STRUMENTI FORMALI PER LA BIOINFORMATICA

La gerarchia di Chomsky - Parte 1

Ringraziamenti

Le figure sono prese dai libri (o dispense):

P. Degano, *Fondamenti di Informatica: Calcolabilità e Complessità*, Dispense, 2019.

J. Hopcroft, R. Motwani, J. Ullman , *Automi, Linguaggi e Calcolabilità*, Addison Wesley Pearson Education Italia s.r.l, Terza Edizione, 2009.

Michael Sipser, *Introduzione alla teoria della Computazione*, Apogeo Education, Maggioli Editore, 2016 (traduzione italiana di Introduction to the Theory of Computation, 3rd Edition).

Peter Linz, *An Introduction to Formal Languages and Automata*, Jones and Bartlett Publishers, Inc, 2011, 5th Edition.

Definizione di grammatica (formale)

Introduciamo adesso una definizione di grammatica che è più generale di quella data in precedenza (grammatica context-free).

Definizione di grammatica (formale)

Introduciamo adesso una definizione di grammatica che è più generale di quella data in precedenza (grammatica context-free).

Questo concetto eredita molte caratteristiche dai sistemi di Thue, di Post, di riscrittura,... e fu introdotto da Noam Chomsky allo scopo di formalizzare le lingue naturali e facilitare la traduzione automatica.

Rivedremo le macchine di Turing e i linguaggi riconosciuti dalle macchine di Turing, i linguaggi Turing-riconoscibili.

Rivedremo le macchine di Turing e i linguaggi riconosciuti dalle macchine di Turing, i linguaggi Turing-riconoscibili.

Le grammatiche, nella loro forma più generale, sono Turing-equivalenti, nel senso che gli insiemi di stringhe che generano sono Turing-riconoscibili.

Definizione di grammatica (formale)

Rivedremo le macchine di Turing e i linguaggi riconosciuti dalle macchine di Turing, i linguaggi Turing-riconoscibili.

Le grammatiche, nella loro forma più generale, sono Turing-equivalenti, nel senso che gli insiemi di stringhe che generano sono Turing-riconoscibili.

Poiché gli informatici sono interessati a usare le grammatiche per descrivere quelli che sono i programmi "ben formati", ci siamo soffermati e ci soffermeremo ancora sulle grammatiche context-free, in modo da garantirci sia la decidibilità che l'efficienza.

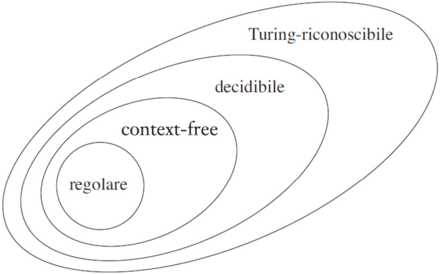

Figura: Una gerarchia di linguaggi

Definizione di grammatica (formale)

Definizione

Una grammatica (di tipo 0 o a struttura di frase) G, è una quadrupla $G = (V, T, P, S)$ dove:

Definizione di grammatica (formale)

Definizione

Una grammatica (di tipo 0 o a struttura di frase) G, è una quadrupla $G = (V, T, P, S)$ dove:

- V*:* **Insieme finito di variabili** *(dette anche non terminali o categorie sintattiche).*

Definizione

Una grammatica (di tipo 0 o a struttura di frase) G, è una quadrupla $G = (V, T, P, S)$ dove:

- V*:* **Insieme finito di variabili** *(dette anche non terminali o categorie sintattiche).*
- T*:* **Insieme finito di simboli terminali** *(o alfabeto dei terminali).* $T \cap V = \emptyset$.

Definizione di grammatica (formale)

Definizione

Una grammatica (di tipo 0 o a struttura di frase) G, è una quadrupla $G = (V, T, P, S)$ dove:

- V: **Insieme finito di variabili** *(dette anche non terminali o categorie sintattiche).*
- T: **Insieme finito di simboli terminali** *(o alfabeto dei terminali).* $T \cap V = \emptyset$.
- P: **Insieme finito delle produzioni**. *Ogni produzione ha la forma* $\alpha \rightarrow \beta$, *dove* $\alpha \in (V \cup T)^+$ *e* $\beta \in (V \cup T)^*$.

Definizione

Una grammatica (di tipo 0 o a struttura di frase) G, è una quadrupla $G = (V, T, P, S)$ dove:

- V*:* **Insieme finito di variabili** *(dette anche non terminali o categorie sintattiche).*
- T*:* **Insieme finito di simboli terminali** *(o alfabeto dei terminali).* $T \cap V = \emptyset$.
- P*:* **Insieme finito delle produzioni***. Ogni produzione ha la forma* $\alpha \rightarrow \beta$*, dove* $\alpha \in (V \cup T)^+$ *e* $\beta \in (V \cup T)^*$.
- S*: Una variabile, detta* **simbolo iniziale** *(o start symbol).*

Definizione di grammatica (formale)

La differenza consiste nella definizione di produzione.

Le definizioni di derivazione diretta, derivazione e linguaggio generato sono le stesse.

Definizione

Sia $G = (V, T, P, S)$ *una grammatica.*

Definizione

Sia $G = (V, T, P, S)$ *una grammatica.*

Sia $\alpha \rightarrow \beta \in P$, $\gamma_1, \gamma_2 \in (V \cup T)^*$.

Definizione

Sia $G = (V, T, P, S)$ *una grammatica.*

Sia $\alpha \to \beta \in P$, $\gamma_1, \gamma_2 \in (V \cup T)^*$.

Allora diremo che

$\gamma_1 \alpha \gamma_2$ *deriva (direttamente)* $\gamma_1 \beta \gamma_2$ *in* G

e scriveremo

$$\gamma_1 \alpha \gamma_2 \underset{G}{\Rightarrow} \gamma_1 \beta \gamma_2$$

Definizione

Sia $G = (V, T, P, S)$ *una grammatica.*

Sia $\alpha \to \beta \in P$, $\gamma_1, \gamma_2 \in (V \cup T)^*$.

Allora diremo che

$\gamma_1\alpha\gamma_2$ *deriva (direttamente)* $\gamma_1\beta\gamma_2$ *in* G

e scriveremo

$$\gamma_1\alpha\gamma_2 \underset{G}{\Rightarrow} \gamma_1\beta\gamma_2$$

o, se G *è chiara dal contesto, semplicemente*

$$\gamma_1\alpha\gamma_2 \Rightarrow \gamma_1\beta\gamma_2$$

Sia $G = (V, T, P, S)$ una grammatica, siano $\alpha, \beta \in (V \cup T)^*$. Diremo che α deriva β in G, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, se esistono parole $\gamma_1, \ldots, \gamma_n$, $n \geq 1$, tali che

Sia $G = (V, T, P, S)$ una grammatica, siano $\alpha, \beta \in (V \cup T)^*$. Diremo che α deriva β in G, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, se esistono parole $\gamma_1, \ldots, \gamma_n$, $n \geq 1$, tali che

1. $\alpha = \gamma_1$,

Sia $G = (V, T, P, S)$ una grammatica, siano $\alpha, \beta \in (V \cup T)^*$. Diremo che α **deriva** β in G, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, se esistono parole $\gamma_1, \ldots, \gamma_n$, $n \geq 1$, tali che

1. $\alpha = \gamma_1$,
2. $\beta = \gamma_n$,

Sia $G = (V, T, P, S)$ una grammatica, siano $\alpha, \beta \in (V \cup T)^*$. Diremo che α deriva β in G, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, se esistono parole $\gamma_1, \ldots, \gamma_n$, $n \geq 1$, tali che

1. $\alpha = \gamma_1$,
2. $\beta = \gamma_n$,
3. Per $i = 1, 2, \ldots, n-1$, si ha $\gamma_i \underset{G}{\Rightarrow} \gamma_{i+1}$

Sia $G = (V, T, P, S)$ una grammatica, siano $\alpha, \beta \in (V \cup T)^*$. Diremo che α deriva β in G, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, se esistono parole $\gamma_1, \dots, \gamma_n$, $n \geq 1$, tali che

1. $\alpha = \gamma_1$,
2. $\beta = \gamma_n$,
3. Per $i = 1, 2, \dots, n-1$, si ha $\gamma_i \underset{G}{\Rightarrow} \gamma_{i+1}$

In questo caso diremo anche che α deriva β in $n - 1$ passi.

