UNIVERSITÀ
DEGLI STUDI
DI TORINO

Linguaggi liberi dal contesto

a.a. 2018-2019

Riferimenti bibliografici

Automi, Linguaggi e Calcolabilità, J. E. Hopcroft, R. Motwani, J. D. Ullman

Linguaggi e grammatiche libere dal contesto [cap. 5]

- 5.1 Grammatiche libere dal contesto.
  - 5.1.1 Un esempio informale.
  - 5.1.2 Definizione delle grammatiche libere dal contesto.
  - 5.1.3 Derivazioni per mezzo di una grammatica.
  - 5.1.4 Derivazioni a sinistra e a destra.
  - 5.1.5 Il linguaggio di una grammatica.
  - 5.1.6 Forme sentenziali.
- 5.2 Alberi sintattici.
  - 5.2.1 Costruzione di alberi sintattici.
  - 5.2.2 Il prodotto di un albero sintattico.
- 5.3 Applicazioni delle grammatiche libere.
  - 5.3.1 Parser. (leggere)
  - 5.3.3 Linguaggi di markup. (leggere)
- 5.4 Ambiguità nelle grammatiche e nei linguaggi.
  - 5.4.1 Grammatiche ambigue.
  - 5.4.3 Derivazioni a sinistra come modo per esprimere l'ambiguità. (Teorema 5.29 senza dimostrazione).
  - 5.4.4 Ambiguità inerente.

Linguaggi non regolari

I linguaggi regolari sono troppo semplici: le espressioni regolari non sono in grado di descrivere, ad esempio, le espressioni aritmetiche parentesizzate o la struttura di un linguaggio di programmazione.

Il linguaggio $\{0^n1^n \mid n \geq 0\}$ non è denotabile con un'espressione regolare (e non è riconosciuto da nessun automa finito).

Come queste stringhe possono essere definite induttivamente?

N_{01}: $\quad \varepsilon \in N_{01}$

$\quad\quad x \in N_{01} \Rightarrow 0x1 \in N_{01}$

Espressioni aritmetiche parentesizzate, con gli operatori + e $*$, formate su identificatori denotati dalla espressione regolare: (a+b)(a+b+0+1)*

ID: $\quad a, b \in ID$

$\quad\quad x \in ID \Rightarrow x0, x1, xa, xb \in ID$

E: $\quad ID \in E$

$\quad\quad x, y \in E \Rightarrow x+y, x*y, (x) \in E$

Linguaggi con una naturale notazione ricorsiva.

Linguaggi non regolari

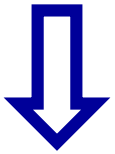
Si può fornire una descrizione di questi linguaggi “non regolari” con una grammatica, nel modo seguente:

$N_{01} \rightarrow \varepsilon$
$N_{01} \rightarrow 0N_{01}1$

$E \rightarrow ID$
$E \rightarrow E + E$
$E \rightarrow E * E$
$E \rightarrow (E)$

Anche per l’insieme I, descritto dall’espressione regolare, la descrizione può essere data con una grammatica:

$ID \rightarrow a$ $\quad$ $ID \rightarrow b$
$ID \rightarrow ID\ a$ $\quad$ $ID \rightarrow ID\ b$
$ID \rightarrow ID\ 0$ $\quad$ $ID \rightarrow ID1$



Grammatiche libere dal contesto

Grammatiche libere dal contesto

In informatica le grammatiche *libere dal contesto* (*libere*) sono usate nelle:

- Definizioni di linguaggi di programmazione.
- Realizzazioni di analizzatori sintattici (parser).
- Descrizioni dei tag ammessi nella definizione dei documenti (DTD), utilizzate dagli utenti di XML per lo scambio di informazioni nel Web.

In analogia con il comando Lex per la generazione di analizzatori lessicali, nei sistemi UNIX il comando YACC serve a generare analizzatori sintattici.

L'input di YACC è una grammatica libera dal contesto, specificata con una notazione molto vicina a quella che vedremo.

Grammatiche libere dal contesto

Una grammatica *G libera dal contesto* è una quadrupla <V, T, P, S> tale che:

V è un insieme finito di simboli (*non terminali* o *variabili*)

T è un insieme finito di simboli (*terminali*)

P è un insieme di *regole di riscrittura* o *produzioni sintattiche*

$S \in V$ è l'*assioma* o *start symbol* della grammatica

le regole sono coppie: <A, α>, che scriveremo:

$$A \rightarrow \alpha$$

dove $A \in V$ (A è un simbolo non terminale) e α una stringa di simboli terminali e non terminali ($\alpha \in (V \cup T)^*$)

Esempio

$$< \{S, A\}, \{a, b\}, \{S \rightarrow aAb, A \rightarrow bSa, A \rightarrow ab\}, S >$$

