Derivazioni e Alberi

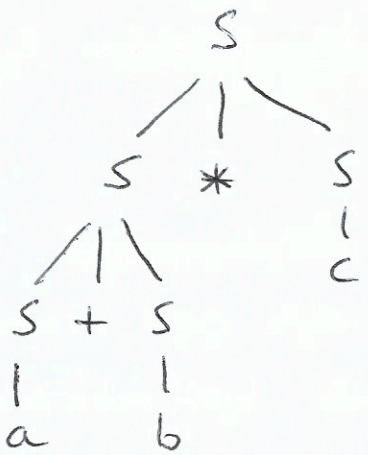
Consideriamo la grammatica

$$S \to a \mid b \mid c \mid S+S \mid S*S$$

- Consideriamo la derivazione

$$S \Rightarrow \underline{S}*S \Rightarrow \underline{S}+S*S \Rightarrow a+\underline{S}*S \Rightarrow a+b*S \Rightarrow a+b*c$$

  - leftmost: ad ogni passo riscriviamo il nonterminale più a sx
  - è possibile associarle un albero di derivazione



- Consideriamo un'altra derivazione per $a+b*c$

$$S \Rightarrow S*\underline{S} \Rightarrow \underline{S}*c \Rightarrow S+\underline{S}*c \Rightarrow \underline{S}+b*c \Rightarrow a+b*c$$

  - rightmost: ad ogni passo riscriviamo il nonterminale più a dx
  - genera lo stesso albero di derivazione

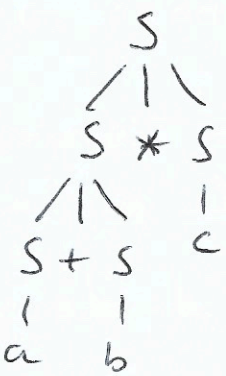
Osservazioni:

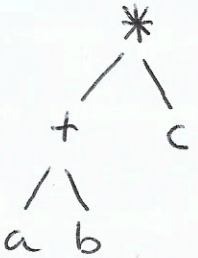
(1) Un albero di derivazione "riassume" tante derivazioni diverse, ma tutte "equivalenti" (nel senso che tutte generano lo stesso albero)

(2) Esiste una corrispondenza biunivoca tra derivazioni canoniche sinistre (o destre) e alberi di derivazione!

- dato un albero di derivazione, esiste una sola derivazione leftmost che lo genera (non ovvio!)
- dato un albero di derivazione, esiste una sola derivazione rightmost che lo genera
- data una derivazione leftmost (o rightmost), ad essa viene associato univocamente un albero di derivazione (ovvio)

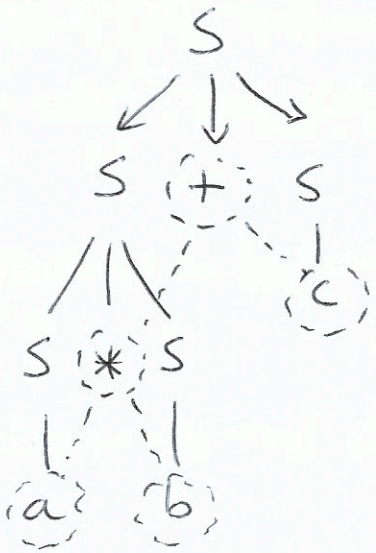
(3) L'albero di derivazione fornisce informazioni semantiche: "quali operandi per quali operatori"