Sia $G = (V, T, P, S)$ una grammatica, siano $\alpha, \beta \in (V \cup T)^*$. Diremo che α deriva β in G, $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, se esistono parole $\gamma_1, \ldots, \gamma_n$, $n \geq 1$, tali che

1. $\alpha = \gamma_1$,
2. $\beta = \gamma_n$,
3. Per $i = 1, 2, \ldots, n-1$, si ha $\gamma_i \underset{G}{\Rightarrow} \gamma_{i+1}$

In questo caso diremo anche che α deriva β in $n-1$ passi.

Ovviamente G è omessa se è chiara dal contesto.

Definizione

Sia $G = (V, T, P, S)$ una grammatica.

Il **linguaggio di** G *è*

$$L(G) = \{w \in T^* \mid S \overset{*}{\underset{G}{\Rightarrow}} w\}$$

Nota: Le parole di $L(G)$ sono tutte e sole le stringhe di caratteri terminali per le quali esiste una derivazione dal simbolo iniziale.

Esempio

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$
e P consiste nelle seguenti produzioni

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$
e P consiste nelle seguenti produzioni

1. $S \rightarrow aAb$

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$
e P consiste nelle seguenti produzioni

1. $S \to aAb$
2. $aA \to aaAb$

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$
e P consiste nelle seguenti produzioni

1. $S \to aAb$
2. $aA \to aaAb$
3. $A \to \epsilon$

Esempio

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$
e P consiste nelle seguenti produzioni

1. $S \rightarrow aAb$
2. $aA \rightarrow aaAb$
3. $A \rightarrow \epsilon$

$$L(G) = \{a^n b^n \mid n \geq 1\}$$

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$ e
$P = \{S \to aAb,\ aA \to aaAb,\ A \to \epsilon\}$

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$ e
$P = \{S \to aAb,\ aA \to aaAb,\ A \to \epsilon\}$

$$L(G) = \{a^n b^n \mid n \geq 1\}$$

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$ e
$P = \{S \to aAb,\ aA \to aaAb,\ A \to \epsilon\}$

$$L(G) = \{a^n b^n \mid n \geq 1\}$$

Come generare $a^n b^n$?
Basta applicare la produzione $S \to aAb$, poi $n - 1$ volte la produzione $aA \to aaAb$ e infine la produzione $A \to \epsilon$.

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$ e
$P = \{S \to aAb,\ aA \to aaAb,\ A \to \epsilon\}$

$$L(G) = \{a^n b^n \mid n \geq 1\}$$

Come generare $a^n b^n$?
Basta applicare la produzione $S \to aAb$, poi $n - 1$ volte la produzione $aA \to aaAb$ e infine la produzione $A \to \epsilon$.

$$S \Rightarrow aAb \Rightarrow aaAbb \stackrel{*}{\Rightarrow} a^n A b^n \Rightarrow a^n b^n$$

Esempio

$G = (V, T, P, S)$, dove $V = \{S, A\}$, $T = \{a, b\}$ e
$P = \{S \to aAb,\ aA \to aaAb,\ A \to \epsilon\}$

$$L(G) = \{a^n b^n \mid n \geq 1\}$$

Come generare $a^n b^n$?
Basta applicare la produzione $S \to aAb$, poi $n - 1$ volte la produzione $aA \to aaAb$ e infine la produzione $A \to \epsilon$.

$$S \Rightarrow aAb \Rightarrow aaAbb \stackrel{*}{\Rightarrow} a^n Ab^n \Rightarrow a^n b^n$$

Occorrerebbe anche mostrare che se $w \in T^*$ e $S \stackrel{*}{\Rightarrow} w$ allora $w = a^n b^n$.

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$

Esempio

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \rightarrow aSBC|aBC$

Esempio

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \to aSBC|aBC$
2. $CB \to BC$

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \to aSBC|aBC$
2. $CB \to BC$
3. $aB \to ab$

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \rightarrow aSBC|aBC$
2. $CB \rightarrow BC$
3. $aB \rightarrow ab$
4. $bB \rightarrow bb$

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \rightarrow aSBC|aBC$
2. $CB \rightarrow BC$
3. $aB \rightarrow ab$
4. $bB \rightarrow bb$
5. $bC \rightarrow bc$

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \to aSBC|aBC$
2. $CB \to BC$
3. $aB \to ab$
4. $bB \to bb$
5. $bC \to bc$
6. $cC \to cc$

$G = (V, T, P, S)$, dove $V = \{S, B, C\}$, $T = \{a, b, c\}$
P consiste nelle seguenti produzioni

1. $S \rightarrow aSBC|aBC$
2. $CB \rightarrow BC$
3. $aB \rightarrow ab$
4. $bB \rightarrow bb$
5. $bC \rightarrow bc$
6. $cC \rightarrow cc$

$$L(G) = \{a^n b^n c^n \mid n \geq 1\}$$

Esempio

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Esempio. La derivazione di $a^3b^3c^3$.

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Esempio. La derivazione di $a^3b^3c^3$.

Utilizzando le produzioni $S \to aSBC|aBC$ otteniamo:

$$S \Rightarrow aSBC \Rightarrow aaSBCBC \Rightarrow aaaBCBCBC$$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Esempio. La derivazione di $a^3b^3c^3$.

Utilizzando le produzioni $S \to aSBC|aBC$ otteniamo:

$$S \Rightarrow aSBC \Rightarrow aaSBCBC \Rightarrow aaaBCBCBC$$

Utilizzando poi la produzione $CB \to BC$ otteniamo:

$$S \overset{*}{\Rightarrow} aaaBCBCBC \Rightarrow aaaBBCCBC \Rightarrow aaaBBCBCC$$

$$\Rightarrow aaaBBBCCC$$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Esempio. La derivazione di $a^3b^3c^3$.

Utilizzando le produzioni $S \to aSBC|aBC$ otteniamo:

$$S \Rightarrow aSBC \Rightarrow aaSBCBC \Rightarrow aaaBCBCBC$$

Utilizzando poi la produzione $CB \to BC$ otteniamo:

$$S \overset{*}{\Rightarrow} aaaBCBCBC \Rightarrow aaaBBCCBC \Rightarrow aaaBBCBCC$$

$$\Rightarrow aaaBBBCCC$$

Ora dobbiamo solo cambiare tutte le B con b e le C con c.