Usiamo $A \rightarrow \alpha_1 \mid \ldots \mid \alpha_n$ come abbreviazione per $A \rightarrow \alpha_1, \ldots, A \rightarrow \alpha_n$

$$<\{S, A\}, \{a, b\}, \{S \rightarrow aAb, A \rightarrow bSa \mid ab\}, S>$$

Grammatiche libere dal contesto: derivazioni

In una grammatica G la stringa β *produce* o *si riscrive come* la stringa γ: $\beta \Rightarrow \gamma$

se $\beta, \gamma \in (V \cup T)^*$ e $\beta = \delta A \mu$ e $\gamma = \delta \alpha \mu$ e $A \rightarrow \alpha \in P$

$\beta \Rightarrow^n \gamma$ (β produce γ in n passi) se $\beta = x_0 \Rightarrow x_1 \Rightarrow \ldots \Rightarrow x_{n-1} \Rightarrow x_n = \gamma$

$\beta \Rightarrow^* \gamma$ se $\beta \Rightarrow^n \gamma$ per qualche $n \geq 0$ ($\Rightarrow^*$ chiusura riflessiva e transitiva della relazione $\Rightarrow$)

$\beta \Rightarrow^+ \gamma$ se $\beta \Rightarrow^n \gamma$ per qualche $n > 0$ ($\Rightarrow^+$ chiusura transitiva della relazione $\Rightarrow$)

Grammatiche libere dal contesto: derivazioni

Esempi:

1) $< \{S, A\}, \{a, b\}, \{S \to aAb, A \to bSa \mid ab\}, S >$

$aAb \Rightarrow abSab$ $\qquad$ $S \Rightarrow aAb \Rightarrow abSab$

$S \Rightarrow^2 abSab$

2) $< \{E, I\}, \{a, b, 0, 1, +, *, (,)\},$
$\{E \to I \mid E + E \mid E * E \mid (E), I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1\}, E >$

$E \Rightarrow E * E \Rightarrow I * E \Rightarrow I * (E) \Rightarrow I * (E + E) \Rightarrow a * (E + E) \Rightarrow$
$\Rightarrow a * (E + I) \Rightarrow a * (E + I0) \Rightarrow a * (b + I0) \Rightarrow a * (b + b0)$

3) $< \{PAL, L\}, \{a, b, ;\},$
$\{L \to L ; PAL \mid PAL, PAL \to \varepsilon \mid a\ PAL\ a \mid b\ PAL\ b\}, L >$

$L ; pal \Rightarrow L ; a\ PAL\ a \Rightarrow L ; aa\ PAL\ aa \Rightarrow L ; PAL ; aa\ PAL\ aa \Rightarrow$
$\Rightarrow L ; b\ PAL\ b;aa\ PAL\ aa \Rightarrow b\ PAL\ b;b\ PAL\ b;aa\ PAL\ aa \Rightarrow$
$\Rightarrow b\ PAL\ b;b\ PAL\ b;aab\ PAL\ baa \Rightarrow bb;b\ PAL\ b;aab\ PAL\ baa \Rightarrow$
$\Rightarrow bb;b\ PAL\ b;aabbaa \Rightarrow bb;ba\ PAL\ ab;aabbaa \Rightarrow$
$\Rightarrow bb;baab;aabbaa$

Grammatiche libere dal contesto: derivazioni

Due strategie per le derivazioni: derivazioni a destra e a sinistra.

Derivazione a sinistra (*leftmost*): sostituire ad ogni passo la variabile più a sinistra con uno dei corpi delle sue produzioni

$$E \underset{lm}{\Rightarrow} E * E \underset{lm}{\Rightarrow} I * E \underset{lm}{\Rightarrow} a * E \underset{lm}{\Rightarrow} a * (E) \Rightarrow$$
$$\Rightarrow a * (E + E) \Rightarrow a * (I + E) \Rightarrow$$
$$\Rightarrow a * (b + E) \Rightarrow a * (b + I) \Rightarrow a * (b + I0) \underset{lm}{\Rightarrow} a * (b + b0)$$
$$E \underset{lm}{\Rightarrow}^{*} a * (b + b0)$$

Derivazione a destra (*rightmost*): sostituire ad ogni passo la variabile più a destra con uno dei corpi delle sue produzioni

$$E \underset{rm}{\Rightarrow} E * E \underset{rm}{\Rightarrow} E * (E) \Rightarrow E * (E + E) \Rightarrow$$
$$\Rightarrow E * (E + I) \Rightarrow E * (E + I0) \Rightarrow E * (E + b0) \Rightarrow$$
$$\Rightarrow E * (I + b0) \Rightarrow E * (b + b0) \Rightarrow I * (b + b0) \underset{rm}{\Rightarrow} a * (b + b0)$$
$$E \underset{rm}{\Rightarrow}^{*} a * (b + b0)$$

Grammatiche libere dal contesto: linguaggio generato

Il *linguaggio generato dal non terminale A in una grammatica* (notazione $L_A(G)$) è l'insieme delle *stringhe di terminali* prodotte da A:

$$L_A(G) = \{x \mid x \in T^* \ \& \ A \Rightarrow^+ x\}$$

Il *linguaggio generato da una grammatica* (notazione L(G)) è l'insieme delle *stringhe di terminali* prodotte dall'assioma:

$$L(G) = \{x \mid x \in T^* \ \& \ S \Rightarrow^+ x\}$$

Le derivazioni dallo start symbol producono stringhe di terminali e variabili che vengono chiamate *forme sentenziali*.

