5 - Grammatiche e macchine

(Forniti in parte da Pice)

Gerarchia di Chomsky

Sia $G = (X, V, S, P)$ una grammatica, dalla sua definizione si ha:

$$P = \{v \to w | v \in (X \cup V)^+ \text{ e v contiene almeno un NT}, w \in (X \cup V)^*\}$$

Definita anche come la produzione di una coppia di stringe (dove v appartiene all'insieme di tutti i simboli con almeno un simbolo terminale e w l'unione dei simboli terminali e non terminali)

Classificazione

A seconda delle restrizioni imposte sulle regole di produzione, si distinguono le varie classi di grammatiche:

- **Tipo 0**: quando le stringhe che appaiono nella produzione non sono soggette ad alcuna limitazione
- **Tipo 1 - Dipendenti da contesto**: quando le produzioni sono limitate alla forma
  1. $yAz \to ywz$ con $A \in V,\ y, z \in (X \cup V)^*,\ w \in (X \cup V)^*$
  2. $S \to \lambda$, purchè S non compaia nella parte destra di alcuna produzione
- **Tipo 2 - Libera da contesto**: quando le produzioni sono limitate alla forma $v \to w \text{ con } v \in V$
- **Tipo 3 - Lineare destra**: quando le produzioni sono limitate alla forma
  1. $A \to bC \text{ con } A, C \in V \text{ e } b \in X$ (A e C producono terminale, b produce non terminale)
  2. $A \to b \text{ con } A \in V \text{ e } b \in X \cup \{\lambda\}$ (A produce terminale e b produce nulla)

Una grammatica di tipo ‘3’ è detta **lineare destra** perché il simbolo NT, se c’è, compare a destra (nella parte destra della produzione).
Un linguaggio generato da una tale grammatica è detto di tipo ‘3’ o lineare a destro

Teorema della gerarchia

Il **Teorema della Gerarchia di Chomsky** dimostra che le quattro classi di linguaggi formali (classificate come tipo 0, 1, 2 e 3) formano una gerarchia strettamente inclusiva, dove ogni classe è un sottoinsieme proprio della precedente.
Denotiamo con $\mathcal{L}_i$ (insieme dei linguaggi di tipo i) il seguente insieme:

$$\mathcal{L}_i = \{L \subset X^* | L = L(G), G \text{ di tipo i}\}$$

La gerarchia di Chomsky è una gerarchia in senso stretto di classi di linguaggi:

$$\mathcal{L}_3 \underset{\neq}{\subset} \mathcal{L}_2 \underset{\neq}{\subset} \mathcal{L}_1 \underset{\neq}{\subset} \mathcal{L}_0$$

Dimostrazione

$\mathcal{L}_3 \underset{\neq}{\subset} \mathcal{L}_2$

- **Inclusione**: Ogni grammatica di tipo 3 (lineare destra) è anche di tipo 2 (libera da contesto), poiché le produzioni $A \to bC$ o $A \to b$ soddisfano la definizione di grammatica libera da contesto.
- **Inclusione stretta**: Esiste almeno un linguaggio di tipo 2 che non è di tipo 3.
  Esempio:
  $L = a^n b^n \mid n > 0$
  Questo linguaggio è generato da una grammatica libera da contesto ma non può essere generato da una grammatica lineare destra.
- $\mathcal{L}_3 \subseteq \mathcal{L}_2$ discende dalle definizioni di linguaggio di tipo 3 e di grammatica di tipo 2. Infatti, si osserva facilmente che ogni grammatica di tipo 3 è anche una grammatica di tipo 2

$\mathcal{L}_2 \underset{\neq}{\subset} \mathcal{L}_1$

- **Inclusione**: Ogni grammatica libera da contesto è anche dipendente da contesto, con l'eccezione delle produzioni $A \to \lambda$ (dove $A \neq S$).
  Lo definiamo come:

  $$\forall L : L \in \mathcal{L}_2 \Leftrightarrow \exists G, \text{G è C.F.} : L = LG$$

  Tuttavia, il **Lemma della stringa vuota** permette di eliminare queste produzioni senza alterare il linguaggio generato, rendendo la grammatica di tipo 1
- **Inclusione stretta**: Esiste almeno un linguaggio di tipo 1 che non è di tipo 2.
  Esempio:
  $L = a^n b^n c^n \mid n > 0$
  Questo linguaggio è dipendente da contesto ma non libero da contesto.

Lemma della stringa vuota

Sia $G = (X, V, S, P)$ una grammatica C.F. con almeno una λ-produzione, allora esiste una grammatica C.F. G' tale che:

1. $L(G) = L(G')$ (con le due grammatiche che si equivalgono)
2. Se $\lambda \notin L(G)$ allora in G' non esistono produzioni del tipo $A \to \lambda$
3. Se $\lambda \in L(G)$ allora in G' esiste un'unica produzione $S' \to \lambda$, ove S' è il simbolo iniziale di G' ed S' non compare nella parte destra di alcuna produzione di G'

Se G ha almeno una λ-produzione, utilizziamo il Lemma della stringa vuota per determinare una grammatica C.F. G' equivalente a G, ma priva di λ-produzioni (al più, in G' compare la produzione , ed S' non compare nella parte destra di alcuna produzione di G'). G' è di tipo 1, dimostrando che $\mathcal{L}_2 \subseteq \mathcal{L}_1$

$\mathcal{L}_1 \subset \mathcal{L}_0$

- **Inclusione**: Ogni grammatica di tipo 1 è anche di tipo 0, poiché le produzioni dipendenti da contesto sono un caso particolare delle produzioni non ristrette (tipo 0).
- **Inclusione stretta**: Esistono linguaggi ricorsivamente enumerabili (tipo 0) che non sono dipendenti da contesto. La dimostrazione formale richiede nozioni avanzate come le macchine di Turing.

