Corso di Linguaggi di Programmazione (corso A)

Docente: Giovanni Semeraro

Capitolo 2 – Grammatiche e Linguaggi

Linguaggi formali e monoidi liberi

- Il concetto di linguaggio formale è strettamente correlato a quello di *monoide libero* (generato da un insieme).

Definizione di Alfabeto

- Un insieme *X* finito e non vuoto di simboli è un *alfabeto*.
- Esempi
  - L'alfabeto latino, con l'aggiunta dei simboli di interpunzione e dello spazio bianco: *a b c* …*z* ; , . :
  - L'insieme delle dieci cifre arabe: 0 1 … 9
- Con i simboli primitivi dell'alfabeto si formano le parole (es.: *abc*, 127, *casa*,…).

Definizione di Parola o Stringa

- Una sequenza finita di simboli $x_1x_2...x_n$, dove ogni x_i è preso da uno stesso alfabeto X è una *parola* (su X).
- Esempio
  - $X = \{0,1\}$.
  - 001110 è una parola su X
- Una parola è ottenuta giustapponendo o concatenando simboli (caratteri) dell'alfabeto.
- Se una stringa ha m simboli (non necessariamente distinti) allora diciamo che ha lunghezza m.

Lunghezza di una Parola o Stringa

- La lunghezza di una stringa w è denotata con $|w|$.
- Le parole di lunghezza 1 sono i simboli di X.
  - Quindi 001110 è una parola di lunghezza 6

$$|001110| = 6$$

- La parola vuota (o stringa vuota), denotata con λ, è una stringa priva di simboli ed ha lunghezza 0

$$|\lambda| = 0$$

Definizioni

- Uguaglianza tra stringhe
  - Due stringhe sono *uguali* se i loro caratteri, letti ordinatamente da sinistra a destra, coincidono.
- *X**
  - L'insieme di tutte le stringhe di lunghezza finita sull'alfabeto *X* si denota con *X**.
- Esempio
  - Se $X = \{0,1\}$, allora $X^* = \{\lambda, 0, 1, 00, 01, 10, 11, \ldots\}$
- *X** ha un numero di elementi che è un infinito numerabile.
  - Dalla definizione, segue che $\lambda \in X^*$, per ogni insieme *X*.

Definizioni

- Concatenazione o prodotto
  - Sia $\alpha \in X^*$ una stringa di lunghezza *m* e $\beta \in X^*$ una stringa di lunghezza *n*, la *concatenazione* di α e β, denotata con $\alpha\beta$ o $\alpha \cdot \beta$, è definita come la stringa di lunghezza *m+n*, i cui primi *m* simboli costituiscono una stringa uguale a α ed i cui ultimi *n* simboli costituiscono una stringa uguale a β.
  - Quindi se $\alpha = x_1 x_2 \ldots x_m$ e $\beta = x_1' x_2' \ldots x_n'$, si ha:

$$\alpha\beta = x_1 x_2 \ldots x_m x_1' x_2' \ldots x_n'$$

Operazione di concatenazione

- La concatenazione di stringhe su *X* è una operazione binaria su *X**:

$$\cdot\ :\ X^* \times X^* \rightarrow X^*$$

  - è associativa: $(\alpha\beta)\gamma = \alpha(\beta\gamma) = \alpha\beta\gamma, \quad \forall \alpha, \beta, \gamma \in X^*$
  - non è commutativa: $\exists \alpha, \beta \in X^*\ :\ \alpha\beta \neq \beta\alpha$
  - ha elemento neutro λ: $\lambda\alpha = \alpha\lambda = \alpha, \quad \forall \alpha \in X^*$

- Dunque $\left(X^*, \cdot\ \right)$ è un monoide (non commutativo).

Osservazione

- In base alla definizione di prodotto, ogni parola non vuota $\alpha = x_1 x_2 \ldots x_n$ si può scrivere in uno ed un solo modo come prodotto di parole di lunghezza 1, cioè di elementi di *X*.
- Ciò si esprime dicendo che:
  - $\left(X^*, \cdot\right)$ è il monoide libero generato dall'insieme *X*.

Definizioni

- Prefisso, Suffisso
  - Se $\gamma \in X^*$ è della forma $\gamma = \alpha\beta$, ove $\alpha, \beta \in X^*$, allora α è un *prefisso* di γ e β è un *suffisso* di γ.
- Sottostringa
  - Se $\delta, \beta \in X^*$ e δ è della forma $\delta = \alpha\beta\gamma$, ove $\alpha, \beta \in X^*$ allora β è una *sottostringa* di δ.
- Esempio
  - Sia γ = *00110*. Allora:
    $\{\lambda, 0, 00, 001, 0011, \gamma\}$ è l'insieme dei prefissi di γ
    $\{\lambda, 0, 10, 110, 0110, \gamma\}$ è l'insieme dei suffissi di γ
    $\{\lambda, 0, 1, 00, 01, 10, 11, 001, 011, 110, 0011, 0110, \gamma\}$
    è l'insieme delle sottostringhe di γ.

