# Grammatiche libere dal contesto

# Costruzione di un albero sintattico

Sia $G = (V, T, P, S)$ una CFG. Un albero e' un **albero sintattico** per $G$ se:

1. Ogni nodo interno e' etichettato con una variabile in $V$.
2. Ogni foglia e' etichettata con un simbolo in $V \cup T \cup \{\epsilon\}$. Ogni foglia etichettata con $\epsilon$ e' l'unico figlio del suo genitore.
3. Se un nodo interno e' etichettato $A$, e i suoi figli (da sinistra a destra) sono etichettati

   $$X_1, X_2, \ldots, X_k,$$

   allora $A \to X_1 X_2 \ldots X_k \in P$.

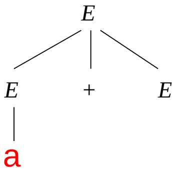
# Esempio

Nella grammnatica

1. $E \rightarrow \text{a}$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$

$\vdots$

il seguente e' un albero sintattico:



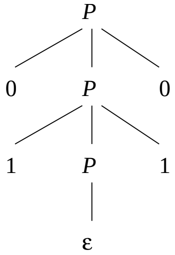

Questo albero sintattico mostra la derivazione $E \overset{*}{\rightarrow} \text{a} + E$

# Esempio

Nella grammatica

1. $P \to \epsilon$
2. $P \to 0$
3. $P \to 1$
4. $P \to 0P0$
5. $P \to 1P1$

il seguente e' un albero sintattico:




Mostra la derivazione $P \xrightarrow{*} 0110$.

# Il prodotto di un albero sintattico

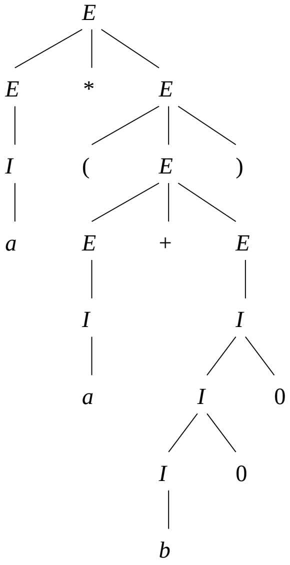
- Il **prodotto** di un albero sintattico e' la stringa di foglie da sinistra a destra.
- Sono importanti quegli alberi sintattici dove:
  1. Il prodotto e' una stringa terminale.
  2. La radice e' etichettata dal simbolo iniziale.
- L'insieme dei prodotti, formati da soli terminali, di questi alberi sintattici e' il linguaggio della grammatica.

# Esempio

Estendiamo la grammatica delle espressioni per arricchire il lingiaggio degli identificatri.
Consideriamo la grammatica $G = (\{E, I\}, T, P, E)$ dove $T = \{+, *, (, ), a, b, 0, 1\}$ e $P$ e' il seguente insieme di produzioni:

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$
5. $I \rightarrow a$
6. $I \rightarrow b$
7. $I \rightarrow Ia$
8. $I \rightarrow Ib$
9. $I \rightarrow I0$
10. $I \rightarrow I1$

# Esempio

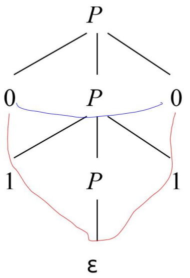


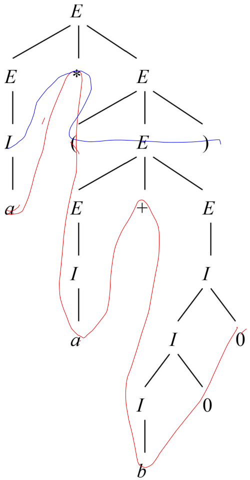

Il prodotto e' $a * (a + b00)$.

Sia $G = (V, T, P, S)$ una CFG, e $A \in V$. I seguenti sono equivalenti:

1. $A \xrightarrow{*} w$
2. $A \underset{lm}{\overset{*}{\rightarrow}} w$, e $A \underset{rm}{\overset{*}{\rightarrow}} w$
3. C'e' un albero sintattico di $G$ con radice $A$ e' prodotto $w$.

