le transizioni dell'automa corrispondente

$$M = \langle \{q\}, \{a, b\}, \{S, T, U, a, b\}, \delta, q, S, \varnothing \rangle$$

con δ definito come

$$\begin{aligned}
\delta(q, \epsilon, S) &= \{(q, TU)\} \\
\delta(q, \epsilon, T) &= \{(q, aTb), (q, \epsilon)\} \\
\delta(q, \epsilon, U) &= \{(q, bUa), (q, \epsilon)\} \\
\delta(q, a, a) &= \{(q, \epsilon)\} \\
\delta(q, b, b) &= \{(q, \epsilon)\}
\end{aligned}$$

Prendendo per esempio $w = abbbaa$, vediamo come viene accettata nondeterministicamente

$$\begin{aligned}(q, abbbaa, S) &\vdash (q, abbbaa, TU)\\ &\vdash (q, abbbaa, TU)\\ &\vdash (q, abbbaa, aTbU)\\ &\vdash (q, bbbaa, TbU)\\ &\vdash (q, bbbaa, bU)\\ &\vdash (q, bbaa, U)\\ &\vdash (q, bbaa, bUa)\\ &\vdash (q, baa, Ua)\\ &\vdash (q, baa, bUaa)\\ &\vdash (q, aa, Uaa)\\ &\vdash (q, aa, aa)\\ &\vdash (q, a, a)\\ &\vdash (q, \epsilon, \epsilon)\end{aligned}$$

Leggere la i terminali consumati fino a un certo punto e il contentuto della pila in quel punto restituisce la forma sentenziale durante la derivazione. Questo corrisponde a

$$\begin{aligned}S &\Rightarrow TU\\ &\Rightarrow aTbU\\ &\Rightarrow abU\\ &\Rightarrow abbUa\\ &\Rightarrow abbbUaa\\ &\Rightarrow abbbaa\end{aligned}$$

L'automa a pila tenta di simulare il processo di derivazione leftmost della stringa.

2.2 Da un automa a pila costruiamo una grammatica di tipo 2

Per la dimostrazione useremo una variazione degli automi a pila che non ne cambia la potenza computazionale. In questa forma normale

- all'inizio la pila contiene un simbolo speciale Z_0 che viene mai rimosso e non viene mai aggiunto
- alla fine l'input è stato letto completamente, la pila contiene solo Z_0 e lo stato è finale.
- le mosse sulla pila possono essere solo
  - push di un simbolo
  - pop di un simbolo
  - pila invariata

  quindi il pop non è più implicito
- se una mossa legge un simbolo da input, allora non modifica la pila. Cioè le mosse che manipolano la pila sono seperate da quelle che manipolano l'input.

In questa forma

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to 2^{Q \times \{-, \text{pop}, a \in \Gamma | \text{push}(A)\}}$$

ed abbiamo che le mosse possono avere le seguenti forme

- mosse di lettura: $(p, -) \in \delta(q, a, A) \qquad a \in \Sigma \cup \{\epsilon\}$
- pop: $(p, \text{pop}) \in \delta(q, \epsilon, A)$
- push: $(p, \text{push}(B)) \in \delta(q, \epsilon, A)$
- mosse che lasciano la pila invariata: $(p, -) \in \delta(q, \epsilon, A)$

> Ad esempio se avessimo una sequenza di parentesi ([()]()), la pila contiene inizialmente Z_0 questo disegno mostra la natura ricorsiva degli automi.

Gli automi che abbiamo visto fino ad ora possono essere simulati da questa versione normalizzata, scomponendo una mossa una pop ed una serie di push utilizzando degli stati ausiliari.