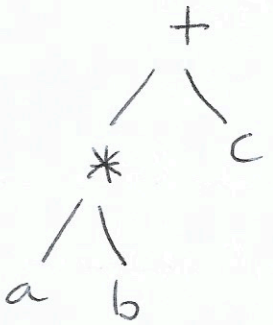




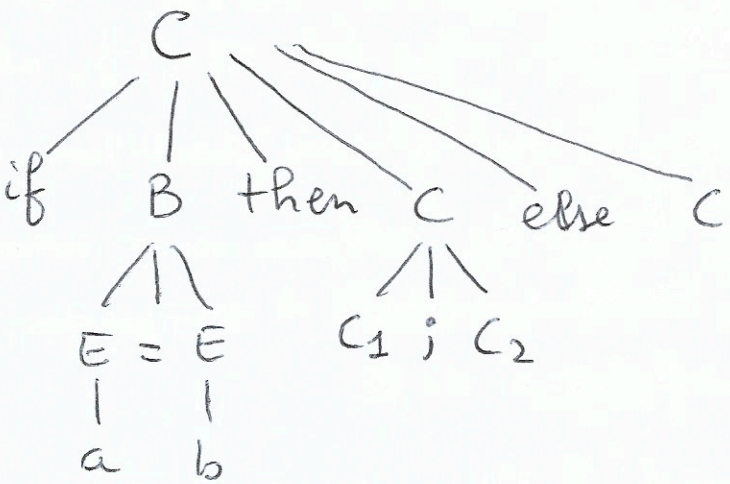
← albero sintattico ricavato dall'albero di derivazione

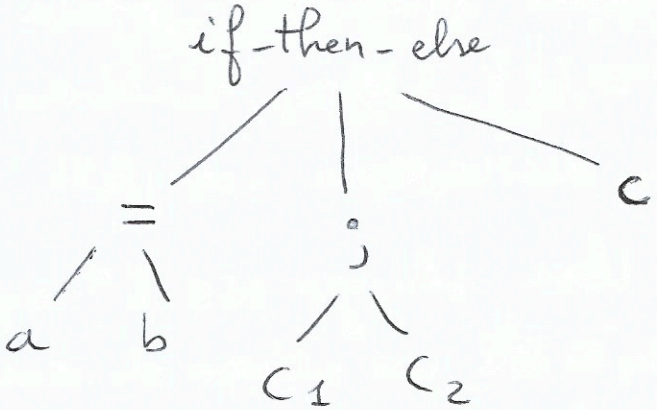
Alberi Sintattici da Alberi di Deriv.





- Albero di solo terminali dove i nodi interni sono etichettati da operatori e le foglie sono gli operandi
- Tipica rappresentazione interna usata dal compilatore per generare codice intermedio





Albero di Derivazione

Def: Data una grammatica libera $G=(NT, T, S, R)$, un albero di derivazione (o di parsing) è un albero ordinato in cui:

- ogni nodo è etichettato con un simbolo in $NT \cup \{\varepsilon\} \cup T$
- la radice è etichettata con S
- ogni nodo interno è etichettato con un simbolo in NT
- se il nodo n
  - ha etichetta $A \in NT$ e
  - i suoi figli sono nell'ordine $m_1, \dots m_k$ con etichetta $X_1, \dots X_k$ (in $NT \cup T$), allora

    $A \to X_1 \dots X_k$ è una produzione in R
- se il nodo n ha etichetta ε, allora n è una foglia, è figlio unico e, detto A suo padre, $A \to \varepsilon$ è una produzione di R
- se inoltre ogni nodo foglia è etichettato su $T \cup \{\varepsilon\}$, allora l'albero corrisponde ad una derivazione completa.

Teorema: Una stringa $w \in T^*$ appartiene a $L(G)$ sse ammette un albero di derivazione completo (le cui foglie, lette da sx a dx diano la stringa w)