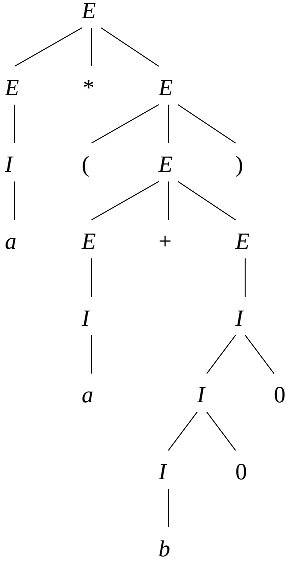
# Alberi sintattici e Forme sentenzali




In particolare ad *ogni frontiera interna* di un albero sintattico corrisponde una forma sentenziale. Le frontiere interne in blu corrispondono rispettivamente alle forme sentanziali 0P0 e I*(E).

Consideriamo l'albero:



Si puo' costruire una *derivazione sinistra* corrispondente espendendo le variabili secondo la lettura in preordine dei nodi dell'albero. Iniziamo con $E \underset{lm}{\rightarrow} E * E$ e espandiamo succesivamente la prima $E$ in $I$ e cosi' via:

$$
\begin{aligned}
& E \underset{lm}{\rightarrow} E * E \underset{lm}{\rightarrow} \\
& I * E \underset{lm}{\rightarrow} a * E \underset{lm}{\rightarrow} \\
& a * (E) \underset{lm}{\rightarrow} a * (E + E) \underset{lm}{\rightarrow} \\
& a * (I + E) \underset{lm}{\rightarrow} a * (a + E) \underset{lm}{\rightarrow} \\
& a * (a + I) \underset{lm}{\rightarrow} a * (a + I0) \underset{lm}{\rightarrow} \\
& a * (a + I00) \underset{lm}{\rightarrow} a * (a + b00)
\end{aligned}
$$

C'e una *corrispodenza biunivoca* tre alberi sintattici e derivazioni *leftmost*.

# Alberi sintattici e Derivazioni rightmost

Allo stesso modo da un albero sintattico si puo' costruire una derivazione *destra* corrispondente espendendo le variabili in ordine inverso.
Per lo stesso albero sintattico si ottiene:

$$
\begin{aligned}
& E \underset{rm}{\rightarrow} E * E \underset{rm}{\rightarrow} \\
& E * (E) \underset{rm}{\rightarrow} E * (E + E) \underset{rm}{\rightarrow} \\
& E * (E + I) \underset{rm}{\rightarrow} E * (E + I0) \underset{rm}{\rightarrow} \\
& E * (E + I00) \underset{rm}{\rightarrow} E * (E + b00) \underset{rm}{\rightarrow} \\
& E * (I + b00) \underset{rm}{\rightarrow} E * (a + b00) \underset{rm}{\rightarrow} \\
& I * (a + b00) \underset{rm}{\rightarrow} a * (a + b00)
\end{aligned}
$$

C'e una *corrispodenza biunivoca* anche tre alberi sintattici e derivazioni *rightmost*.

# Ambiguita' in Grammatiche e Linguaggi

Nella grammatica

1. $E \rightarrow I$
2. $E \rightarrow E + E$
3. $E \rightarrow E * E$
4. $E \rightarrow (E)$

$\cdots$

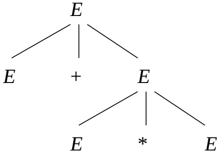
la forma sentenziale $E + E * E$ ha due derivazioni:

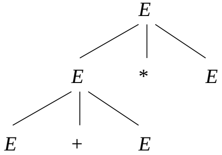
$$E \rightarrow E + E \rightarrow E + E * E$$

e

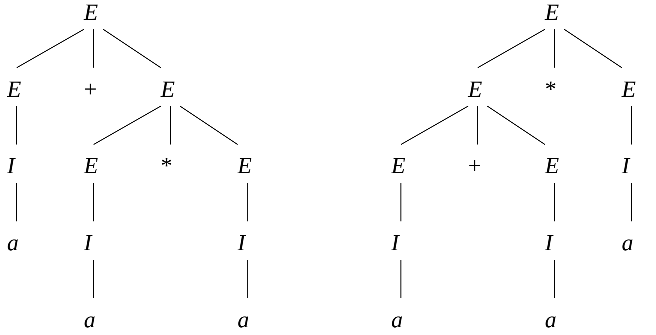
$$E \rightarrow E * E \rightarrow E + E * E$$

Questo ci da' due alberi sintattici:



**Definizione:** Sia $G = (V, T, P, S)$ una CFG. Diciamo che $G$ e' **ambigua** se esiste una stringa in $T^*$ che ha piu' di un albero sintattico.
Se ogni stringa in $L(G)$ ha al piu' un albero sintattico, $G$ e' detta **non ambigua**.
Esempio: La stringa terminale $a + a * a$ ha due alberi sintattici:



(a) (b)

L'esistenza di **varie derivazioni** di per se non e' pericolosa, e' l'esistenza di **vari alberi sintattici** che rovina la grammatica. Esempio: Nella stessa grammatica la stringa $a + b$ ha varie derivazioni:

$$E \rightarrow E + E \rightarrow I + E \rightarrow a + E \rightarrow a + I \rightarrow a + b$$

e

$$E \rightarrow E + E \rightarrow E + I \rightarrow I + I \rightarrow I + b \rightarrow a + b$$

Pero' il loro albero sintattico e' lo stesso, e la struttura di $a + b$ e' quindi non ambigua.

Ma per dire che una grammatica e' ambigua basta che ci sia anche solo <u>una</u> (o piu') stringa con due alberi di derivazione diversi. Come quella della slide precedente (e molte altre in questo caso).

# Rimuovere l'ambiguita' dalle grammatiche

- Buone notizie: a volte possiamo rimuovere l'ambiguita'
- Cattive notizie: non c'e' nessun algoritmo per farlo in modo sistematico
- Ancora cattive notizie: alcuni CFL hanno solo CFG ambigue
- Studiamo la grammatica

$$E \to I \mid E + E \mid E * E \mid (E)$$

$$I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1$$

  - Non c'e' precedenza tra * e $+$
  - Non c'e' raggruppamento di sequenze di operatori: $E + E + E$ e' inteso come $E + (E + E)$ o come $(E + E) + E$?

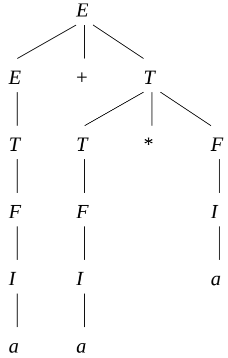
Soluzione: Introduciamo piu' variabili, ognuna che rappresenta espressioni con lo stesso grado di "forza di legamento"

1. Un *fattore* e' un'espressione che non puo' essere spezzata da un * o un + adiacente. I nostri fattori sono:
   1. Identificatori
   2. Un'espressione racchiusa tra parentesi.
2. Un *termine* e' un'espressione che non puo' essere spezzata da un +. Ad esempio, $a * b$ non puo' essere spezzata da +, perche' ad esempio $a1 + a * b$ e' (secondo le regole di precedenza) lo stesso di $a1 + (a * b)$, e $a * b + a1$ e' lo stesso di $(a * b) + a1$.
3. Il resto sono *espressioni*, cioe' possono essere spezzate con * o +.

Usiamo $F$ per i fattori, $T$ per i termini, e $E$ per le espressioni. Consideriamo la seguente grammatica:

$$
\begin{array}{ll}
1. & I \to a \mid b \mid Ia \mid Ib \mid I0 \mid I1 \\
2. & F \to I \mid (E) \\
3. & T \to F \mid T * F \\
4. & E \to T \mid E + T
\end{array}
$$

Ora l'unico albero sintattico per $a + a * a$ e':



Perche' la nuova grammatica non e' ambigua?