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Esempio

$P = \{S \rightarrow aSBC|aBC,\ CB \rightarrow BC,\ aB \rightarrow ab,\ bB \rightarrow bb,$
$bC \rightarrow bc,\ cC \rightarrow cc\}$

Utilizzando le produzioni $aB \rightarrow ab$ e $bB \rightarrow bb$ cambiamo tutte le B con b:

$$S \stackrel{*}{\Rightarrow} aaaBBBCCC \Rightarrow aaabBBCCC \Rightarrow aaabbBCCC$$

$$\Rightarrow aaabbbCCC$$

Esempio

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,\ bC \to bc,\ cC \to cc\}$

Utilizzando le produzioni $aB \to ab$ e $bB \to bb$ cambiamo tutte le B con b:

$$S \overset{*}{\Rightarrow} aaaBBBCCC \Rightarrow aaabBBCCC \Rightarrow aaabbBCCC$$

$$\Rightarrow aaabbbCCC$$

Infine, utilizzando le produzioni $bC \to bc$ e $cC \to cc$ cambiamo tutte le C con c:

$$S \overset{*}{\Rightarrow} aaabbbCCC \Rightarrow aaabbbcCC \Rightarrow aaabbbccC \Rightarrow aaabbbccc$$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Come generare $a^n b^n c^n$?
Iniziamo applicando $n-1$ volte la produzione $S \to aSBC$ e poi la produzione $S \to aBC$:

$$S \stackrel{*}{\Rightarrow} a^{n-1}S(BC)^{n-1} \Rightarrow a^n(BC)^n$$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Come generare $a^n b^n c^n$?
Iniziamo applicando $n-1$ volte la produzione $S \to aSBC$ e poi la produzione $S \to aBC$:

$$S \stackrel{*}{\Rightarrow} a^{n-1}S(BC)^{n-1} \Rightarrow a^n(BC)^n$$

Poi usiamo la produzione $CB \to BC$ per spostare le B e le C e fare in modo che tutte le B precedano tutte le C:

$$S \stackrel{*}{\Rightarrow} a^n(BC)^n \stackrel{*}{\Rightarrow} a^n B^n C^n$$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

Come generare $a^n b^n c^n$?

Iniziamo applicando $n-1$ volte la produzione $S \to aSBC$ e poi la produzione $S \to aBC$:

$$S \stackrel{*}{\Rightarrow} a^{n-1}S(BC)^{n-1} \Rightarrow a^n(BC)^n$$

Poi usiamo la produzione $CB \to BC$ per spostare le B e le C e fare in modo che tutte le B precedano tutte le C:

$$S \stackrel{*}{\Rightarrow} a^n(BC)^n \stackrel{*}{\Rightarrow} a^n B^n C^n$$

Ora dobbiamo solo cambiare tutte le B con b e le C con c.

Esempio

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$$S \stackrel{*}{\Rightarrow} a^n(BC)^n \stackrel{*}{\Rightarrow} a^nB^nC^n$$

$P = \{S \to aSBC | aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$$S \stackrel{*}{\Rightarrow} a^n(BC)^n \stackrel{*}{\Rightarrow} a^nB^nC^n$$

Usiamo una volta la produzione $aB \to ab$ e $n-1$ volte la produzione $bB \to bb$ per cambiare tutte le B con b:

$$S \stackrel{*}{\Rightarrow} a^nB^nC^n \Rightarrow a^nbB^{n-1}C^n \stackrel{*}{\Rightarrow} a^nb^nC^n$$

$P = \{S \to aSBC|aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$$S \stackrel{*}{\Rightarrow} a^n(BC)^n \stackrel{*}{\Rightarrow} a^nB^nC^n$$

Usiamo una volta la produzione $aB \to ab$ e $n-1$ volte la produzione $bB \to bb$ per cambiare tutte le B con b:

$$S \stackrel{*}{\Rightarrow} a^nB^nC^n \Rightarrow a^nbB^{n-1}C^n \stackrel{*}{\Rightarrow} a^nb^nC^n$$

Infine usiamo una volta la produzione $bC \to bc$ e $n-1$ volte la produzione $cC \to cc$ per cambiare tutte le C con c:

$$S \stackrel{*}{\Rightarrow} a^nb^nC^n \Rightarrow a^nb^ncC^{n-1} \stackrel{*}{\Rightarrow} a^nb^nc^n$$

Esempio

$P = \{S \to aSBC | aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$

$$S \stackrel{*}{\Rightarrow} a^n(BC)^n \stackrel{*}{\Rightarrow} a^n B^n C^n$$

Usiamo una volta la produzione $aB \to ab$ e $n-1$ volte la produzione $bB \to bb$ per cambiare tutte le B con b:

$$S \stackrel{*}{\Rightarrow} a^n B^n C^n \Rightarrow a^n b B^{n-1} C^n \stackrel{*}{\Rightarrow} a^n b^n C^n$$

Infine usiamo una volta la produzione $bC \to bc$ e $n-1$ volte la produzione $cC \to cc$ per cambiare tutte le C con c:

$$S \stackrel{*}{\Rightarrow} a^n b^n C^n \Rightarrow a^n b^n c C^{n-1} \stackrel{*}{\Rightarrow} a^n b^n c^n$$

Occorrerebbe anche mostrare che se $w \in T^*$ e $S \stackrel{*}{\Rightarrow} w$ allora $w = a^n b^n c^n$.

Un esempio di grammatica G' più semplice che genera lo stesso linguaggio è $G' = (V', T, P', S)$, dove

Un esempio di grammatica G' più semplice che genera lo stesso linguaggio è $G' = (V', T, P', S)$, dove $V' = \{S, B\}$, $T = \{a, b, c\}$ e P' consiste nelle seguenti produzioni

Un esempio di grammatica G' più semplice che genera lo stesso linguaggio è $G' = (V', T, P', S)$, dove $V' = \{S, B\}$, $T = \{a, b, c\}$ e P' consiste nelle seguenti produzioni

1. $S \to aSBc \mid abc$

Un esempio di grammatica G' più semplice che genera lo stesso linguaggio è $G' = (V', T, P', S)$, dove $V' = \{S, B\}$, $T = \{a, b, c\}$ e P' consiste nelle seguenti produzioni

1. $S \to aSBc \mid abc$
2. $cB \to Bc$

Un esempio di grammatica G' più semplice che genera lo stesso linguaggio è $G' = (V', T, P', S)$, dove $V' = \{S, B\}$, $T = \{a, b, c\}$ e P' consiste nelle seguenti produzioni

1. $S \to aSBc \mid abc$
2. $cB \to Bc$
3. $bB \to bb$

Un esempio di grammatica G' più semplice che genera lo stesso linguaggio è $G' = (V', T, P', S)$, dove $V' = \{S, B\}$, $T = \{a, b, c\}$ e P' consiste nelle seguenti produzioni

1. $S \to aSBc \mid abc$
2. $cB \to Bc$
3. $bB \to bb$

$$L(G') = \{a^n b^n c^n \mid n \geq 1\}$$

Le produzioni di una grammatica di tipo 0 non hanno alcun vincolo, se non quello di avere almeno un simbolo nella loro parte sinistra.

Le produzioni di una grammatica di tipo 0 non hanno alcun vincolo, se non quello di avere almeno un simbolo nella loro parte sinistra.

Adesso porremo dei vincoli sulle produzioni, e vedremo che quanto più questi vincoli sono stringenti, tanto minore è il potere espressivo delle grammatiche in questione.

Le produzioni di una grammatica di tipo 0 non hanno alcun vincolo, se non quello di avere almeno un simbolo nella loro parte sinistra.

Adesso porremo dei vincoli sulle produzioni, e vedremo che quanto più questi vincoli sono stringenti, tanto minore è il potere espressivo delle grammatiche in questione.

Cioè i vincoli riducono la capacità computazionale del modello.

Le produzioni di una grammatica di tipo 0 non hanno alcun vincolo, se non quello di avere almeno un simbolo nella loro parte sinistra.