cioè visita in ordine anticipato tralasciando i nonterminali

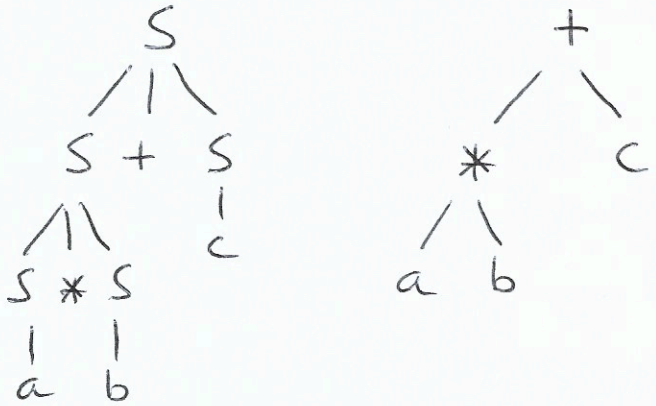
Ambiguità

- Consideriamo la grammatica

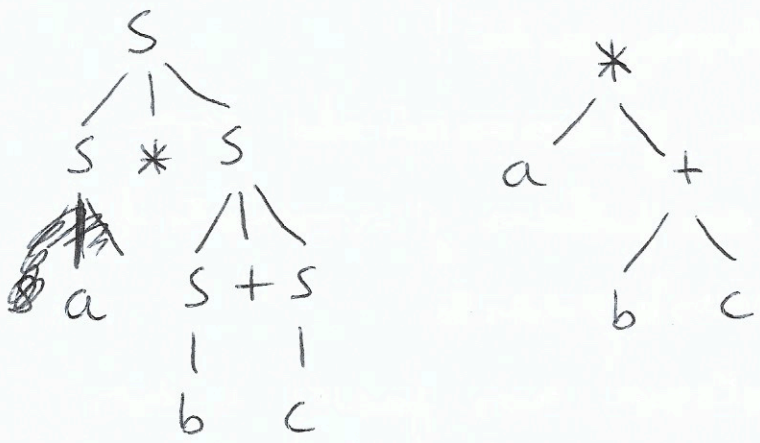
  $S \to a \mid b \mid c \mid S+S \mid S*S$

- Due diverse derivazioni leftmost per $a*b+c$

(1) $S \Rightarrow \underline{S}+S \Rightarrow \underline{S}*S+S \Rightarrow a*\underline{S}+S \Rightarrow a*b+\underline{S} \Rightarrow a*b+c$

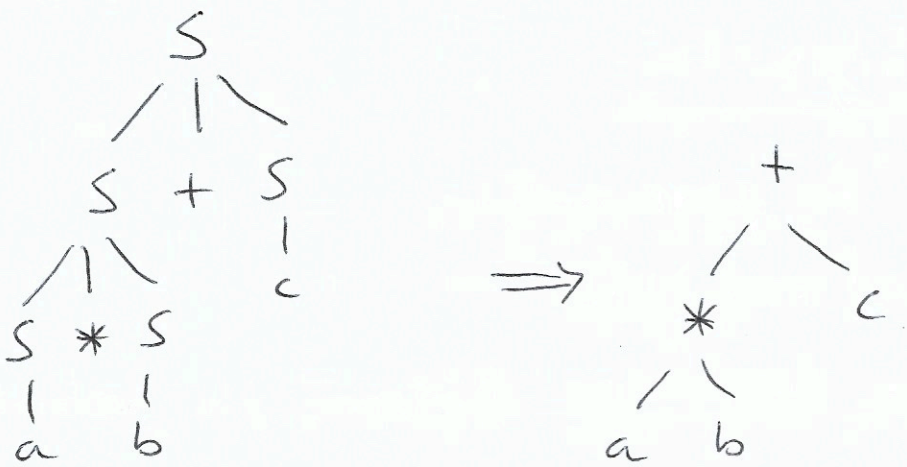


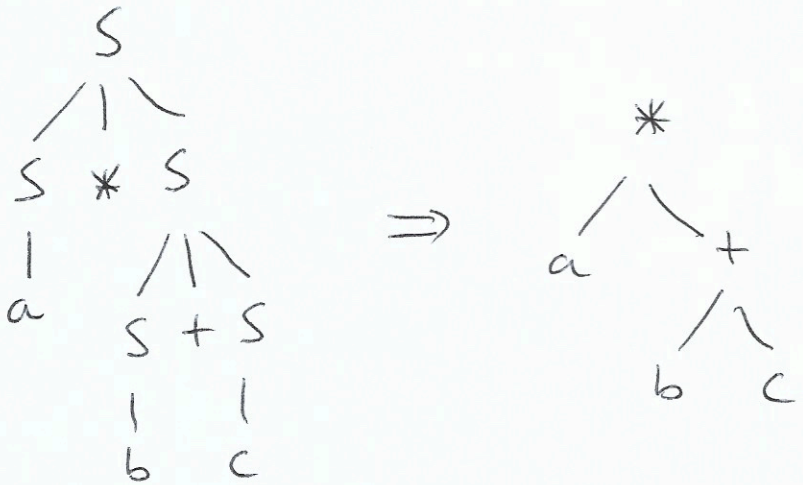
(2) $S \Rightarrow \underline{S}*S \Rightarrow a*\underline{S} \Rightarrow a*\underline{S}+S \Rightarrow a*b+\underline{S} \Rightarrow a*b+c$



Per questa grammatica, la stringa $a*b+c$ ha più di un albero di derivazione $\Rightarrow$ la grammatica è ambigua