Definizioni

- Potenza di una stringa
  - Data una stringa α su X, la *potenza h*-esima di α è definita (induttivamente) come segue:

$$\alpha^h = \begin{cases} \lambda & \text{se } h = 0 \\ \alpha\alpha^{h-1} & \text{altrimenti} \end{cases}$$

    con $h = 0, 1, 2, \ldots$
  - La potenza *h*-esima di una stringa è un caso speciale di concatenamento (in quanto la si ottiene concatenando una stringa *h* volte con se stessa).

Definizioni

- Potenza di un alfabeto
  - Sia *X* un alfabeto, poniamo:

1) $X^1 = X$

2) $X^2 = \{x_1 x_2 \mid x_1, x_2 \in X,\ \ x_1 x_2 \equiv x_1 \cdot x_2\}$

3) $X^3 = \{x_1 x_2 x_3 \mid x_1 x_2 \in X^2,\ \ x_3 \in X,\ \ x_1 x_2 x_3 \equiv x_1 x_2 \cdot x_3\}$

..) ………

i) $X^i = \{x_1 x_2 \ldots x_{i-1} x_i \mid x_1 x_2 \ldots x_{i-1} \in X^{i-1},\ \ x_i \in X,\ \ x_1 x_2 \ldots x_i \equiv x_1 x_2 \ldots x_{i-1} \cdot x_i\}$

Definizioni

- **Potenza di un alfabeto**
  - Se $i \geq 2$ si ha:

$$X^{+} = X \cup X^{2} \cup \ldots \cup X^{i} \cup \ldots = \bigcup_{i=1}^{+\infty} X^{i}$$

  - Se λ è la parola vuota e prendiamo un $w \in X^{+}$ tale che $w \cdot \lambda = \lambda \cdot w = w$ si ha:

$$X^{*} = \{\lambda\} \cup X^{+}$$

  - Inoltre si ha:

$$X^{h} = \begin{cases} \{\lambda\} & \text{se } h = 0 \\ X \cdot X^{h-1} & \text{altrimenti} \end{cases}$$

Definizioni

- Linguaggio formale
  - Un *linguaggio formale L* su un alfabeto *X* è un sottoinsieme di *X**.

$$L \subseteq X^*$$

- Esempio:
  - Il linguaggio delle parentesi ben formate è un linguaggio formale in quanto, denotato con *M* tale linguaggio, si ha:

$$M \subset \{\,(\,,\,)\,\}^*$$

  - I linguaggi formali possono essere di natura molto diversa l'uno dall'altro.

Esempi di linguaggi formali

- Un linguaggio di programmazione può essere costruito a partire dall'alfabeto *X* dei simboli sulla tastiera.
- L'insieme, finito o infinito, dei programmi ben costruiti sintatticamente (ossia, che rispettano la sintassi) costituisce un linguaggio.
- Consideriamo l'insieme dei teoremi di una teoria matematica. I teoremi sono particolari stringhe di simboli del nostro alfabeto. L'insieme dei teoremi "ben formati" rappresenta un linguaggio. Ad esempio, la stringa "*ab=ba*" non è un teorema della teoria dei gruppi, ma della teoria dei gruppi abeliani.

Generazione e riconoscimento di linguaggi formali

A noi interessano i linguaggi formali da (almeno)

2 *punti di vista*

- Descrittivo/Generativo
  - Come possiamo *generare* gli elementi di un dato linguaggio *L*? Un linguaggio *finito* può essere descritto/generato per *estensione*, ossia per elencazione degli elementi (se il numero non è troppo grande). Un linguaggio *infinito* non è elencabile. Questi sono i più interessanti perché devono essere specificati necessariamente attraverso una *proprietà* che ne caratterizza gli elementi, che ne definisce l'*intensione*. Tale proprietà può essere vista come una regola da seguire per generare gli elementi del linguaggio. Il vero problema è trovare la(e) regola(e) generativa(e) (di produzione) di un linguaggio. È quello che accade quando si impara un linguaggio: non è possibile memorizzare tutte le frasi del linguaggio.