Adesso porremo dei vincoli sulle produzioni, e vedremo che quanto più questi vincoli sono stringenti, tanto minore è il potere espressivo delle grammatiche in questione.

Cioè i vincoli riducono la capacità computazionale del modello.

Otterremo quindi una gerarchia di grammatiche al variare dei vincoli e in corrispondenza una gerarchia di linguaggi.

Le produzioni di una grammatica di tipo 0 non hanno alcun vincolo, se non quello di avere almeno un simbolo nella loro parte sinistra.

Adesso porremo dei vincoli sulle produzioni, e vedremo che quanto più questi vincoli sono stringenti, tanto minore è il potere espressivo delle grammatiche in questione.

Cioè i vincoli riducono la capacità computazionale del modello.

Otterremo quindi una gerarchia di grammatiche al variare dei vincoli e in corrispondenza una gerarchia di linguaggi.

Vedremo poi che a tale gerarchia corrisponde una gerarchia di automi.

Le produzioni di una grammatica di tipo 0 non hanno alcun vincolo, se non quello di avere almeno un simbolo nella loro parte sinistra.

Adesso porremo dei vincoli sulle produzioni, e vedremo che quanto più questi vincoli sono stringenti, tanto minore è il potere espressivo delle grammatiche in questione.

Cioè i vincoli riducono la capacità computazionale del modello.

Otterremo quindi una gerarchia di grammatiche al variare dei vincoli e in corrispondenza una gerarchia di linguaggi.

Vedremo poi che a tale gerarchia corrisponde una gerarchia di automi.

In base alla forma delle produzioni, classificheremo le grammatiche in quattro tipi, indicati brevemente come

tipo 0, 1, 2 e 3

Definizione

Una grammatica $G = (V, T, P, S)$ *è una grammatica di tipo 0 se ogni* $\alpha \to \beta \in P$ *è tale che* $\alpha \in (V \cup T)^+$ *e* $\beta \in (V \cup T)^*$*.*

Definizione

Una grammatica $G = (V, T, P, S)$ è una grammatica di tipo 1 o dipendente dal contesto se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in (V \cup T)^+$, $\beta \in (V \cup T)^+$ e il lato destro di ogni produzione ha lunghezza almeno pari al lato sinistro

$$\forall \alpha \to \beta \in P \quad |\beta| \geq |\alpha|$$

Il nome di "grammatica dipendente dal contesto" deriva dal fatto che per ogni G di tipo 1, esiste G' tale che $L(G) = L(G')$ e in cui le produzioni hanno la forma

$$\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2$$

dove A è una variabile, $\alpha_1, \alpha_2, \beta$ sono stringhe di variabili e terminali e $\beta \neq \epsilon$. (Quindi anche G' è di tipo 1).

Il nome di "grammatica dipendente dal contesto" deriva dal fatto che per ogni G di tipo 1, esiste G' tale che $L(G) = L(G')$ e in cui le produzioni hanno la forma

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$$

dove A è una variabile, $\alpha_1, \alpha_2, \beta$ sono stringhe di variabili e terminali e $\beta \neq \epsilon$. (Quindi anche G' è di tipo 1).

Quindi è possibile sostituire A con β ogni volta che A appare nella stringa in un contesto particolare, cioè tra α_1 e α_2.

Il nome di "grammatica dipendente dal contesto" deriva dal fatto che per ogni G di tipo 1, esiste G' tale che $L(G) = L(G')$ e in cui le produzioni hanno la forma

$$\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2$$

dove A è una variabile, $\alpha_1, \alpha_2, \beta$ sono stringhe di variabili e terminali e $\beta \neq \epsilon$. (Quindi anche G' è di tipo 1).

Quindi è possibile sostituire A con β ogni volta che A appare nella stringa in un contesto particolare, cioè tra α_1 e α_2.

Alcuni autori adottano come definizione quella in cui le produzioni hanno questo secondo vincolo. Ma vedremo che la prima formulazione mette maggiormente in evidenza alcuni importanti aspetti.

Definizione

Una grammatica $G = (V, T, P, S)$ *è una grammatica di tipo 2 se ogni* $\alpha \to \beta \in P$ *è tale che* $\alpha \in V$ *e* $\beta \in (V \cup T)^+$ *cioè il lato sinistro di ogni produzione* $A \to \beta$ *è una variabile* A *e il lato destro* β *è una stringa non vuota di variabili e terminali.*

Definizione

Una grammatica $G = (V, T, P, S)$ è una grammatica di tipo 2 se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in V$ e $\beta \in (V \cup T)^+$ cioè il lato sinistro di ogni produzione $A \to \beta$ è una variabile A e il lato destro β è una stringa non vuota di variabili e terminali.

Possiamo dire che questa definizione coincide con quella data di grammatica context-free?

Definizione

Una grammatica $G = (V, T, P, S)$ *è una grammatica di tipo 3 o regolare se ogni* $\alpha \to \beta \in P$ *è tale che* $\alpha \in V$ *e* $\beta \in T \cup (T \cdot V)$*, cioè se ogni produzione è della forma* $A \to aB$ *o* $A \to a$*, dove* A, B *sono variabili e* a *è un terminale.*

Nota. Alcuni autori distinguono le grammatiche $G = (V, T, P, S)$ di tipo tre in

Nota. Alcuni autori distinguono le grammatiche $G = (V, T, P, S)$ di tipo tre in

- grammatica lineare destra se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in V$ e $\beta \in T \cup (T \cdot V)$, cioè se ogni produzione è della forma $A \to aB$ o $A \to a$, dove A, B sono variabili e a è un terminale.

Nota. Alcuni autori distinguono le grammatiche $G = (V, T, P, S)$ di tipo tre in

- grammatica lineare destra se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in V$ e $\beta \in T \cup (T \cdot V)$, cioè se ogni produzione è della forma $A \to aB$ o $A \to a$, dove A, B sono variabili e a è un terminale.
- grammatica lineare sinistra se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in V$ e $\beta \in T \cup (V \cdot T)$, cioè se ogni produzione è della forma $A \to Ba$ o $A \to a$, dove A, B sono variabili e a è un terminale.

Nota. Alcuni autori distinguono le grammatiche $G = (V, T, P, S)$ di tipo tre in

- grammatica lineare destra se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in V$ e $\beta \in T \cup (T \cdot V)$, cioè se ogni produzione è della forma $A \to aB$ o $A \to a$, dove A, B sono variabili e a è un terminale.
- grammatica lineare sinistra se ogni $\alpha \to \beta \in P$ è tale che $\alpha \in V$ e $\beta \in T \cup (V \cdot T)$, cioè se ogni produzione è della forma $A \to Ba$ o $A \to a$, dove A, B sono variabili e a è un terminale.

Non faremo questa distinzione ma si può intuire che per ogni grammatica lineare destra (risp. sinistra) G esiste una grammatica G' lineare sinistra (risp. destra) tale che $L(G) = L(G')$.

Notiamo che per $i = 1, 2, 3$, le grammatiche di tipo i sono un sottoinsieme delle grammatiche di tipo $i - 1$.

Notiamo che per $i = 1, 2, 3$, le grammatiche di tipo i sono un sottoinsieme delle grammatiche di tipo $i - 1$.

- Se G è una grammatica di tipo 3, allora G è anche di tipo 2

Notiamo che per $i = 1, 2, 3$, le grammatiche di tipo i sono un sottoinsieme delle grammatiche di tipo $i - 1$.

- Se G è una grammatica di tipo 3, allora G è anche di tipo 2

- Se G è una grammatica di tipo 2, allora G è anche di tipo 1

Notiamo che per $i = 1, 2, 3$, le grammatiche di tipo i sono un sottoinsieme delle grammatiche di tipo $i - 1$.