(Inutilizzabile per dare semantica a $a*b+c$)





- Poiché tutti e due questi alberi sintattici sono derivabili per a*b+c, la semantica non è univoca! Ambiguità!
- Bisogna utilizzare grammatiche non ambigue, o manipolare grammatiche ambigue per disambiguarle
  (problema non facile e non sempre risolvibile)

Grammatica Ambigua

Def: Una grammatica libera G è ambigua se $\exists w \in L(G)$ che ammette più alberi di derivazione

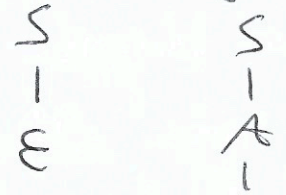
Def: Un linguaggio L è ambiguo se tutte le grammatiche G, tali che $L(G)=L$, sono ambigue

---

Alcune grammatiche possono essere manipolate in modo da
- rimuovere l'ambiguità, e
- generare lo stesso linguaggio

Esempio

$$\left.\begin{array}{l} S \to A \mid \varepsilon \\ A \to \varepsilon \end{array}\right] \quad G \text{ è ambigua perché}$$



Tuttavia $\exists\, G'$ non ambigua tale che $L(G) = L(G')$

$$G' \left[S \to \varepsilon \right.$$

Esistono grammatiche (patologiche) dalle quali non è possibile rimuovere l'ambiguità.

$\Rightarrow$ il linguaggio generato da queste grammatiche è ambiguo!

Esempio di un ling. ambiguo

Sappiamo che $L_1 = \{a^n b^n \mid n \geq 1\}$ è libero.
Non è difficile immaginare che

$L_2 = \{a^n b^n c^m d^m \mid n, m \geq 1\}$ sia libero non ambiguo

ed anche

$L_3 = \{a^n b^m c^m d^n \mid n, m \geq 1\}$ sia libero. non ambiguo

È facile verificare che

$L = L_2 \cup L_3$ è libero (perché l'unione di 2 ling. liberi è un ling. libero)

Tuttavia si può dimostrare che L è ambiguo!
Infatti, tutte le stringhe del tipo $a^n b^n c^n d^n$ hanno doppia derivazione

- una attraverso le produzioni che generano il ling. L_2
- e un'altra attraverso le produzioni che generano il ling. L_3

(dimostrazione tecnicamente difficile)

N.B. G_1 $S_1 \to a S_1 b \mid ab$

G_2 $S_2 \to S_1 S_1'$ $S_1' \to c S_1' d \mid cd$

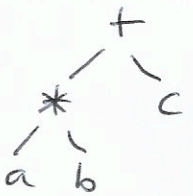
G_3 $S_3 \to a S_3 d \mid a S_3' d$ $S_3' \to b S_3' c \mid bc$

G $S_4 \to S_2 \mid S_3$

Rimuovere l'ambiguità (1)

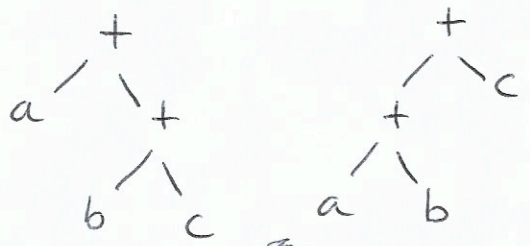
$S \to a \mid b \mid c \mid S+S \mid S*S$ è ambigua!

Problemi: (1) precedenza del * rispetto al +
in modo che a*b+c sia interpretato come