Generazione di linguaggi formali

- Esempio
  - Non è possibile “elencare” tutti i teoremi della teoria dei gruppi, perché sono infiniti i teoremi realizzabili combinando quelli noti.
  - Un libro di teoria dei gruppi non è l’elencazione dei teoremi, ma fornisce una serie di assiomi e le regole con le quali, a partire dagli assiomi, è possibile costruire tutti i teoremi della teoria dei gruppi.
  - Per descrivere la regola di produzione di un linguaggio, utilizzeremo una notazione insiemistica.

Generazione di linguaggi formali

- Esempio
  - Sia *L* il linguaggio su $X = \{0\}$ costituito da tutte e sole le stringhe che hanno un numero pari di 0, cioè:

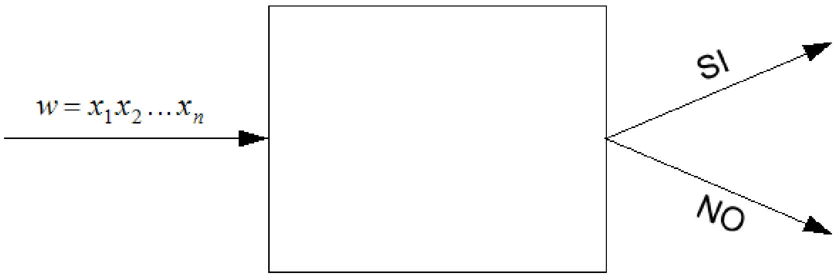
$$L = \{\lambda, 00, 0000, 000000, \ldots\}$$

La regola di produzione di *L* viene espressa come segue:

$$L = \{\lambda\} \cup \left\{ w^n \mid w = 00, n = 1,2,\ldots \right\}$$

Generazione e riconoscimento di linguaggi formali

- A noi interessano i linguaggi formali da almeno due *punti di vista*
  - Riconoscitivo
    - Come possiamo *riconoscere* gli elementi di un dato linguaggio *L*? Questo secondo punto di vista ha come obiettivo la costruzione di "macchine" in grado di decidere/stabilire se una stringa è un elemento di *L* oppure no. Si intende costruire una "macchinetta" cui dare in ingresso una particolare parola e che produce una tra due possibili risposte: $sì \equiv ' \in L'$ e $no \equiv ' \notin L'$



Riconoscimento di linguaggi formali

- Esempio
  - L’esecuzione di un programma errato sintatticamente viene inibita. Questo è indice dell’esistenza di una “macchinetta” che stabilisce se il programma appartiene o no all’insieme dei programmi sintatticamente ben costruiti.
  - Analizziamo il problema della generazione di *L*.

Riconoscimento di linguaggi formali: esempio

- Sia dato l'alfabeto: $X = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -\}$

  Voglio generare il linguaggio *L* dei numeri interi relativi. Ovviamente: $L \subseteq X^*$

  Più precisamente, $L \subset X^*$ poiché, ad esempio,

  $$1 + + - 5 \notin L$$

  Non possiamo elencare gli elementi di *L*. Cerchiamo dunque una serie di regole mediante le quali è possibile produrre tutti e soli gli elementi di *L*.

  Assumiamo, per semplicità, che un numero relativo sia costituito da una serie di cifre precedute da + o -.

Riconoscimento di linguaggi formali: esempio

- Adottiamo la BNF per descrivere le produzioni:

$$< S > ::= + < I > \mid - < I >$$

$$< I > ::= < D > \mid < I > < D >$$

$$< D > ::= 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

  - Queste regole generano tutti gli interi relativi purché partiamo dal simbolo nonterminale *S*.
  - *I* è il simbolo nonterminale (da ora in poi, talvolta abbreviato in *NT*), anche detto *categoria sintattica*, che sta ad indicare (e da cui si genera) la classe dei numeri interi.
  - *I* è definito ricorsivamente o come una cifra oppure come un intero seguito da una cifra.
  - Ogni intero relativo è generato da queste regole e niente che non sia un intero relativo può essere generato da queste regole.