Qualunque stringa $\alpha \in (V \cup T)^*$ tale che esiste una derivazione dall'assioma $S \Rightarrow^* \alpha$, è una forma sentenziale.

Il linguaggio generato da una grammatica è l'insieme delle forme sentenziali formate solo da simboli terminali.

Grammatiche libere dal contesto

Esempi:

1) $S \rightarrow aSb \mid ab$ $\quad L(G) = \{a^n b^n \mid n > 0\}$

2) $S \rightarrow aB \mid bA$
$A \rightarrow a \mid aS \mid bAA$
$B \rightarrow b \mid bS \mid aBB$
$\quad L(G) = \{x \mid x \in \{a, b\}^* \ \& \ n_a(x) = n_b(x) > 0\}$

3) Grammatica che genera le espressioni regolari su $\Sigma = \{0, 1\}$:

$G = < \{R\}, \{\Phi, \varepsilon, 0, 1, (,), +, *\}, \{R \rightarrow \Phi \mid \varepsilon \mid 0 \mid 1 \mid (R+R) \mid (RR) \mid (R)^*\}, R >$

$R \Rightarrow (RR) \Rightarrow (R(R+R)) \Rightarrow (1(R+R)) \Rightarrow (1(\varepsilon+R)) \Rightarrow (1(\varepsilon+0))$

$R \Rightarrow (RR) \Rightarrow (R(R)^*) \Rightarrow (1((R+R))^*) \Rightarrow^* (1((0+1))^*)$

Grammatiche libere dal contesto

Esempi:

4) $G = \langle \{S, A\}, \{a, b\}, \{S \rightarrow aAb, A \rightarrow bSa \mid ab\}, S \rangle$

$A \Rightarrow bSa \Rightarrow baAba \Rightarrow baabba$
$A \Rightarrow bSa \Rightarrow baAba \Rightarrow babSaba \Rightarrow babaAbaba \Rightarrow babaabbaba$

$S \Rightarrow aAb \Rightarrow aabb$
$S \Rightarrow aAb \Rightarrow abSab \Rightarrow abaAbab \Rightarrow abaabbab$

$L_A(G) = \{baabba, babaabbaba, \ldots\} = \{(ba)^n ab (ba)^n \mid \geq 0\}$

$L(G) = \{aabb, abaabbab, \ldots\} = \{(ab)^n aabb (ab)^n \mid \geq 0\}$

5) $G = \langle \{Z, U\}, \{0, 1, 2\}, \{Z \rightarrow 0Z1 \mid U, U \rightarrow 1U \mid 1\}, Z \rangle$

$L(G) = ?$

Esercizi

Fornire una grammatica libera dal contesto per ognuno dei seguenti linguaggi:

- $\{a^n b^m c^p \mid n = m + p \quad e \quad n \geq 0\}$
- $\{a^n c\, b^n \mid n \geq 0\}$
- $\{(ab)^n (cd)^n \mid n > 0\}$
- $\{a^n b^m c^m d^n \mid n, m \geq 0\}$
- $\{a^i b^{3j} \mid 0 \leq i < j\}$
- $\{a^{2i} b^j c^i \mid i, j \geq 0\}$
- $\{a^n b^m c^p d^q \mid n + m = p + q \quad e \quad n, m, p, q \geq 0\}$

Esercizi

Per ognuno dei seguenti linguaggi:

- $\{ab^n ab^{2n} \mid n > 0\}$
- $\{ab^n c^k b^n \mid k, n > 0\}$
- $\{ab^n c^k b^m \mid k, n, m > 0\}$
- $\{a^n b^n a^k b^k \mid k, n > 0\}$

fornire una grammatica libera che lo generi; se il linguaggio è regolare fornire anche un'espressione regolare che lo denoti.

Alberi sintattici

Gli alberi sintattici di una grammatica G sono rappresentazioni (ad albero) delle derivazioni in G.

Data una grammatica G = <V, T, P, S>, gli alberi sintattici di G sono alberi che soddisfano le seguenti condizioni:

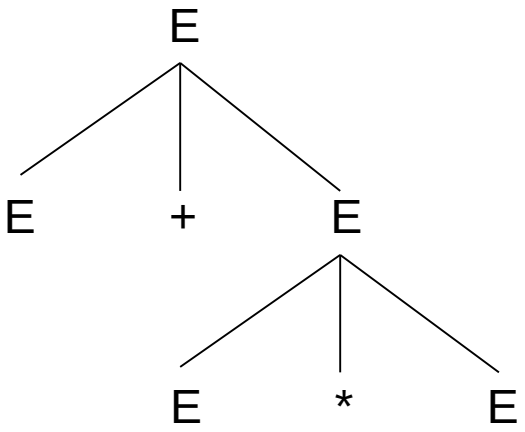
1) Ogni nodo interno è etichettato da una variabile;
2) Ogni foglia è etichettata da una variabile, da un terminale o da ε;
3) Se una foglia è etichettata ε, allora è l'unico figlio del nodo padre;
4) Se un nodo interno è etichettato A e i suoi figli sono etichettati, da sinistra verso destra, $X_1, X_2, \ldots, X_k$, allora $A \rightarrow X_1, X_2, \ldots, X_k$ è una produzione della grammatica.