(2) associatività del + e del *

a+b+c



Bisogna scegliere se associare a dx o a sx

associatività a sx

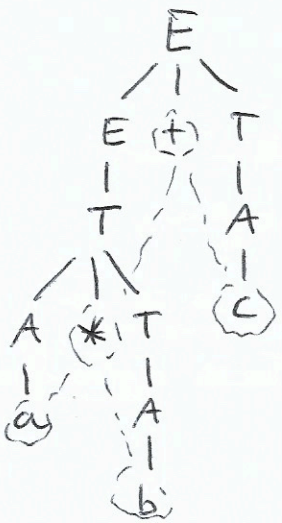
$E \to E+T \mid T$

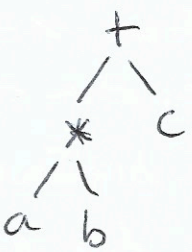
$T \to A*T \mid A$

$A \to a \mid b \mid c$

associatività a dx

* ha la precedenza sul +, perché è più interno nell'albero generato

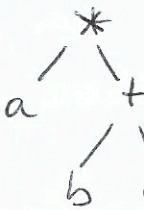




$L(S) = L(E)$

⇒ la nuova gr. è non ambigua ed equivalente

ma l'albero



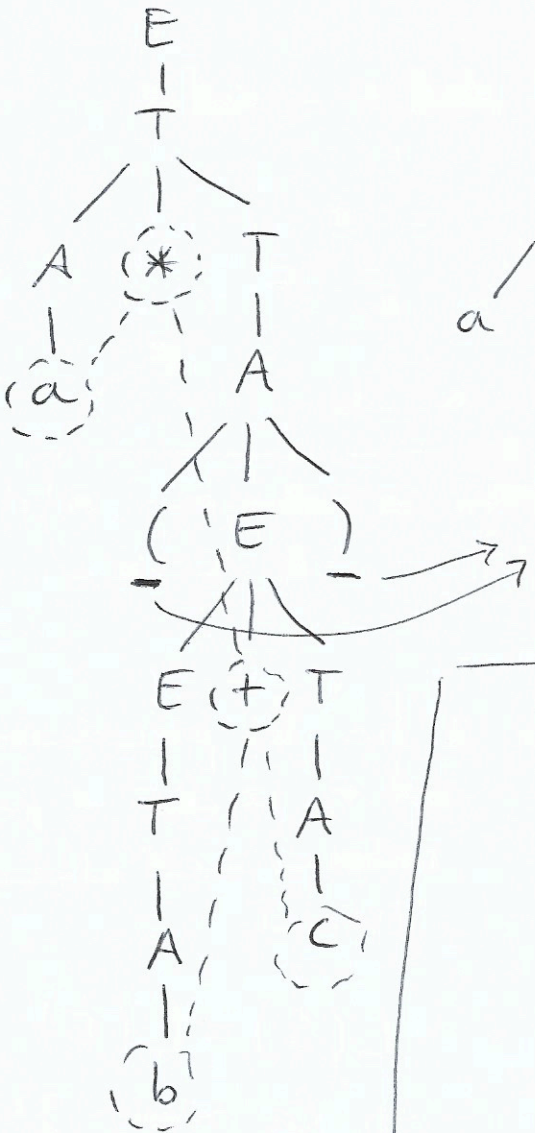
non è generabile!

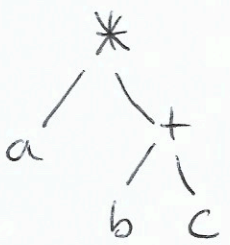
Per ottenere (quasi) lo stesso linguaggio, ma sicuramente gli stessi alberi sintattici, modifichiamo la grammatica così:

$$E \to E+T \mid T$$
$$T \to A*T \mid A$$
$$A \to a \mid b \mid c \mid (E)$$

nuovi terminali ausiliari, usati per disambiguare

$a*(b+c)$





albero sintattico astratto da sintassi "concreta" che prevede anche terminali ausiliari "(" e ")" semanticamente irrilevanti.

zucchero sintattico!

Sintassi Concreta:

grammatica non ambigua che fa uso di zucchero sintattico (come per E sopra)

Sintassi Astratta:

grammatica semplice ed intuitiva, ma ambigua.