Operazioni sui linguaggi

Siano L_1 ed L_2 due linguaggi definiti su uno stesso alfabeto $X \quad (L_1, L_2, \subseteq X^*)$, le operazioni attuabili su essi sono:

Unione insiemistica

L'unione di due linguaggi è l'insieme di tutte le stringhe che appartengono **almeno a uno** dei due linguaggi, anche se definiti su alfabeti diversi.

$$L_1 \cup L_2 = \{w | w \in L_1 \vee w \in L_2\}$$

Concatenazione

La concatenazione genera tutte le possibili combinazioni prima una stringa di L_1, poi una di L_2

$$L_1 \cdot L_2 = \{w | w = w_1 w_2, w \in L_1 \vee w \in L_2\}$$

Iterazione

L'iterazione è un operazione unaria, si parte da un linguaggio e si fa una generalizzazione dalle parole di uno stesso linguaggio. Si ottiene un linguaggio infinito, definito con la formula:

$$L_1^* = \{w | w = w_1 w_2 \dots w_n, n \geq 0 \text{ e } \forall i : w_i \in L_1\}$$

L^* = potenza all'ennesimo di tutti i linguaggi

Complemento

Il complemento è l'insieme di tutte le parole meno l'insieme di partenza, definito con la formula

$$\overline{L_1} = X^* - L_1$$

Intersezione

L'intersezione tra due linguaggi è l'operazione di prendere due elementi in comune tra due linguaggi, quindi stringhe presenti in entrambi.

$$L_1 \cap L_2 = \{w | w \in L_1 \wedge w \in L_2\}$$

L'intersezione, la concatenazione e l'unione sono dette operazioni binarie, in quanto prevedono l'uso di due insiemi. Complemento e iterazione sono invece operazioni unarie.

Proprietà

L'operazione di concatenazione gode delle seguenti proprietà:
Dati $L_1, L_2, L_3 \subseteq X^* \quad (\equiv L_1, L_2, L_3 \in 2^{X^*})$, si possono avere:

- **Associatività**, l'ordine in cui concateni tre linguaggi non cambia il risultato: $(L_1 \cdot L_2) \cdot L_3 = L_1 \cdot (L_2 \cdot L_3)$
- **Non commutativa**,l'ordine dei linguaggi **influenza** il risultato: $L_1 \cdot L_2 \neq L_2 \cdot L_1$
- **Elemento neutro**: $L_1 \cdot \{\lambda\} = \{\lambda_1\} \cdot L_1 = L_1$

$(2^{X^*}, \cdot)$ è anch'esso un **monoide** (ovvero una struttura algebrica che ha elemento neutro e gode della proprietà commutativa) quindi, in quanto presenta:

- $L_1 \cdot \emptyset = \emptyset \cdot L_1 = \emptyset \quad$, $(\emptyset)$ è l'**elemento assorbente**
- Se un linguaggio contiene la stringa vuota ($\lambda \in L_1$ oppure $\lambda \in L_2$), valgono queste inclusioni:
  - $L_2 \subseteq L_1 \cdot L_2$
  - $L_2 \subseteq L_2 \cdot L_1$
  - $L_1 \subseteq L_1 \cdot L_2$
  - $L_1 \subseteq L_2 \cdot L_1$

Potenza di un linguaggio

Sia L un linguaggio definito su un alfabeto X, dicesi **potenza n-esima** di L, e si denota con L^n , $n \geq 0$, il seguente linguaggio:

$$L^n = \begin{cases} \{\lambda\} \text{ se n=0} \\ L^{n-1} \cdot L \text{ altrimenti} \end{cases}$$

Si ha dunque che:
$L^+ = \bigcup_{i \geq i} L^i$, (unione di tutte le potenze maggiori di 1, quindi deve avere almeno una concatenazione)

Si può definire l'unione di tutte le potenze anche con la stringa vuota:
$L^* = \{\lambda\} \cup L^+ = \bigcup_{i \geq 0} L^*$

Proprietà di chiusura delle classi di linguaggi

Un linguaggio definito su un alfabeto è un insieme di parole

Una classe di linguaggi è un insieme di linguaggi

Definizione di chiusura

Si suppone di avere un operazione binaria, definita su una coppia di linguaggi ()

Teorema di chiusura

La classe dei linguaggi di tipo $i, i = 0, 1, 2, 3$ è chiusa rispetto alle operazioni di unione, concatenazione ed iterazione.
Dati quindi due linguaggi quindi, dopo aver effettuato una di queste operazioni tra i due linguaggi, si ottiene sempre un linguaggio della stessa classe.

Dimostrazione del teorema

Lo schema generale della dimostrazione è il seguente:

- consideriamo una certa operazione, denotata con a;
- date G_1 e G_2, costruiamo una nuova grammatica G:

  $$G = (X, V, S, P)$$

  Per la quale si dimostra che:
  - se G_1 e G_2 sono di tipo i, allora G è di tipo i;
  - $L(G) = \alpha(L_1, L_2)$

[da finire]