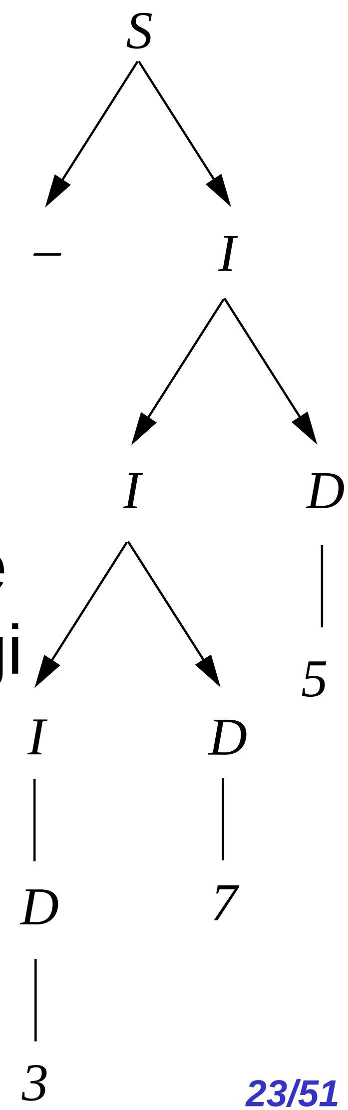
Riconoscimento di linguaggi formali: esempio

- Generazione ad albero:
  - proviamo a generare l'intero relativo -375
- Tale albero prende il nome di *albero di derivazione*.
- $S \overset{*}{\Rightarrow} -375 \quad \Leftrightarrow \quad -375 \in L$
- Nella notazione vista per il linguaggio delle parentesi ben formate, tipica per i linguaggi formali, la grammatica diventa:

$$S \rightarrow +I \mid -I$$

$$I \rightarrow D \mid I\,D$$

$$D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$



Grammatiche generative

- Dagli esempi di linguaggi visti, possiamo trarre le seguenti conclusioni. Per generare un linguaggio sono necessari:
  - un insieme *X* di simboli primitivi con cui si formano le parole del linguaggio, detto *alfabeto dei simboli terminali* o *alfabeto terminale*;
  - un insieme *V* di simboli ausiliari o variabili con cui si identificano le categorie sintattiche del linguaggio, detto *alfabeto dei simboli nonterminali (ausiliari)* o *alfabeto nonterminale* o *alfabeto delle variabili*;
  - un simbolo speciale *S*, scelto tra i nonterminali, da cui far partire la generazione delle parole del linguaggio. Tale simbolo è detto *assioma* o *scopo* o *simbolo distintivo* o *simbolo di partenza* o *simbolo iniziale*;
  - un insieme *P* di *produzioni*, espresse in un formalismo quali regole di riscrittura, BNF (*a* ::= *b*), carte sintattiche, …

Definizione di Grammatica generativa o a struttura di frase o a forma di frase

- Una *grammatica generativa* o *a struttura di frase G* è una quadrupla

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

ove:

  - *X* è l'*alfabeto terminale* per la grammatica;
  - *V* è l'*alfabeto nonterminale* o delle variabili per la grammatica;
  - *S* è il *simbolo di partenza* per la grammatica;
  - *P* è l'insieme delle *produzioni* della grammatica

  ed inoltre valgono le seguenti condizioni:

$$X \cap V = \emptyset \quad \text{e} \quad S \in V$$

Definizione di Produzione

Una *produzione* è una coppia (*v*,*w*)

ove $v \in (X \cup V)^{+}$ e *v* ha come sottostringa un *NT* $\Leftrightarrow v \in (X \cup V)^{*} V (X \cup V)^{*}$

$w \in (X \cup V)^{*}$ *(w può essere anche λ).*

Un elemento (v,w) di P viene comunemente scritto nella forma:

$$v \rightarrow w$$

Una produzione deve, in qualche modo, riscrivere un *NT*.

Definizione di Produzione

- Per convenzione, gli elementi di *X* sono rappresentati di solito con lettere minuscole (con o senza pedici e di solito sono le prime lettere dell'alfabeto) o cifre ed operatori (connettivi), mentre gli elementi di *V* sono rappresentati con lettere maiuscole (con o senza pedici) o con stringhe delimitate dalle parentesi angolari “<“ e “>“.
- La notazione $a \rightarrow b_1 \mid b_2 \mid \ldots \mid b_k$ è impiegata come abbreviazione della seguente:

$$a \rightarrow b_1$$
$$a \rightarrow b_2$$
$$\ldots$$
$$a \rightarrow b_k$$

Esempi di grammatiche

- La grammatica per il linguaggio delle parentesi ben formate

$$G_1 = \left(\{\, (\, ,\,)\, \},\quad \{S\},\quad S,\quad \{S \rightarrow (\ \ ),\ S \rightarrow (S),\ S \rightarrow SS\} \right)$$

- La grammatica per il linguaggio dei numeri interi relativi

$$G_2 = \left(\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, \text{-}\},\quad \{S, I, D\},\quad S,\right.$$
$$\left. \{S \rightarrow +I,\ S \rightarrow -I,\ I \rightarrow D,\ I \rightarrow ID,\ D \rightarrow 0,\ D \rightarrow 1,\ \ldots,\ D \rightarrow 9\} \right)$$