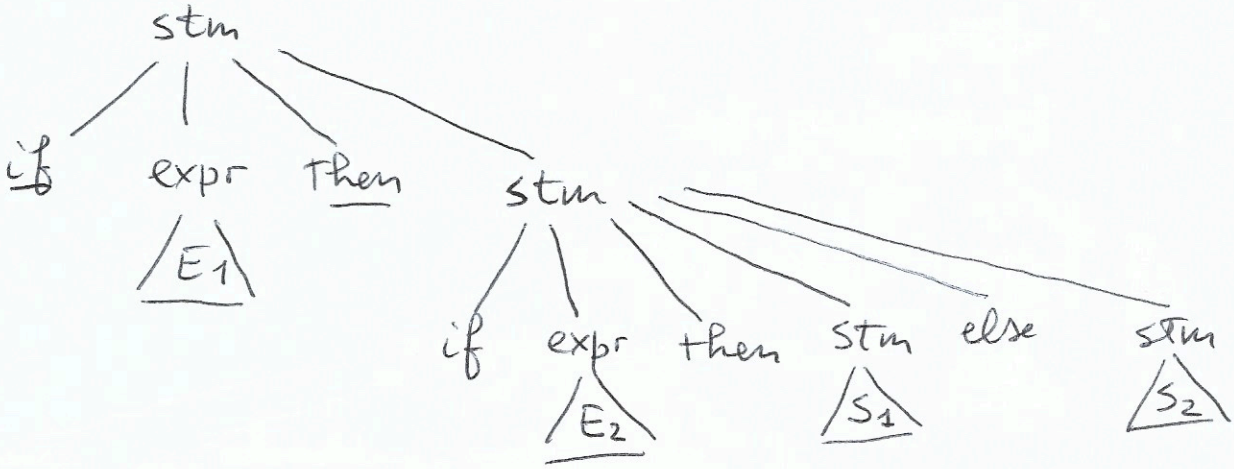
Dall'albero di derivazione da sintassi concreta, estraiamo un albero sintattico da sintassi astratta, detto albero sintattico astratto.

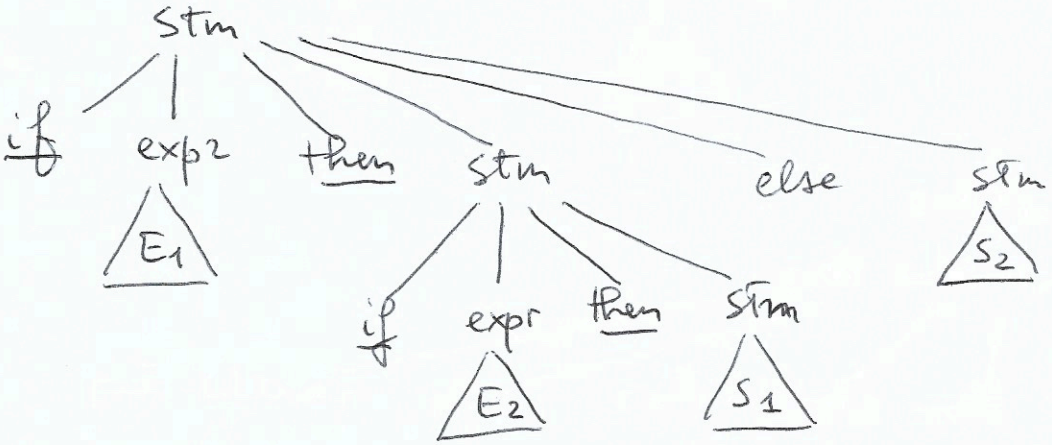
Rimuovere l'ambiguità (2)

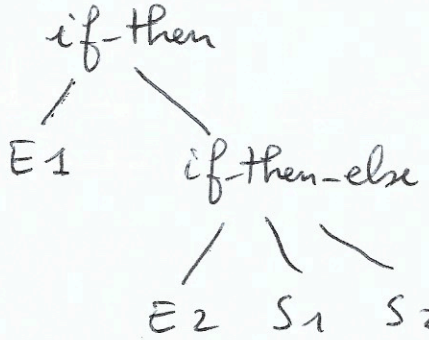
Consideriamo questa porzione di gram. per comandi

<stm> ::= if <expr> then <stm> |
if <expr> then <stm> else <stm> | ...

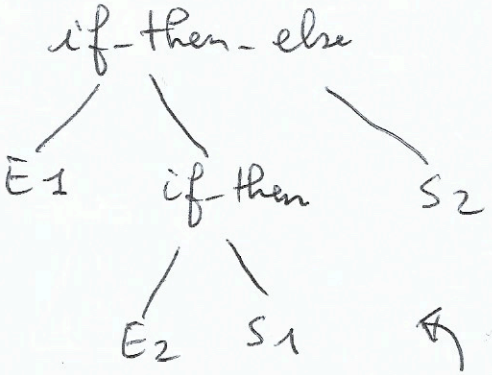
È ambigua. Infatti if E_1 then if E_2 then S_1 else S_2 ammette due alberi di derivazione