- Se G è una grammatica di tipo 3, allora G è anche di tipo 2

- Se G è una grammatica di tipo 2, allora G è anche di tipo 1

- Se G è una grammatica di tipo 1, allora G è anche di tipo 0

Più formalmente possiamo enunciare la seguente proprietà, facilmente dimostrabile.

Teorema (Gerarchia di grammatiche)

La classe delle grammatiche di tipo i include strettamente quella delle grammatiche di tipo $i+1$, $0 \leq i \leq 2$.

- La grammatica definita dalle produzioni

$$\begin{array}{rcl} S & \to & 0AB \qquad B \to 01 \qquad A1 \to SB1 \\ 1B & \to & 0 \qquad B \to SA \qquad A0 \to S0B \end{array}$$

è di tipo 0 ma non di tipo 1.

- La grammatica definita dalle produzioni

$$\begin{array}{rclccc} S & \to & 0AB & B \to 01 & A1 \to SB1 \\ 1B & \to & 0 & B \to SA & A0 \to S0B \end{array}$$

  è di tipo 0 ma non di tipo 1.

- La grammatica definita dalle produzioni

$$\begin{aligned} & S \to aSBc \mid abc \\ & cB \to Bc \\ & bB \to bb \end{aligned}$$

  è di tipo 1 ma non di tipo 2.

- La grammatica definita dalle produzioni

$$\begin{array}{rclcc} S & \rightarrow & 0AB & B \rightarrow 01 & A1 \rightarrow SB1 \\ 1B & \rightarrow & 0 & B \rightarrow SA & A0 \rightarrow S0B \end{array}$$

  è di tipo 0 ma non di tipo 1.
- La grammatica definita dalle produzioni

$$\begin{aligned} &S \rightarrow aSBc \mid abc \\ &cB \rightarrow Bc \\ &bB \rightarrow bb \end{aligned}$$

  è di tipo 1 ma non di tipo 2.
- La grammatica definita dalle produzioni

$$S \rightarrow aSb \mid ab$$

  è di tipo 2 ma non di tipo 3.

Il problema della parola vuota

Torniamo alla domanda precedente.

Torniamo alla domanda precedente.

Definizione

Una grammatica $G = (V, T, P, S)$ *è una grammatica di tipo 2 se ogni* $\alpha \to \beta \in P$ *è tale che* $\alpha \in V$ *e* $\beta \in (V \cup T)^+$ *cioè il lato sinistro di ogni produzione* $A \to \beta$ *è una variabile* A *e il lato destro* β *è una stringa non vuota di variabili e terminali.*

Torniamo alla domanda precedente.

Definizione

Una grammatica $G = (V, T, P, S)$ *è una grammatica di tipo 2 se ogni* $\alpha \to \beta \in P$ *è tale che* $\alpha \in V$ *e* $\beta \in (V \cup T)^+$ *cioè il lato sinistro di ogni produzione* $A \to \beta$ *è una variabile* A *e il lato destro* β *è una stringa non vuota di variabili e terminali.*

Possiamo dire che questa definizione coincide con quella data di grammatica context-free?

- Guardando le definizioni ci accorgiamo che la definizione di grammatica di tipo 2 non corrisponde alla definizione di grammatica context-free che abbiamo dato.

- Guardando le definizioni ci accorgiamo che la definizione di grammatica di tipo 2 non corrisponde alla definizione di grammatica context-free che abbiamo dato.
- In generale, le grammatiche di tipo i, $i \geq 1$, non possono generare la parola vuota.

- Guardando le definizioni ci accorgiamo che la definizione di grammatica di tipo 2 non corrisponde alla definizione di grammatica context-free che abbiamo dato.
- In generale, le grammatiche di tipo i, $i \geq 1$, non possono generare la parola vuota.

  Questo non è un problema.

Esiste una soluzione “ad hoc” per le grammatiche di tipo 2 e una “più elegante” applicabile a tutti e tre i tipi di grammatiche.

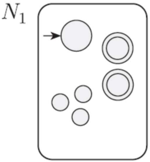

Figura: Un automa finito che riconosce un linguaggio L

La parola vuota

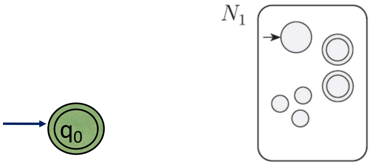

Figura: Un automa finito che riconosce L e un automa che riconosce $\{\epsilon\}$

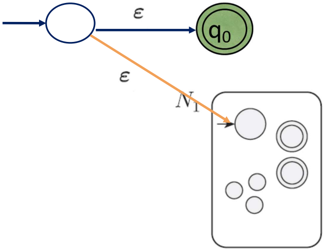

Figura: Un automa finito che riconosce $L \cup \{\epsilon\}$

Definizione

Due grammatiche G, G' sono equivalenti se generano lo stesso linguaggio, cioè $L(G) = L(G')$.

Definizione

Due grammatiche G, G' sono equivalenti se generano lo stesso linguaggio, cioè $L(G) = L(G')$.

- (Risultato preliminare)
  Data G di tipo i, $i = 1, 2, 3$, possiamo sempre trovare G' dello stesso tipo ed equivalente a G tale che il simbolo iniziale di G' non appaia alla destra di nessuna produzione.

Proposizione

Sia $G = (V, T, P, S)$ *una grammatica di tipo* i, $i = 1, 2, 3$.
Sia $G' = (V', T, P', S')$, *con* $S' \notin V$, $V' = V \cup \{S'\}$ *e dove* P' *è l'insieme delle produzioni in* P *più tutte le produzioni* $S' \to \alpha$, *dove* $S \to \alpha \in P$, *cioè*

$$P' = P \cup \{S' \to \alpha \mid S \to \alpha \in P\}.$$

Allora

Proposizione

Sia $G = (V, T, P, S)$ *una grammatica di tipo* i, $i = 1, 2, 3$.
Sia $G' = (V', T, P', S')$, *con* $S' \notin V$, $V' = V \cup \{S'\}$ *e dove* P' *è l'insieme delle produzioni in* P *più tutte le produzioni* $S' \to \alpha$, *dove* $S \to \alpha \in P$, *cioè*

$$P' = P \cup \{S' \to \alpha \mid S \to \alpha \in P\}.$$

Allora

1. G' *è una grammatica di tipo* i, $i = 1, 2, 3$;

Proposizione

Sia $G = (V, T, P, S)$ *una grammatica di tipo* i, $i = 1, 2, 3$.
Sia $G' = (V', T, P', S')$, *con* $S' \notin V$, $V' = V \cup \{S'\}$ *e dove* P' *è l'insieme delle produzioni in* P *più tutte le produzioni* $S' \to \alpha$, *dove* $S \to \alpha \in P$, *cioè*

$$P' = P \cup \{S' \to \alpha \mid S \to \alpha \in P\}.$$

Allora

1. G' *è una grammatica di tipo* i, $i = 1, 2, 3$;
2. $L(G') = L(G)$;

Proposizione

Sia $G = (V, T, P, S)$ *una grammatica di tipo* i, $i = 1, 2, 3$.
Sia $G' = (V', T, P', S')$, *con* $S' \notin V$, $V' = V \cup \{S'\}$ *e dove* P' *è l'insieme delle produzioni in* P *più tutte le produzioni* $S' \to \alpha$, *dove* $S \to \alpha \in P$, *cioè*

$$P' = P \cup \{S' \to \alpha \mid S \to \alpha \in P\}.$$

Allora

1. G' *è una grammatica di tipo* i, $i = 1, 2, 3$;
2. $L(G') = L(G)$;
3. *Il simbolo iniziale di* S' *non compare alla destra di nessuna produzione in* G'.