Alberi sintattici

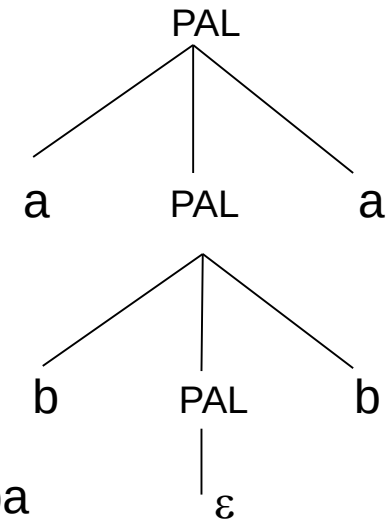
Esempi:

< {E, ID}, {a, b,+, $*$},
{E $\rightarrow$ ID | E + E | E $*$ E, ID $\rightarrow$ a | b}, E >

< {PAL}, {a, b},
{PAL $\rightarrow \varepsilon$ | a PAL a | b PAL b}, PAL >



Prodotto: E+E $*$ E



Prodotto: abba

Gli alberi sintattici delle parole del linguaggio sono alberi in cui:

- il prodotto è una stringa di terminali (le foglie sono etichettate da simboli terminali o da ε);
- la radice è etichettata dallo start symbol.

Alberi sintattici

Data una grammatica G = <V, T, P, S> i seguenti enunciati sono equivalenti:

1) La stringa w è nel linguaggio della variabile A ($w \in L_A$);
2) $A \Rightarrow^* w$;
3) $A \underset{lm}{\Rightarrow}^* w$;
4) $A \underset{rm}{\Rightarrow}^* w$;
5) Esiste un albero sintattico con radice A e *prodotto* w.

Per ogni albero sintattico si ha una e una sola derivazione destra e una e una sola derivazione sinistra.

Esercizi

- Costruire una derivazione a sinistra, una derivazione a destra e un albero sintattico per ognuna delle stringhe "bbc" e "aaabcc" nella grammatica:
  - $S \rightarrow ABC$
  - $A \rightarrow aA \quad A \rightarrow \varepsilon$
  - $B \rightarrow bB \quad B \rightarrow \varepsilon$
  - $C \rightarrow cC \quad C \rightarrow \varepsilon$

  Quale linguaggio viene generato dalla grammatica ?

- Costruire gli alberi sintattici delle stringhe "113" e "2012" nella grammatica:
  - $S \rightarrow R \mid 0$
  - $R \rightarrow RC \mid R0 \mid C$
  - $C \rightarrow 1 \mid 2 \mid 3$

  Quale linguaggio viene generato dalla grammatica ?

Grammatiche ambigue

Nella grammatica ({E,ID}, {a,b,+,*}, $\mathbb{P}$, E)

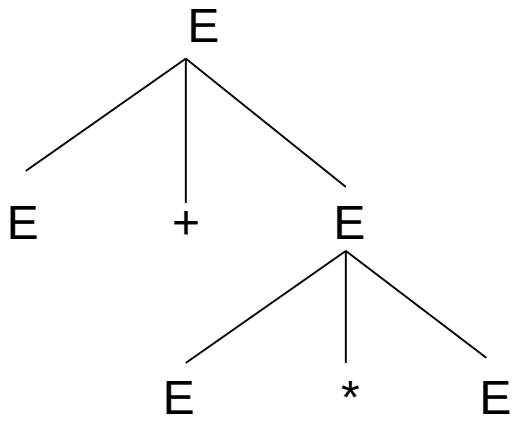
$\mathbb{P}$:

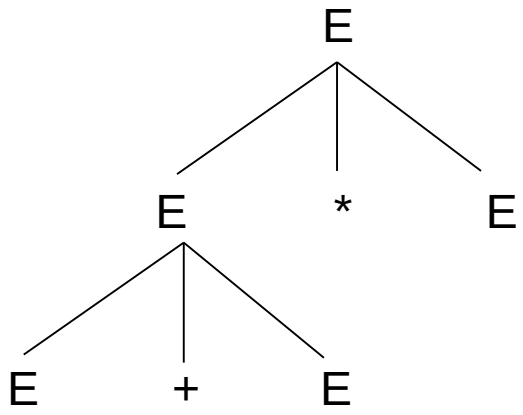
- $E \rightarrow ID$
- $E \rightarrow E + E$
- $E \rightarrow E * E$
- $ID \rightarrow a$
- $ID \rightarrow b$

La forma sentenziale E + E * E ha due derivazioni:

$$E \Rightarrow E + E \Rightarrow E + E * E$$
$$E \Rightarrow E * E \Rightarrow E + E * E$$

che danno due diversi alberi sintattici:





Grammatiche ambigue

Nella stessa grammatica la stringa a + b ha varie derivazioni:

$$E \Rightarrow E + E \Rightarrow ID + E \Rightarrow a + E \Rightarrow a + ID \Rightarrow a + b$$

$$E \Rightarrow E + E \Rightarrow E + ID \Rightarrow ID + ID \Rightarrow ID + b \Rightarrow a + b$$

Il loro albero sintattico è però lo stesso, e la struttura di a + b resta la stessa nelle due derivazioni.