(interpretazione usuale)



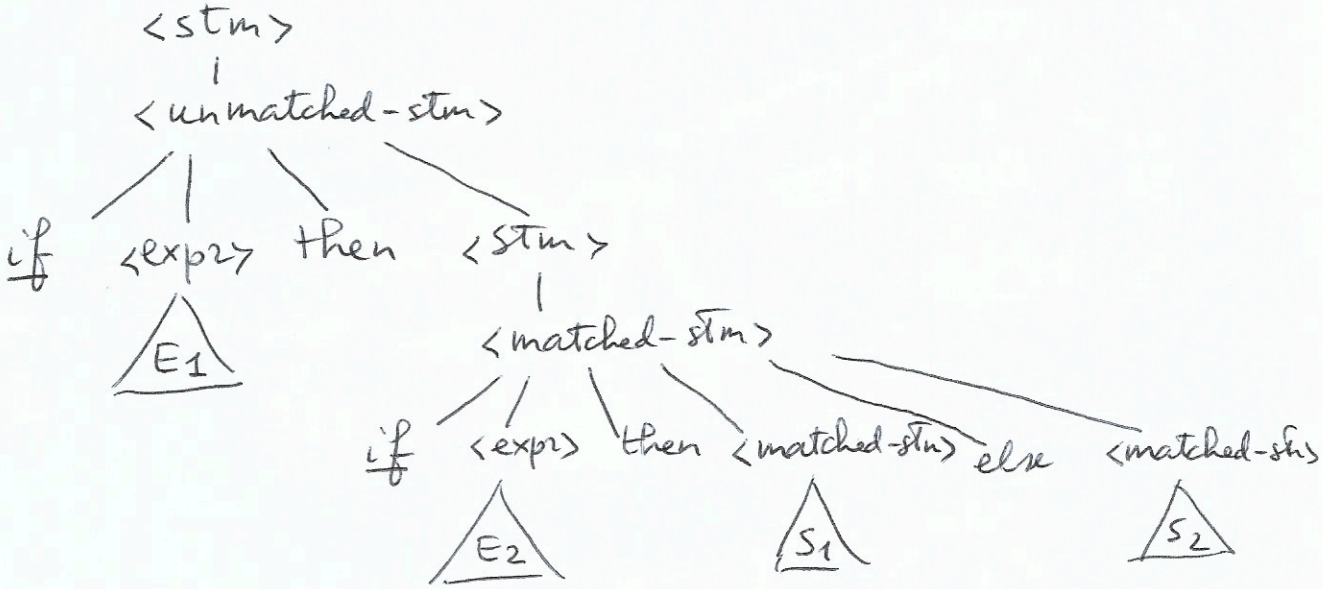
(albero "errato")