- **Esempio**: Sia G la grammatica definita da

$$S \to aSb \mid ab$$

La grammatica G' definita da

$$S' \to aSb \mid ab, \quad S \to aSb \mid ab$$

è equivalente a G e inoltre il simbolo iniziale non compare alla destra di nessuna produzione in G.

- **Esempio**: Sia G la grammatica definita da

$$S \rightarrow aSb \mid ab$$

La grammatica G' definita da

$$S' \rightarrow aSb \mid ab, \quad S \rightarrow aSb \mid ab$$

è equivalente a G e inoltre il simbolo iniziale non compare alla destra di nessuna produzione in G.

- Sia S' il simbolo iniziale di G'. Se S' non appare alla destra di nessuna produzione in G' e aggiungiamo $S' \rightarrow \epsilon$, questa produzione potrebbe essere usata solo come come primo (e unico) passo in una derivazione.

- Una grammatica $G = (V, T, P, S)$, con V, T insiemi finiti e disgiunti, $S \in V$ è

- Una grammatica $G = (V, T, P, S)$, con V, T insiemi finiti e disgiunti, $S \in V$ è
  - **di tipo 1** se ogni produzione in P ha la forma $\alpha \to \beta$, $\alpha, \beta \in (V \cup T)^+$ e $|\beta| \geq |\alpha|$.

- Una grammatica $G = (V, T, P, S)$, con V, T insiemi finiti e disgiunti, $S \in V$ è
  - **di tipo 1** se ogni produzione in P ha la forma $\alpha \to \beta$, $\alpha, \beta \in (V \cup T)^+$ e $|\beta| \geq |\alpha|$.
  - **di tipo 2** se ogni produzione in P ha la forma $A \to \beta$, con $A \in V$ e $\beta \in (V \cup T)^+$

- Una grammatica $G = (V, T, P, S)$, con V, T insiemi finiti e disgiunti, $S \in V$ è
  - **di tipo 1** se ogni produzione in P ha la forma $\alpha \to \beta$, $\alpha, \beta \in (V \cup T)^+$ e $|\beta| \geq |\alpha|$.
  - **di tipo 2** se ogni produzione in P ha la forma $A \to \beta$, con $A \in V$ e $\beta \in (V \cup T)^+$
  - **di tipo 3** se ogni produzione in P ha la forma $A \to aB$ o $A \to a$, dove A, B sono variabili e a è un terminale.

- Una grammatica $G = (V, T, P, S)$, con V, T insiemi finiti e disgiunti, $S \in V$ è
  - **di tipo 1** se ogni produzione in P ha la forma $\alpha \to \beta$, $\alpha, \beta \in (V \cup T)^+$ e $|\beta| \geq |\alpha|$.
  - **di tipo 2** se ogni produzione in P ha la forma $A \to \beta$, con $A \in V$ e $\beta \in (V \cup T)^+$
  - **di tipo 3** se ogni produzione in P ha la forma $A \to aB$ o $A \to a$, dove A, B sono variabili e a è un terminale.
- In più permettiamo che $S \to \epsilon$ sia in P a condizione che S non compaia alla destra di nessuna produzione.

- Confrontiamo le classi di linguaggi generati con la vecchia e la nuova definizione.

- Confrontiamo le classi di linguaggi generati con la vecchia e la nuova definizione.
- Se L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la nuova definizione ed $\epsilon \notin L$, allora L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la vecchia definizione.

- Confrontiamo le classi di linguaggi generati con la vecchia e la nuova definizione.
- Se L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la nuova definizione ed $\epsilon \notin L$, allora L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la vecchia definizione.
- Se L è generato da una grammatica G di tipo i, $i = 1, 2, 3$, secondo la nuova definizione ed $\epsilon \in L$, elimino in G la produzione $S \to \epsilon$ (usata solo per generare ϵ) e ottengo una grammatica che genera $L \setminus \{\epsilon\}$ ed è di tipo i, $i = 1, 2, 3$, secondo la vecchia definizione.

- Confrontiamo le classi di linguaggi generati con la vecchia e la nuova definizione.
- Se L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la nuova definizione ed $\epsilon \notin L$, allora L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la vecchia definizione.
- Se L è generato da una grammatica G di tipo i, $i = 1, 2, 3$, secondo la nuova definizione ed $\epsilon \in L$, elimino in G la produzione $S \rightarrow \epsilon$ (usata solo per generare ϵ) e ottengo una grammatica che genera $L \setminus \{\epsilon\}$ ed è di tipo i, $i = 1, 2, 3$, secondo la vecchia definizione.
- Se L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la vecchia definizione, allora L è generato da una grammatica di tipo i, $i = 1, 2, 3$, secondo la nuova definizione.

- Data $G = (V, T, P, S)$ di tipo i, $i \geq 1$, (vecchia definizione) che genera $L(G)$, possiamo trovare una grammatica $G' = (V', T, P', S')$ di tipo i, $i \geq 1$, (nuova definizione) che genera $L(G) \cup \{\epsilon\}$ come segue

- Data $G = (V, T, P, S)$ di tipo i, $i \geq 1$, (vecchia definizione) che genera $L(G)$, possiamo trovare una grammatica $G' = (V', T, P', S')$ di tipo i, $i \geq 1$, (nuova definizione) che genera $L(G) \cup \{\epsilon\}$ come segue
  - introduciamo una nuova variabile S' che sarà il simbolo iniziale di G': $V' = V \cup \{S'\}$

- Data $G = (V, T, P, S)$ di tipo i, $i \geq 1$, (vecchia definizione) che genera $L(G)$, possiamo trovare una grammatica $G' = (V', T, P', S')$ di tipo i, $i \geq 1$, (nuova definizione) che genera $L(G) \cup \{\epsilon\}$ come segue
  - introduciamo una nuova variabile S' che sarà il simbolo iniziale di G': $V' = V \cup \{S'\}$
  - definiamo P' come l'insieme delle produzioni in P più tutte le produzioni $S' \rightarrow \alpha$, dove $S \rightarrow \alpha \in P$.

- Data $G = (V, T, P, S)$ di tipo i, $i \geq 1$, (vecchia definizione) che genera $L(G)$, possiamo trovare una grammatica $G' = (V', T, P', S')$ di tipo i, $i \geq 1$, (nuova definizione) che genera $L(G) \cup \{\epsilon\}$ come segue
  - introduciamo una nuova variabile S' che sarà il simbolo iniziale di G': $V' = V \cup \{S'\}$
  - definiamo P' come l'insieme delle produzioni in P più tutte le produzioni $S' \rightarrow \alpha$, dove $S \rightarrow \alpha \in P$.
  - aggiungiamo la produzione $S' \rightarrow \epsilon$.

- Data $G = (V, T, P, S)$ di tipo i, $i \geq 1$, (vecchia definizione) che genera $L(G)$, possiamo trovare una grammatica $G' = (V', T, P', S')$ di tipo i, $i \geq 1$, (nuova definizione) che genera $L(G) \cup \{\epsilon\}$ come segue
  - introduciamo una nuova variabile S' che sarà il simbolo iniziale di G': $V' = V \cup \{S'\}$
  - definiamo P' come l'insieme delle produzioni in P più tutte le produzioni $S' \to \alpha$, dove $S \to \alpha \in P$.
  - aggiungiamo la produzione $S' \to \epsilon$.
    Quindi

$$P' = P \cup \{S' \to \alpha \mid S \to \alpha \in P\} \cup \{S' \to \epsilon\}$$

Torniamo alla domanda precedente.