Definizione di derivazione o produzione diretta

- Sia G = (X, V, S, P) una grammatica e siano y e z due stringhe finite di simboli in $X \cup V$ (stringhe di terminali e nonterminali) tali che:

  $y = \gamma\alpha\delta$ e $z = \gamma\beta\delta$, ove $y \in (X \cup V)^+$, $z \in (X \cup V)^*$, $\alpha, \beta, \gamma, \delta \in (X \cup V)^*$

  1) Scriviamo

  $$y \Rightarrow z$$

  e diciamo che *y* *produce direttamente* *z* o che *z* è *derivata direttamente da y* se:

  $$\alpha \to \beta \in P$$

ossia se esiste in *G* una produzione $\alpha \to \beta$

Definizione di derivazione o produzione diretta

2) Scriviamo

$$y \overset{*}{\Rightarrow} z$$

e diciamo che *y* *produce* *z* o che *z* è *derivabile da y* se *y* = *z* o esiste una sequenza di stringhe $w_1, w_2, \ldots, w_n$, con $w_1, w_2, \ldots, w_{n-1} \in (X \cup V)^+, \quad w_n \in (X \cup V)^*$
$w_1 = y$ e $w_n = z$ tali che $\forall i,\ i = 1, 2, \ldots, n-1 : w_i \underset{G}{\Rightarrow} w_{i+1}$
(w_i produce direttamente w_{i+1}), cioè:

$$y \overset{*}{\Rightarrow} z \quad \Leftrightarrow \quad \begin{cases} y = z \\ \quad \text{oppure} \\ w_1 = y \Rightarrow w_2 \Rightarrow w_3 \Rightarrow \ldots \Rightarrow w_{n-1} \Rightarrow w_n = z \end{cases}$$

Osservazione

- La nozione di derivazione diretta stabilisce una relazione binaria in $(X \cup V)^*$. Date due stringhe *y* e *z*, il simbolo $\Rightarrow$ può esserci o meno; dipende dall'esistenza di una produzione.
- Allora possiamo anche definire una composizione di relazioni:

$$y \overset{2}{\Rightarrow} z \quad \overset{def}{\Leftrightarrow} \quad \exists\, w : y \Rightarrow w \text{ e } w \Rightarrow z$$

dove 2 è il numero di trascrizioni necessarie per passare da *y* a *z* (ossia, la *lunghezza della derivazione*).

Osservazione

- Da ciò si ha:

$$\overset{*}{\Rightarrow} \;=\; I \cup \Rightarrow \cup \overset{2}{\Rightarrow} \cup \overset{3}{\Rightarrow} \cup \ldots$$

ove *I* è la relazione identica e $\overset{n}{\Rightarrow}$ indica la composizione della relazione $\Rightarrow$ *n* volte con se stessa.

$\overset{*}{\Rightarrow}$ è la chiusura riflessiva e transitiva della relazione di derivazione diretta;

$\overset{+}{\Rightarrow}$ è la chiusura transitiva della stessa relazione.

Esempio

Definizione di linguaggio generato da una grammatica

- Sia $G = (X, V, S, P)$ una grammatica. Il *linguaggio generato da G*, denotato con $L(G)$, è l'insieme delle stringhe di terminali derivabili dal simbolo di partenza S.

$$L(G) = \left\{ w \in X^* \;\middle|\; S \overset{*}{\underset{G}{\Rightarrow}} w \right\}$$

- Sono, dunque, parole di $L(G)$ le stringhe che:
  - consistono di soli terminali;
  - possono essere derivate da S in G.

Definizione di forma di frase

- Sia $G = (X, V, S, P)$ una grammatica. Una stringa w, $w \in (X \cup V)^*$, è una *forma di frase* di G se

$$S \overset{*}{\underset{G}{\Rightarrow}} w$$

- Alle forme di frasi si applicano le stesse definizioni (es.: potenza) e gli stessi operatori (es.: concatenazione) dati per le stringhe.
- Proposizione:
  - Data una grammatica $G = (X, V, S, P)$, $L(G)$ è l'insieme delle forme di frase terminali (o *frasi*) di G.