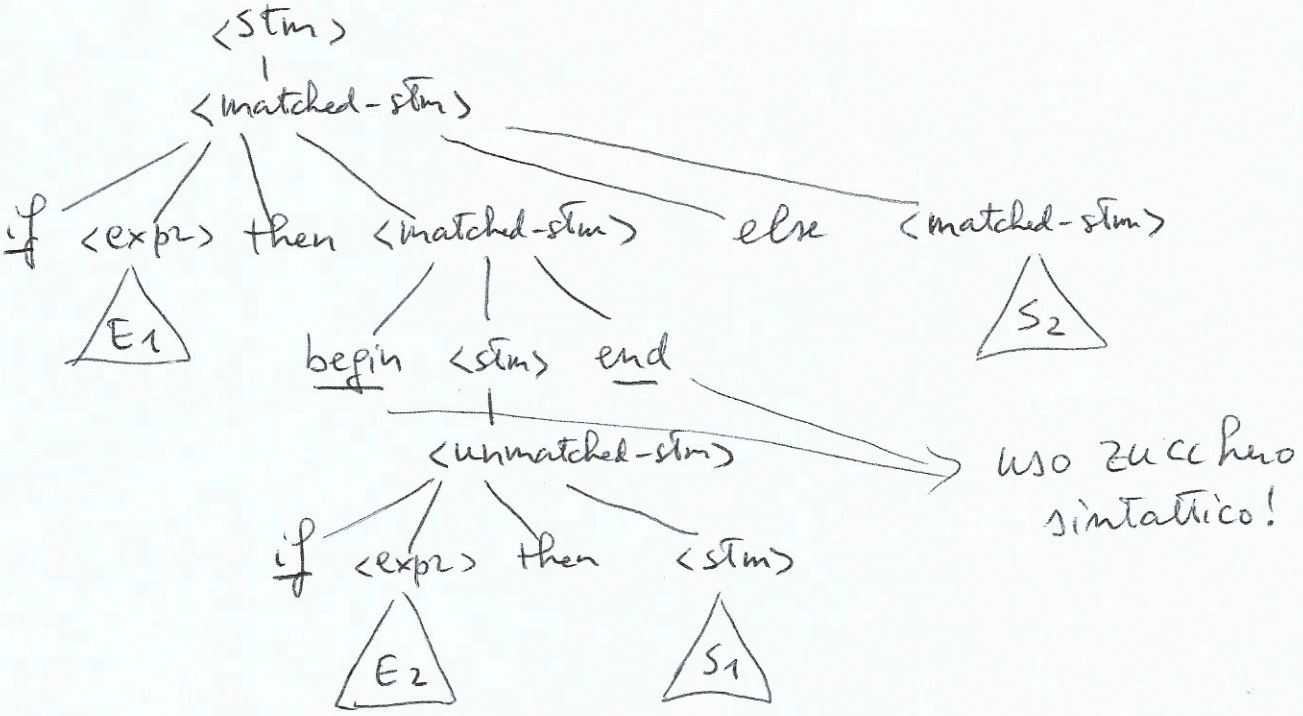
Grammatica Concreta

<stm> ::= <matched-stm> | <unmatched-stm>

<matched-stm> ::= if <expr> then <matched-stm>
else <matched-stm> | ... | begin <stm> end

<unmatched-stm> ::= if <expr> then <stm> |
if <expr> then <matched-stm>
else <unmatched-stm> | ...

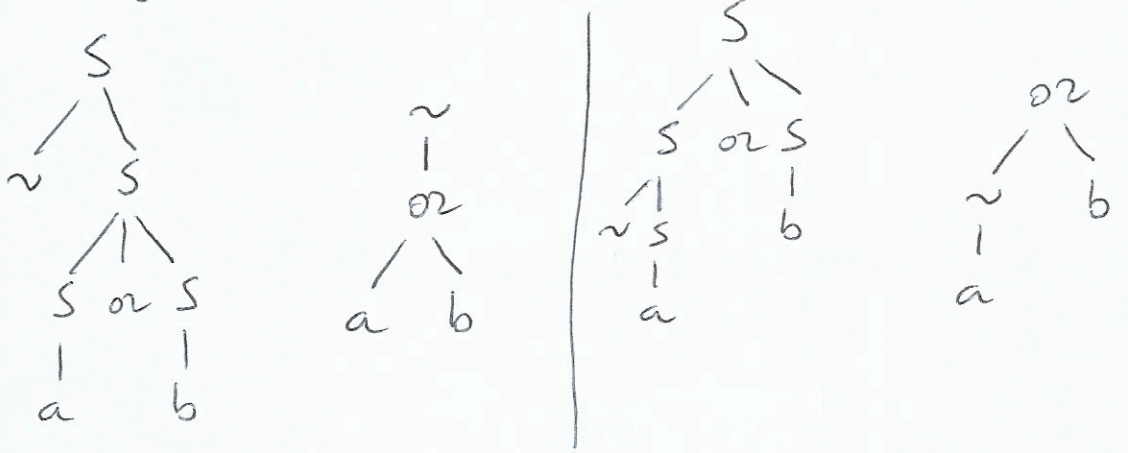




Esercizio : Espressioni booleane

$S \to a \mid b \mid \sim S \mid S \text{ or } S$ (sintassi astratta)

1) È ambigua? Sì



2) Grammatica ~~non~~ ambigua equivalente?

$B \to B \text{ or } C \mid C$

$C \to \sim C \mid A$

$A \to a \mid b$

- ~ lega più fortemente dell'or
- or associativo a SX

3) La nuova grammatica genera l'albero

? NO

4) Modificare la grammatica per generare quell'albero

Sintassi Concreta

$B \to B \text{ or } C \mid C$

$C \to \sim C \mid A$

$A \to a \mid b \mid (B)$