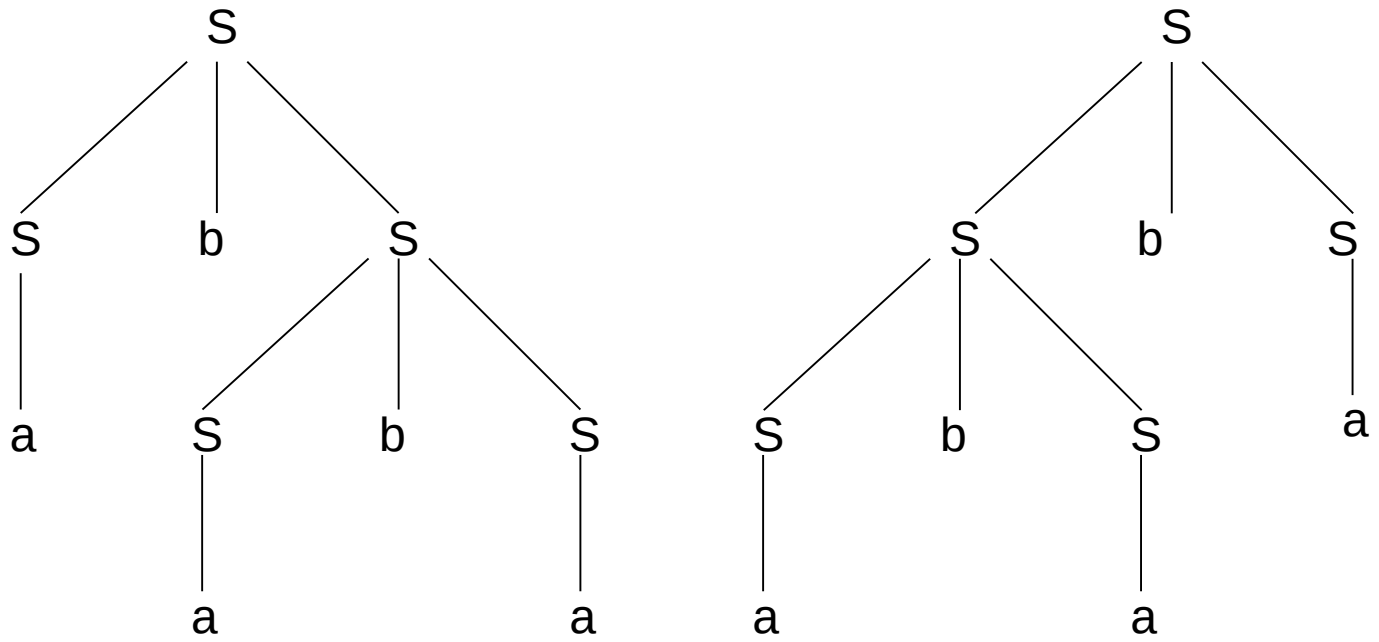
Mentre l'esistenza di diverse derivazioni non è in generale un problema, costituisce un problema l'esistenza di diversi alberi sintattici.

Grammatiche ambigue

Definizione

Una frase è *ambigua* se ha almeno due alberi sintattici distinti.

$G = (\{S\}, \{a, b\}, \{S \rightarrow SbS \mid a\}, S)$



Il grado di ambiguità di una frase è il numero dei suoi alberi sintattici distinti.

Grammatiche ambigue

Poiché ad ogni albero sintattico è associata un'unica derivazione a destra (e un'unica derivazione a sinistra), si può dire che una frase è ambigua se ha più di una derivazione a destra (o a sinistra).

Definizione

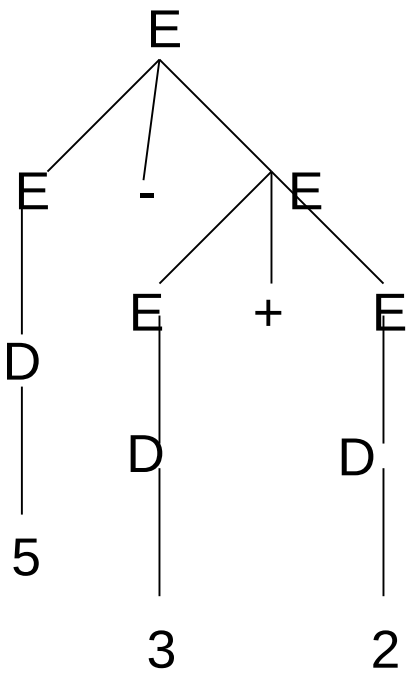
Una grammatica è *ambigua* se almeno una frase del linguaggio generato è ambigua.