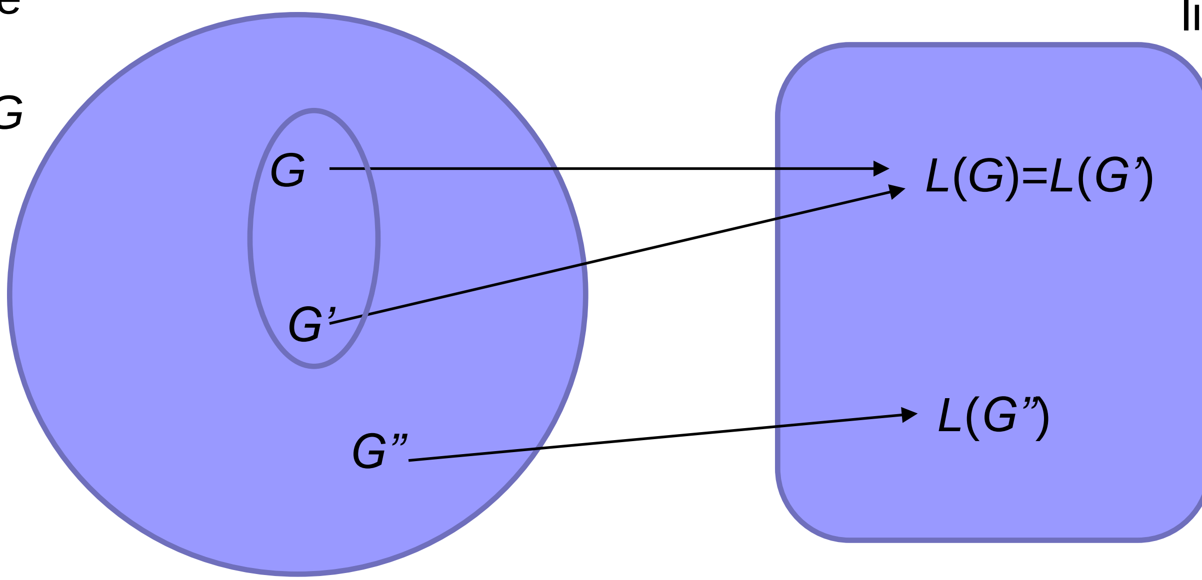
Definizione di grammatiche equivalenti

- Due grammatiche *G* e *G'* si dicono *equivalenti* se generano lo stesso linguaggio, ossia se

$$L(G)=L(G')$$

Insieme di tutti i possibili linguaggi *L*

Insieme di tutte le possibili grammatiche *G*



Esempio

- Sia $G = (X, V, S, P)$, ove

$$X = \{a, b\}, \qquad V = \{S\}, \qquad P = \left\{ S \overset{(1)}{\rightarrow} aSb, S \overset{(2)}{\rightarrow} ab \right\}$$

Determiniamo $L(G)$.

$ab \in L(G)$ poiché $S \underset{(2)}{\Rightarrow} ab$

Se numeriamo le produzioni, possiamo indicare la produzione usata immediatamente al di sotto del simbolo $\Rightarrow$.

$\underset{(n)}{\Longrightarrow}$ $\equiv$ ho applicato la produzione n

$y \overset{k}{\Rightarrow} z$ $\equiv$ y produce z in k passi, dove k=lunghezza della derivazione

Esempio

- $a^2b^2 \in L(G)$ poiché $S \underset{(1)}{\Rightarrow} aSb \underset{(2)}{\Rightarrow} a^2b^2$
- $a^3b^3 \in L(G)$ poiché $S \overset{3}{\Rightarrow} a^3b^3$

……

$$\{a^n b^n \mid n > 0\} \subseteq L(G)$$

- Inoltre, qualsiasi derivazione da S in G produce frasi terminali del tipo $a^n b^n$.
  - Dunque $L(G) \subseteq \{a^n b^n \mid n > 0\}$ e quindi

$$L(G) = \{a^n b^n \mid n > 0\}$$

Notazione

- Per rendere più concisa la descrizione di una grammatica, spesso ci limiteremo ad elencarne le produzioni, quando sia chiaro quale sia il simbolo di partenza e quali siano i terminali ed i nonterminali.
- Inoltre, le produzioni con la stessa parte sinistra vengono accorpate attraverso l’uso del simbolo “ | “ (preso a prestito dalla BNF).
- Infine, ometteremo l’indicazione della grammatica dalla simbologia di derivazione e derivazione diretta quando sia chiaro dal contesto a quale grammatica si fa riferimento.