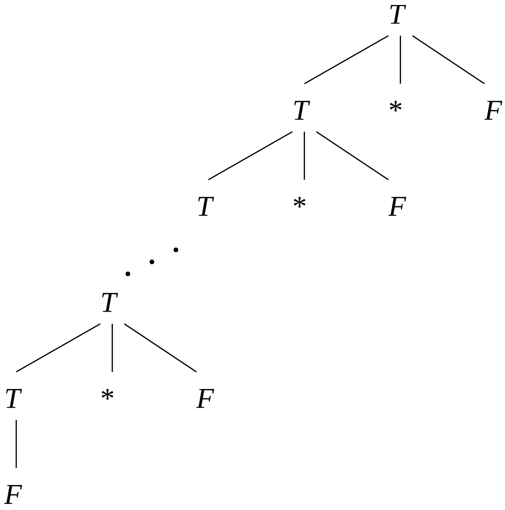
- Un fattore e' o un identificatore o $(E)$, per qualche espressione $E$.
- L'unico albero sintattico per una sequenza

$$f_1 * f_2 * \cdots * f_{n-1} * f_n$$

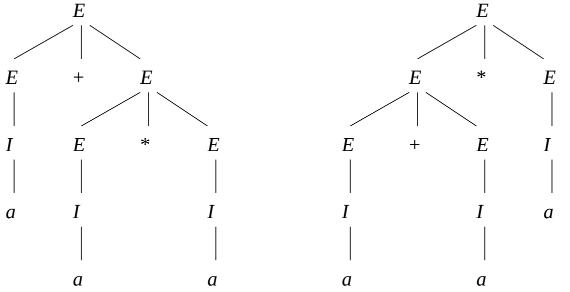
  di fattori e' quello che da' $f_1 * f_2 * \cdots * f_{n-1}$ come termine e $f_n$ come fattore, come nell'albero del prossimo lucido.
- Un'espressione e' una sequenza

$$t_1 + t_2 + \cdots + t_{n-1} + t_n$$

  di termini $t_i$. Puo' essere solo raggruppata con $t_1 + t_2 + \cdots + t_{n-1}$ come un'espressione e $t_n$ come un termine.


T
T * F
T * F
T
T * F
F

I due alberi sintattici per $a + a * a$




(a) (b)

danno luogo a due derivazioni:

$$E \underset{lm}{\rightarrow} E + E \underset{lm}{\rightarrow} I + E \underset{lm}{\rightarrow} a + E \underset{lm}{\rightarrow} a + E * E$$

$$\underset{lm}{\rightarrow} a + I * E \underset{lm}{\rightarrow} a + a * E \underset{lm}{\rightarrow} a + a * I \underset{lm}{\rightarrow} a + a * a$$

e

$$E \underset{lm}{\rightarrow} E * E \underset{lm}{\rightarrow} E + E * E \underset{lm}{\rightarrow} I + E * E \underset{lm}{\rightarrow} a + E * E \ldots\ldots$$

In generale:

- Un albero sintattico, ma molte derivazioni
- Molte derivazioni **a sinistra** implica molti alberi sintattici.
- Molte derivazioni **a destra** implica molti alberi sintattici.

**Teorema 5.29:** Data una CFG $G$, una stringa terminale $w$ ha due distinti alberi sintattici se e solo se $w$ ha due distinte derivazioni a sinistra dal simbolo iniziale.

**Prova:**

- *(Solo se.)* Se due alberi sintattici sono diversi, hanno un nodo dove sono state usate due diverse produzioni: $A \rightarrow X_1 X_2 \cdots X_k$ e $B \rightarrow Y_1 Y_2 \cdots Y_m$. Le corrispondenti derivazioni a sinistra useranno queste diverse produzioni e quindi saranno distinte.
- *(Se.)* Per come costruiamo un albero da una derivazione, e' chiaro che due derivazioni distinte generano due alberi distinti.