Grammatiche ambigue

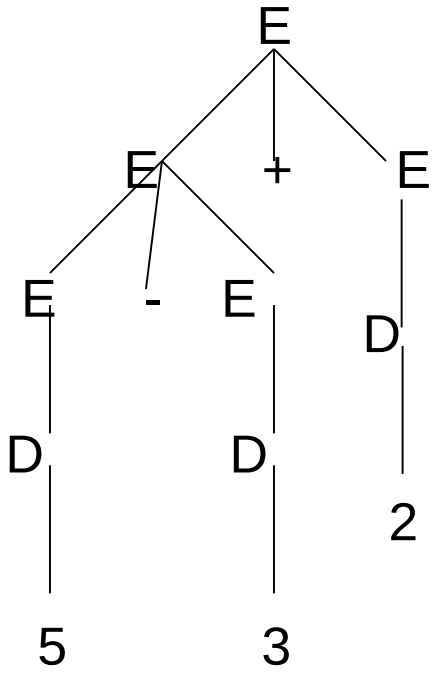
Esempio

$E \rightarrow E + E \mid E - E \mid D$

$D \rightarrow 0 \mid 1 \mid 2 \mid \ldots \mid 9$



5 – (3 + 2)



(5 – 3) + 2

Più di un albero  più di un significato

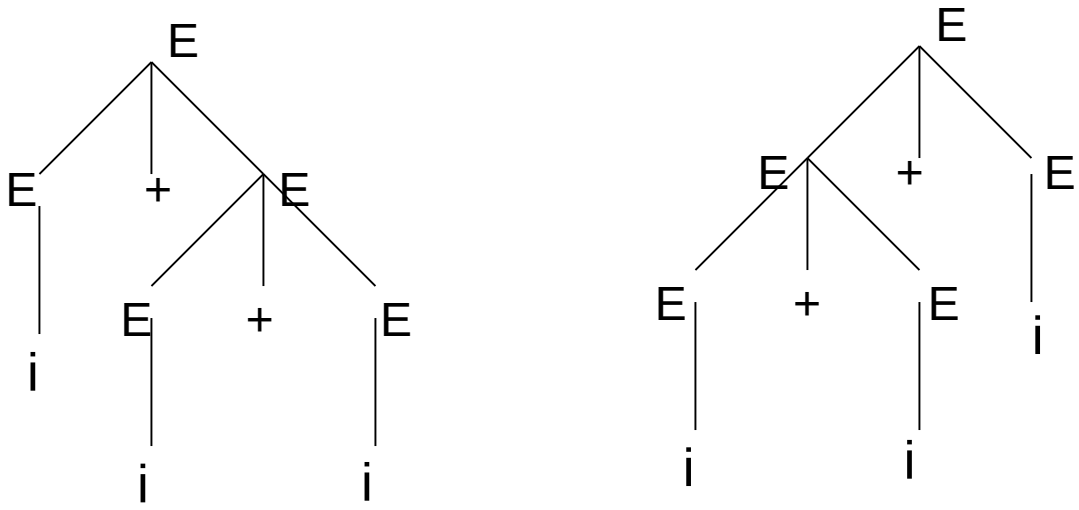
Eliminare l'ambiguità

Esempio

$E \rightarrow E + E \mid i$

$E \Rightarrow E + E \Rightarrow i + E \Rightarrow i + E + E \Rightarrow i + i + E \Rightarrow i + i + i$

$E \Rightarrow E + E \Rightarrow E + E + E \Rightarrow i + E + E \Rightarrow i + i + E \Rightarrow i + i + i$



Imporre un ordine: $E \rightarrow E + i \mid i$ (associativa a sinistra)
$E \rightarrow i + E \mid i$ (associativa a destra)

Eliminare l'ambiguità

Esempio

Frasi condizionali

$S \rightarrow$ *if* b *then* S *else* S | *if* b *then* S | a

Ruoli diversi

$S \rightarrow S_E \mid S_T$

$S_E \rightarrow$ *if* b *then* S_E *else* S_E | a

$S_T \rightarrow$ *if* b *then* S_E *else* S_T | *if* b *then* S

$S \Rightarrow S_T \Rightarrow$ *if* b *then* S $\Rightarrow$ *if* b *then* $S_E \Rightarrow$
$\Rightarrow$ *if* b *then if* b *then* S_E *else* $S_E \Rightarrow$
$\Rightarrow$ *if* b *then if* b *then* a *else* a

Marche finali:

$S \rightarrow$ *if* b *then* S *else* S *end_if* | *if* b *then* S *end_if* | a

Linguaggi inerentemente ambigui

Definizione

Un linguaggio è *inerentemente ambiguo* se tutte le grammatiche che lo generano sono ambigue.