Esempio

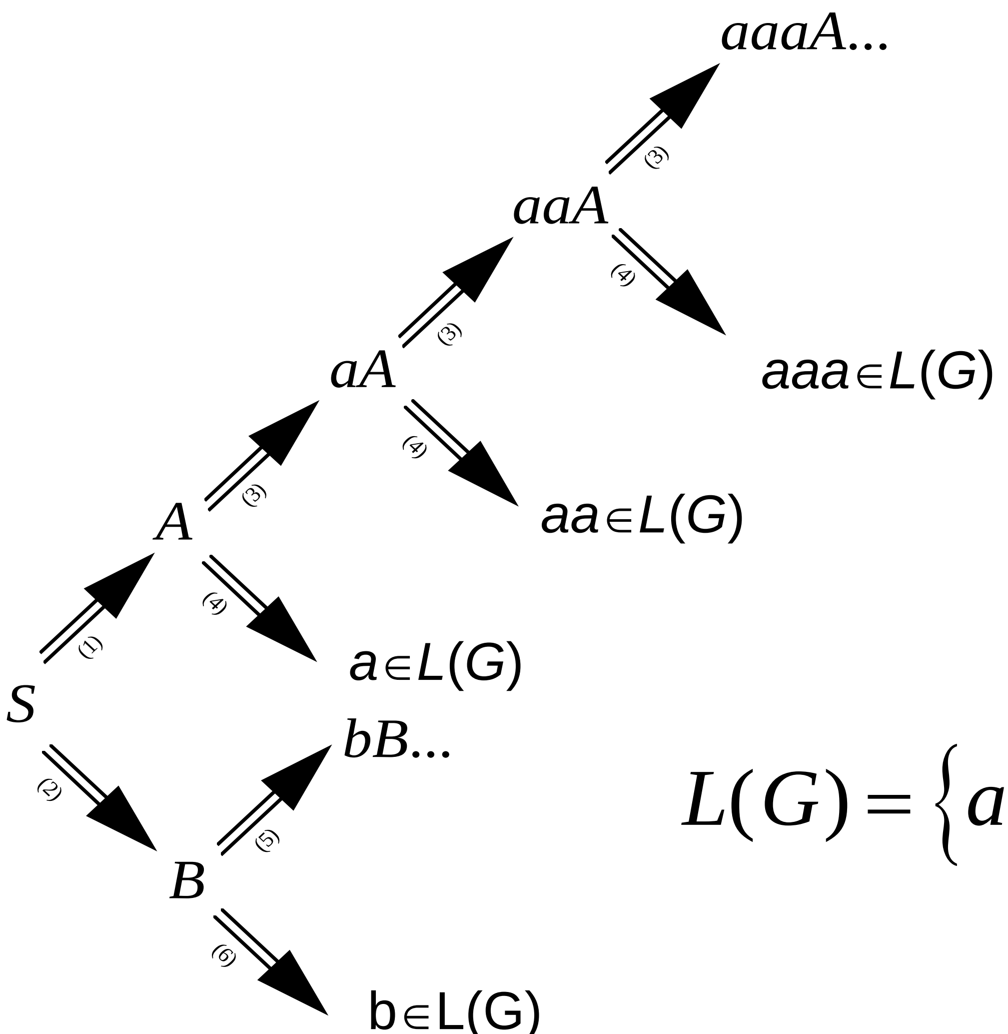
- Sia data la seguente grammatica:

$$S \xrightarrow{(1)} A \overset{(2)}{|} B, \quad A \xrightarrow{(3)} aA \overset{(4)}{|} a, \quad B \xrightarrow{(5)} bB \overset{(6)}{|} b$$

Determinare *L(G).*

Non sappiamo se applicare $S \xrightarrow{(1)} A$ oppure $S \xrightarrow{(2)} B$ inizialmente. I meccanismi di costruzione di un linguaggio sono generalmente non deterministici, poiché può non essere univoca la sostituzione da operare ad una forma di frase se uno stesso NT si trova a sinistra di 2 o più produzioni, come illustrato nella figura seguente.

Esempio



$$L(G)=\left\{a^n \middle| n>o\right\}\cup\left\{b^n \middle| n>0\right\}$$

Osservazione

- Dunque, una grammatica è uno *strumento generativo* di un linguaggio perché, data una qualsiasi parola di quel linguaggio, possiamo risalire mediante le produzioni al simbolo di partenza della grammatica.
- Viceversa, dato il simbolo di partenza di una grammatica, seguendo uno qualsiasi dei cammini dell'albero di derivazione, si produce una parola "valida" del linguaggio.

Osservazione

- In generale, dato un linguaggio *L* ed una grammatica *G*, <u>non esiste un algoritmo</u> in grado di dimostrare che la grammatica genera il linguaggio, ossia che $L = L(G)$.
  Più specificamente, <u>non</u> esiste un algoritmo che stabilisce se una data stringa è generata o no dalla grammatica presa in considerazione.
- Tutto ciò si riassume nella seguente proposizione:
  - Il problema di dimostrare la correttezza di una grammatica non è risolubile algoritmicamente, <u>in generale</u>.