# Ambiguita' inerente

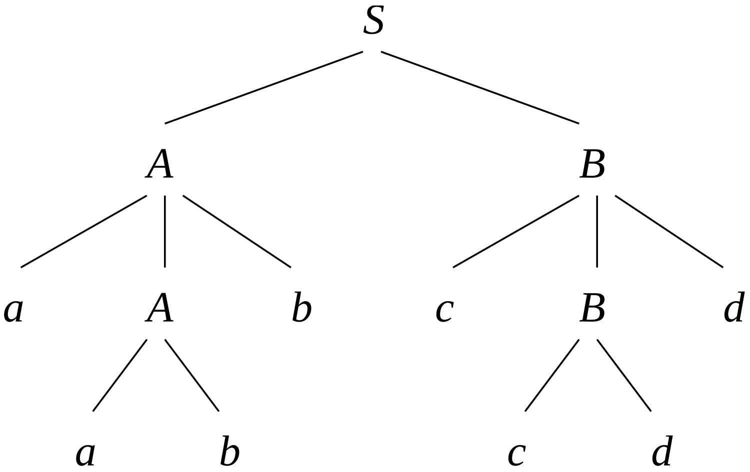
Un CFL $L$ e' **inerentemente ambiguo** se **tutte** le grammatiche per $L$ sono ambigue.
Esempio: Consideriamo $L =$

$$\{a^n b^n c^m d^m : n \geq 1, m \geq 1\} \cup \{a^n b^m c^m d^n : n \geq 1, m \geq 1\}.$$

Una grammatica per $L$ e'

$$\begin{aligned} S &\to AB \mid C \\ A &\to aAb \mid ab \\ B &\to cBd \mid cd \\ C &\to aCd \mid aDd \\ D &\to bDc \mid bc \end{aligned}$$

Guardiamo la struttura sintattica della stringa *aabbccdd*.



Vediamo che ci sono due derivazioni a sinistra:

$$S \underset{lm}{\rightarrow} AB \underset{lm}{\rightarrow} aAbB \underset{lm}{\rightarrow} aabbB \underset{lm}{\rightarrow} aabbcBd \underset{lm}{\rightarrow} aabbccdd$$

e

$$S \underset{lm}{\rightarrow} C \underset{lm}{\rightarrow} aCd \underset{lm}{\rightarrow} aaDdd \underset{lm}{\rightarrow} aabDcdd \underset{lm}{\rightarrow} aabbccdd$$

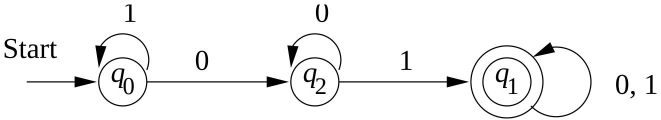
Puo' essere provato che **ogni** grammatica per $L$ si comporta come questa. Il linguaggio $L$ e' quindi inerentemente ambiguo.

- Un linguaggio regolare e' anche libero da contesto.
- Da una espressione regolare, o da un automa, si puo' ottenere una grammatica che genera lo stesso linguaggio.

# Da automa a grammatica

- Un simbolo non-terminale per ogni stato.
- Simbolo iniziale = stato iniziale.
- Per ogni transizione da stato $s$ a stato $p$ con simbolo $a$, produzione $S \rightarrow aP$.
- Se $p$ stato finale, allora produzione $P \rightarrow \epsilon$

Automa:




Grammatica:

$$Q_0 \to 1Q_0 \mid 0Q_2$$
$$Q_2 \to 0Q_2 \mid 1Q_1$$
$$Q_1 \to 0Q_1 \mid 1Q_1 \mid \epsilon$$

La stringa 1101 e' accettata dall'automa. Nella grammatica, ha la derivazione:

$$Q_0 \to 1Q_0 \to 11Q_0 \to 110Q_2 \to 1101Q_1 \to 1101$$