Esempio:

$L = \{ a^n b^n c^m d^m \mid n \geq 1, m \geq 1\} \cup \{ a^n b^m c^m d^n \mid n \geq 1; m \geq 1\}$

Una grammatica per L ha come insieme di produzioni:

$S \rightarrow AB \mid C$

$A \rightarrow aAb \mid ab$ $\quad\quad$ $B \rightarrow cBd \mid cd$

$C \rightarrow aCd \mid aDd$ $\quad\quad$ $D \rightarrow bDc \mid bc$

Per la stringa aabbccdd ci sono due derivazioni a sinistra:

$S \Rightarrow AB \Rightarrow aAbB \Rightarrow aabbB \Rightarrow aabbcBd \Rightarrow aabbccdd$

e

$S \Rightarrow C \Rightarrow aCd \Rightarrow aaDdd \Rightarrow aabDcdd \Rightarrow aabbccdd$

Si può dimostrare che ogni grammatica per L si comporta come questa.
Il linguaggio L è quindi inerentemente ambiguo.

Linguaggi inerentemente ambigui

Un altro linguaggio inerentemente ambiguo:

$$\{a^i b^i c^j \mid i, j \geq 0\} \cup \{a^h b^k c^k \mid h, k \geq 0\} =$$

$$= \{a^i b^j c^k \mid i, j, k \geq 0 \quad \text{e} \quad (i = j \quad \text{oppure} \quad j = k)\}$$

$G = < \{S\}, \{a, b, c\}, \mathbb{P}, S >$

$\mathbb{P} = \{S \rightarrow BC \mid AD,$

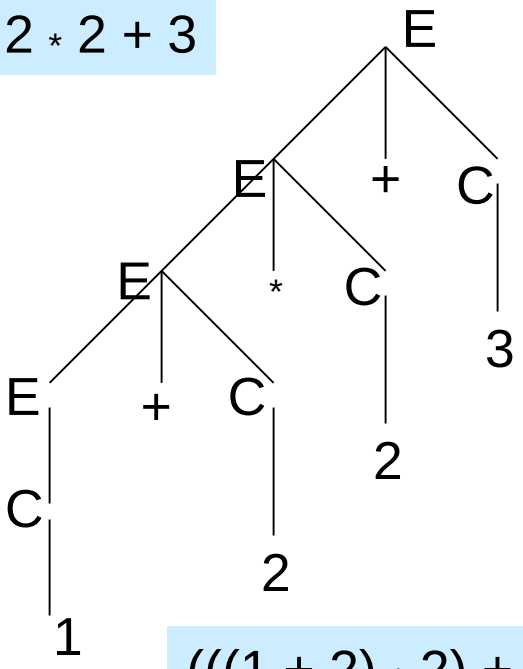
$B \rightarrow aBb \mid \varepsilon, \qquad C \rightarrow cC \mid \varepsilon,$

$A \rightarrow aA \mid \varepsilon, \qquad D \rightarrow bBc \mid \varepsilon\}$

Adeguatezza strutturale

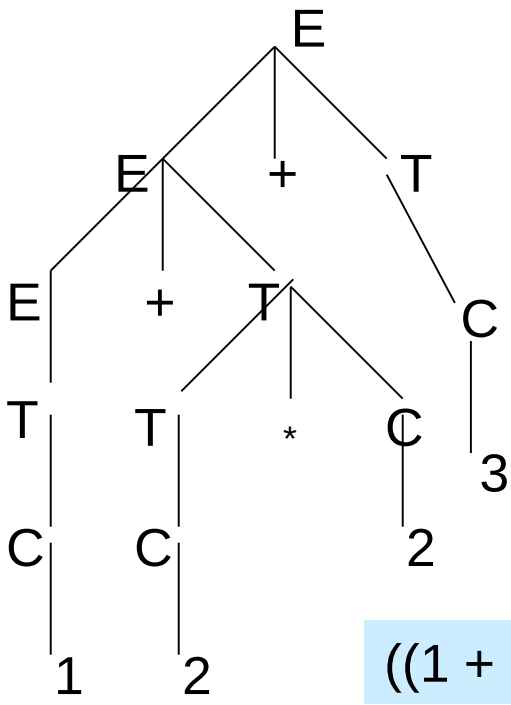
G_1 $E \rightarrow E + C \mid E * C \mid C$
$C \rightarrow 0 \mid 1 \mid \ldots \mid 9$

1 + 2 * 2 + 3



(((1 + 2) * 2) + 3)

G_2 $E \rightarrow E + T \mid T$
$T \rightarrow T * C \mid C$
$C \rightarrow 0 \mid 1 \mid \ldots \mid 9$



((1 + (2 * 2)) + 3)

Le due grammatiche sono non ambigue, ma solo la seconda è strutturalmente adeguata.