Torniamo alla domanda precedente.

La definizione di grammatica context-free non coincide con quella di grammatica di tipo 2.

Torniamo alla domanda precedente.

La definizione di grammatica context-free non coincide con quella di grammatica di tipo 2.

Possiamo dire che la classe dei linguaggi context-free coincide con quella dei linguaggi generati da una grammatica di tipo 2?

Torniamo alla domanda precedente.

La definizione di grammatica context-free non coincide con quella di grammatica di tipo 2.

Possiamo dire che la classe dei linguaggi context-free coincide con quella dei linguaggi generati da una grammatica di tipo 2?

SÌ

Una grammatica di tipo 2 è un caso particolare di grammatica context-free. Quindi se un linguaggio L è generato da una grammatica di tipo 2, allora L è generato da una grammatica context-free.

Viceversa, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ è possibile ottenere G' tale che

Viceversa, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ è possibile ottenere G' tale che

1. G' non ha ϵ-produzioni, cioè ogni produzione in G' ha la forma $A \to \beta$, dove A è una variabile e β è una stringa non vuota di variabili e terminali.

Viceversa, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ è possibile ottenere G' tale che

1. G' non ha ϵ-produzioni, cioè ogni produzione in G' ha la forma $A \to \beta$, dove A è una variabile e β è una stringa non vuota di variabili e terminali.
2. G' genera $L(G) \setminus \{\epsilon\}$, cioè $L(G') = L(G) \setminus \{\epsilon\}$.

Viceversa, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ è possibile ottenere G' tale che

1. G' non ha ϵ-produzioni, cioè ogni produzione in G' ha la forma $A \to \beta$, dove A è una variabile e β è una stringa non vuota di variabili e terminali.
2. G' genera $L(G) \setminus \{\epsilon\}$, cioè $L(G') = L(G) \setminus \{\epsilon\}$.

Omettiamo la prova. La prova del punto 1 è costruttiva.

Quindi, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ costruiamo prima $G' = (V', T, P', S')$ tale che $L(G') = L(G) \setminus \{\epsilon\}$ e senza produzioni della forma $A \to \epsilon$, dove A è una variabile.

Quindi, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ costruiamo prima $G' = (V', T, P', S')$ tale che $L(G') = L(G) \setminus \{\epsilon\}$ e senza produzioni della forma $A \rightarrow \epsilon$, dove A è una variabile.

Ovviamente se $\epsilon \notin L(G)$ allora G' è una grammatica di tipo 2, equivalente a G.

Quindi, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ costruiamo prima $G' = (V', T, P', S')$ tale che $L(G') = L(G) \setminus \{\epsilon\}$ e senza produzioni della forma $A \to \epsilon$, dove A è una variabile.

Ovviamente se $\epsilon \notin L(G)$ allora G' è una grammatica di tipo 2, equivalente a G.

Se $\epsilon \in L(G)$, applichiamo a G' la costruzione precedente e otteniamo una grammatica G'' di tipo 2, equivalente a G.

Quindi, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ costruiamo prima $G' = (V', T, P', S')$ tale che $L(G') = L(G) \setminus \{\epsilon\}$ e senza produzioni della forma $A \to \epsilon$, dove A è una variabile.

Ovviamente se $\epsilon \notin L(G)$ allora G' è una grammatica di tipo 2, equivalente a G.

Se $\epsilon \in L(G)$, applichiamo a G' la costruzione precedente e otteniamo una grammatica G'' di tipo 2, equivalente a G.

Nel caso delle grammatiche context-free esiste una trasformazione più semplice per passare da G' a una grammatica G_1 di tipo 2, equivalente a G.

Quindi, data una grammatica context-free $G = (V, T, P, S)$ che genera $L(G)$ costruiamo prima $G' = (V', T, P', S')$ tale che $L(G') = L(G) \setminus \{\epsilon\}$ e senza produzioni della forma $A \to \epsilon$, dove A è una variabile.

Ovviamente se $\epsilon \notin L(G)$ allora G' è una grammatica di tipo 2, equivalente a G.

Se $\epsilon \in L(G)$, applichiamo a G' la costruzione precedente e otteniamo una grammatica G'' di tipo 2, equivalente a G.

Nel caso delle grammatiche context-free esiste una trasformazione più semplice per passare da G' a una grammatica G_1 di tipo 2, equivalente a G.

Possiamo infatti considerare $G_1 = (V_1, T, P_1, S_1)$, dove $V_1 = V' \cup \{S_1\}$ e $P_1 = P' \cup \{S_1 \to S', S_1 \to \epsilon\}$.

Torniamo alla gerarchia di Chomsky

Alla gerarchia di grammatiche introdotta corrisponde una gerarchia sui linguaggi generati.

Alla gerarchia di grammatiche introdotta corrisponde una gerarchia sui linguaggi generati.

Definizione

Un linguaggio L è di tipo i se esiste una grammatica G di tipo i tale che $L = L(G)$.

Alla gerarchia di grammatiche introdotta corrisponde una gerarchia sui linguaggi generati.

Definizione

Un linguaggio L è di tipo i se esiste una grammatica G di tipo i tale che $L = L(G)$.

Otteniamo quattro classi di linguaggi, i linguaggi di tipo 0, 1, 2, 3. Le quattro classi formano la

Gerarchia di Chomsky

Definizione

Un linguaggio L è di tipo i se esiste una grammatica G di tipo i tale che $L = L(G)$.

Definizione

Un linguaggio L è di tipo i se esiste una grammatica G di tipo i tale che $L = L(G)$.

Otteniamo quattro classi di linguaggi, i linguaggi di tipo 0, 1, 2, 3.

Definizione

Un linguaggio L è di tipo i se esiste una grammatica G di tipo i tale che $L = L(G)$.

Otteniamo quattro classi di linguaggi, i linguaggi di tipo 0, 1, 2, 3.

Che relazione c'è tra le quattro classi?

Siccome la classe delle grammatiche di tipo i include strettamente quella delle grammatiche di tipo $i+1$, possiamo dire che ogni linguaggio di tipo $i+1$ è anche di tipo i, $0 \leq i \leq 2$.

Siccome la classe delle grammatiche di tipo i include strettamente quella delle grammatiche di tipo $i+1$, possiamo dire che ogni linguaggio di tipo $i+1$ è anche di tipo i, $0 \leq i \leq 2$.

Cioè la classe dei linguaggi di tipo $i+1$ è contenuta in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Siccome la classe delle grammatiche di tipo i include strettamente quella delle grammatiche di tipo $i+1$, possiamo dire che ogni linguaggio di tipo $i+1$ è anche di tipo i, $0 \leq i \leq 2$.

Cioè la classe dei linguaggi di tipo $i+1$ è contenuta in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Possiamo però dimostrare il seguente più forte risultato.

Teorema

La classe dei linguaggi di tipo $i+1$ è contenuta strettamente in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Teorema

La classe dei linguaggi di tipo $i+1$ è contenuta strettamente in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Teorema

La classe dei linguaggi di tipo $i+1$ è contenuta strettamente in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Per provare questo teorema occorre esibire:

Teorema

La classe dei linguaggi di tipo $i+1$ è contenuta strettamente in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Per provare questo teorema occorre esibire:

- Un linguaggio generato da una grammatica di tipo zero che non può essere generato da una grammatica dipendente dal contesto.