Osservazione

- In molti casi importanti, però, è possibile dimostrare per induzione che una particolare grammatica genera proprio un particolare linguaggio.
- Queste dimostrazioni ci consentono di stabilire se, data una grammatica *G* ed un linguaggio *L*, risulta:
  - $w \in L(G) \Rightarrow w \in L$ cioè $L(G) \subseteq L$
  - $w \in L \Rightarrow w \in L(G)$ cioè $L \subseteq L(G)$

Osservazione

- In molti casi importanti, però, è possibile dimostrare per induzione che una particolare grammatica genera proprio un particolare linguaggio.
- Queste dimostrazioni ci consentono di stabilire se, data una grammatica *G* ed un linguaggio *L*, risulta:
  - $w \in L(G) \Rightarrow w \in L$ cioè $L(G) \subseteq L$
  - $w \in L \Rightarrow w \in L(G)$ cioè $L \subseteq L(G)$

La grammatica *G* genera solo stringhe appartenenti al linguaggio *L* (*coerenza* o *consistenza* di *G*).

Osservazione

- In molti casi importanti, però, è possibile dimostrare per induzione che una particolare grammatica genera proprio un particolare linguaggio.
- Queste dimostrazioni ci consentono di stabilire se, data una grammatica *G* ed un linguaggio *L*, risulta:
  - $w \in L(G) \;\Rightarrow\; w \in L$ cioè $L(G) \subseteq L$
  - $w \in L \;\Rightarrow\; w \in L(G)$ cioè $L \subseteq L(G)$

Il linguaggio *L* comprende solo parole generabili dalla grammatica *G* (*completezza* di *G*).

Principio di induzione (I forma)

Sia n_0 un intero e sia $P=P(n)$ un enunciato che ha senso per ogni intero n maggiore o uguale ad n_0. Se:

- $P(n_0)$ è vero
- per ogni $n>n_0$, $P(n-1)$ vero implica $P(n)$ vero

allora $P(n)$ è vero per tutti gli n maggiori o uguali ad n_0

Principio di induzione (II forma: Noetheriana)

Sia n_0 un intero e sia $P=P(n)$ un enunciato che ha senso per ogni intero maggiore o uguale ad n_0. Se:

- $P(n_0)$ è vero
- per ogni n ed m, con $n>m\geq n_0$, $P(m)$ vero implica $P(n)$ vero

allora $P(n)$ è vero per tutti gli n maggiori o uguali ad n_0

Esercizi

- Determinare una grammatica che genera il seguente linguaggio:

$$L = \{a^n b^n \mid n > 0\}$$

  e dimostrare questo risultato.

- Che tipo di grammatica genera *L* ?

Soluzione esercizio

Esercizi

- Determinare una grammatica che genera il seguente linguaggio:

$$L = \{a^n b^{2n} \mid n > 0\}$$

  e dimostrare questo risultato.

- Di che tipo è la grammatica che genera *L* ?

  Soluzione esercizio

Esercizi

- Sia data la seguente grammatica:

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

$$X = \{0, 1\} \qquad V = \{S, A, B\}$$

$$P = \left\{ S \to \overset{(1)}{0B} \mid \overset{(2)}{1A},\ A \to \overset{(3)}{0} \mid \overset{(4)}{0S} \mid \overset{(5)}{1AA},\ B \to \overset{(6)}{1} \mid \overset{(7)}{1S} \mid \overset{(8)}{0BB} \right\}$$

- Determinare il linguaggio generato da *G* e dimostrare il risultato.

Soluzione esercizio

Esercizi

- Dimostrare per induzione che il linguaggio *L* generato dalla seguente grammatica è vuoto:

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

$$X = \{a, b, c\} \qquad V = \{S, A, B\}$$

$$P = \left\{ S \rightarrow \overset{(1)}{aBS} \mid \overset{(2)}{bA},\ \ aB \rightarrow \overset{(3)}{Ac} \mid \overset{(4)}{a},\ \ bA \rightarrow \overset{(5)}{S} \mid \overset{(6)}{Ba} \right\}$$

Soluzione esercizio

Riferimenti

- Semeraro, G., Elementi di Teoria dei Linguaggi Formali, ilmiolibro.it, 2017 (http://ilmiolibro.kataweb.it/libro/informatica-e-internet/317883/elementi-di-teoria-dei-linguaggi-formali/).
    - Capitolo 2