Adeguatezza strutturale

Una grammatica non ambigua e strutturalmente adeguata per le espressioni aritmetiche parentesizzate

Operatori: * e / associativi a sinistra
+ e - associativi a sinistra
* e / hanno precedenza su + e -

Si devono introdurre due non terminali per i due livelli di precedenza e uno per le unità di base: num e espressioni tra parentesi.

espr $\rightarrow$ espr + term | espr - term | term

term $\rightarrow$ term * factor | term / factor | factor

factor $\rightarrow$ num | (expr)

num $\rightarrow$ 0 | 1 | … | 9

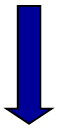
1 + 2 * 2 + 3

((1 + (2 * 2)) + 3)

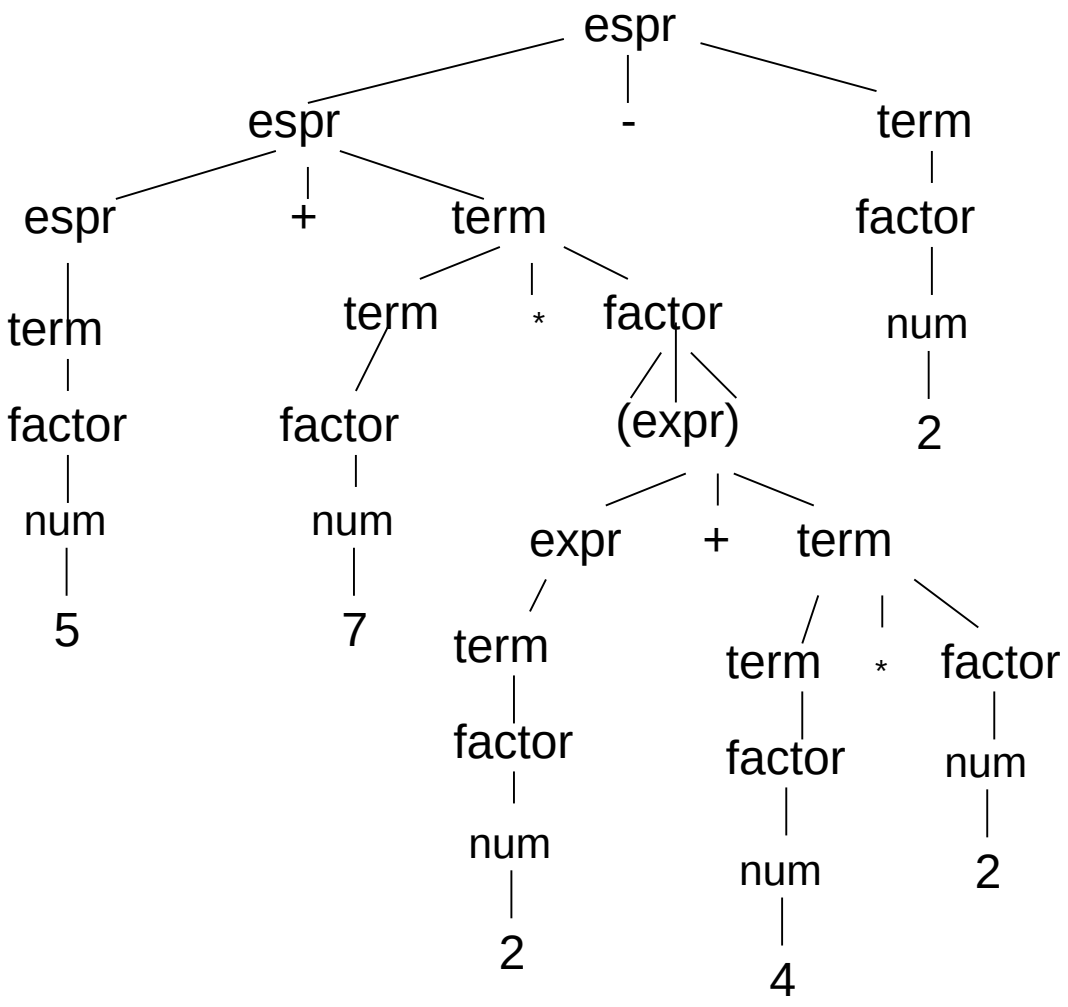
Adeguatezza strutturale

espr → espr + term | espr - term | term
term → term * factor | term / factor | factor
factor → num | (expr)
num → 0 | 1 | … | 9

5 + 7 * (2 + 4 * 2) - 2



((5 + (7 * (2 + (4 * 2)))) - 2)



Esercizi

- Dire se le seguenti grammatiche sono ambigue e disegnare gli alberi di derivazione per le stringhe specificate.

Insieme di produzioni	Stringa da derivare
P1: $S \rightarrow aSb \mid Sb \mid b$	aabbbb
P2: $E \rightarrow (E + T) \mid (E - T) \mid T$ $T \rightarrow T * num \mid T / num \mid num$	(((7 + 4*10) + 3*6) - 15/3)

- La seguente grammatica genera il linguaggio delle espressioni regolari sull'alfabeto {a, b}:

$$R \rightarrow \Phi \mid \varepsilon \mid a \mid b \mid R + R \mid R.R \mid R^* \mid (R)$$

Essa però non definisce le priorità tra gli operatori.
Scrivere una grammatica che generi il linguaggio delle espressioni regolari in cui * ha precedenza su . e . ha precedenza su +.