Teorema

La classe dei linguaggi di tipo $i+1$ è contenuta strettamente in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Per provare questo teorema occorre esibire:

- Un linguaggio generato da una grammatica di tipo zero che non può essere generato da una grammatica dipendente dal contesto.
- Un linguaggio generato da una grammatica dipendente dal contesto che non può essere generato da una grammatica libera dal contesto.

Teorema

La classe dei linguaggi di tipo $i+1$ è contenuta strettamente in quella dei linguaggi di tipo i, $0 \leq i \leq 2$.

Per provare questo teorema occorre esibire:

- Un linguaggio generato da una grammatica di tipo zero che non può essere generato da una grammatica dipendente dal contesto.
- Un linguaggio generato da una grammatica dipendente dal contesto che non può essere generato da una grammatica libera dal contesto.
- Un linguaggio generato da una grammatica libera dal contesto che non può essere generato da una grammatica regolare.

Abbiamo definito una gerarchia di grammatiche e una corrispondente gerarchia di linguaggi.

Abbiamo definito una gerarchia di grammatiche e una corrispondente gerarchia di linguaggi.

Vedremo una corrispondente gerarchia di automi che ci permetterà di fornire gli esempi di linguaggi che separano le classi.

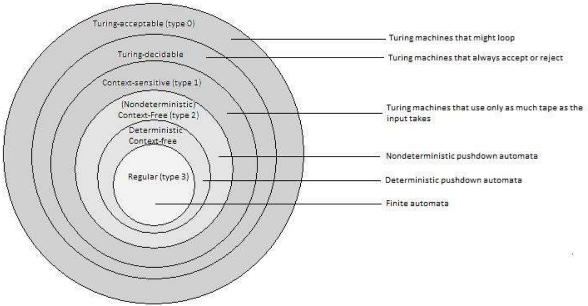

Figura: Una gerarchia di linguaggi

Esaminiamo, attraverso alcuni esempi, le relazioni tra le classi di linguaggi appena introdotte e alcuni aspetti dei linguaggi di programmazione, astraendo da molti dettagli di cui si deve tener conto quando si tratti di linguaggi di programmazione reali.

Esaminiamo, attraverso alcuni esempi, le relazioni tra le classi di linguaggi appena introdotte e alcuni aspetti dei linguaggi di programmazione, astraendo da molti dettagli di cui si deve tener conto quando si tratti di linguaggi di programmazione reali.

$$
\begin{aligned}
L_1 &= \{wcw \mid w \in \{a, b\}^+\} \\
L_1' &= \{a^n b^m c^n d^m \mid n, m \geq 1\}
\end{aligned}
$$

Questi linguaggi sono di tipo 1 (dipendenti dal contesto), ma non context-free.

$$
\begin{aligned}
L_1 &= \{wcw \mid w \in \{a, b\}^+\} \\
L_1' &= \{a^n b^m c^n d^m \mid n, m \geq 1\}
\end{aligned}
$$

$$
\begin{aligned}
L_1 &= \{wcw \mid w \in \{a, b\}^+\} \\
L_1' &= \{a^n b^m c^n d^m \mid n, m \geq 1\}
\end{aligned}
$$

Un'interpretazione di L_1 vede il prefisso w come la dichiarazione di un identificatore con tale nome, c come (una parte di) un programma, e il suffisso w come l'uso dell'identificatore medesimo.

$$L_1 = \{wcw \mid w \in \{a, b\}^+\}$$
$$L_1' = \{a^n b^m c^n d^m \mid n, m \geq 1\}$$

Un'interpretazione di L_1 vede il prefisso w come la dichiarazione di un identificatore con tale nome, c come (una parte di) un programma, e il suffisso w come l'uso dell'identificatore medesimo.

Per quanto riguarda le stringhe di L_1' potremmo supporre di avere due procedure con n ed m parametri rispettivamente. Allora possiamo interpretare a^n e b^m come i parametri formali che appaiono nella dichiarazione delle procedure in questione e c^n e d^m come i parametri attuali delle loro chiamate.

In generale, un linguaggio di programmazione non è context-free.

In generale, un linguaggio di programmazione non è context-free.

La sintassi dei linguaggi di programmazione è data in termini di linguaggi context-free, e quindi l'aspetto definizione-uso contestuale non è esprimibile da una grammatica libera.

In generale, un linguaggio di programmazione non è context-free.

La sintassi dei linguaggi di programmazione è data in termini di linguaggi context-free, e quindi l'aspetto definizione-uso contestuale non è esprimibile da una grammatica libera.

Infatti, il controllo che un identificatore sia definito prima del suo uso o che i parametri attuali siano tanti quanti i formali viene solitamente fatto, **per ragioni di efficienza**, da un componente specifico dei compilatori o degli interpreti.

Come ulteriore esempio, interpretando il carattere a come l'apertura di una delle tante parentesi usate nei programmi, per esempio (, e il carattere b come la parentesi chiusa, per esempio), una stringa appartiene a $\{a^n b^n \mid n \geq 1\}$ se tutte le parentesi aperte vengono chiuse.

Più in generale, una stringa di parentesi $(,)$ è bilanciata se è possibile associare una parentesi aperta a quella chiusa immediatamente alla sua destra e, cancellandole e ripetendo il procedimento, si giunge a cancellarle tutte.

Definizione

Una stringa di parentesi $(,)$ *è bilanciata se soddisfa le seguenti due regole.*

Definizione

Una stringa di parentesi $(,)$ *è bilanciata se soddisfa le seguenti due regole.*

1. *Una stringa bilanciata contiene lo stesso numero di parentesi aperte e chiuse.*

Definizione

Una stringa di parentesi $($, $)$ *è bilanciata se soddisfa le seguenti due regole.*

1. *Una stringa bilanciata contiene lo stesso numero di parentesi aperte e chiuse.*
2. *Seguendo la stringa da sinistra a destra il suo profilo non diventa mai negativo, dove il* **profilo** *è il valore ottenuto sottraendo al numero delle parentesi aperte incontrate il numero delle parentesi chiuse incontrate.*

Definizione

Una stringa di parentesi $(,)$ *è bilanciata se soddisfa le seguenti due regole.*

1. *Una stringa bilanciata contiene lo stesso numero di parentesi aperte e chiuse.*
2. *Seguendo la stringa da sinistra a destra il suo profilo non diventa mai negativo, dove il* **profilo** *è il valore ottenuto sottraendo al numero delle parentesi aperte incontrate il numero delle parentesi chiuse incontrate.*

Ad esempio ()(), (()()), ϵ sono bilanciate mentre)(e (() non lo sono.

Definizione ricorsiva delle stringhe di parentesi bilanciate

Definizione ricorsiva delle stringhe di parentesi bilanciate

PASSO BASE: La stringa vuota ϵ è una stringa di parentesi bilanciata.

Definizione ricorsiva delle stringhe di parentesi bilanciate

PASSO BASE: La stringa vuota ϵ è una stringa di parentesi bilanciata.

PASSO RICORSIVO: Se x e y sono stringhe di parentesi bilanciate, allora anche $(x)y$ è una stringa di parentesi bilanciata.

Definizione ricorsiva delle stringhe di parentesi bilanciate

PASSO BASE: La stringa vuota ϵ è una stringa di parentesi bilanciata.

PASSO RICORSIVO: Se x e y sono stringhe di parentesi bilanciate, allora anche $(x)y$ è una stringa di parentesi bilanciata.

$G_{bal} = (\{B\}, \{(,)\}, P, B)$ dove P consiste nelle produzioni

$$B \rightarrow BB \mid (B) \mid \epsilon$$

genera tutte e sole le stringhe di parentesi bilanciate.