

Derivazioni e Alberi

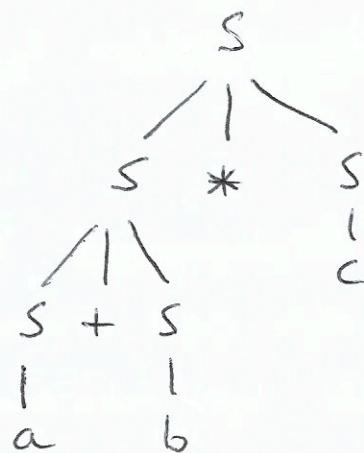
Consideriamo la grammatica

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

- 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 associare un albero di derivazione



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

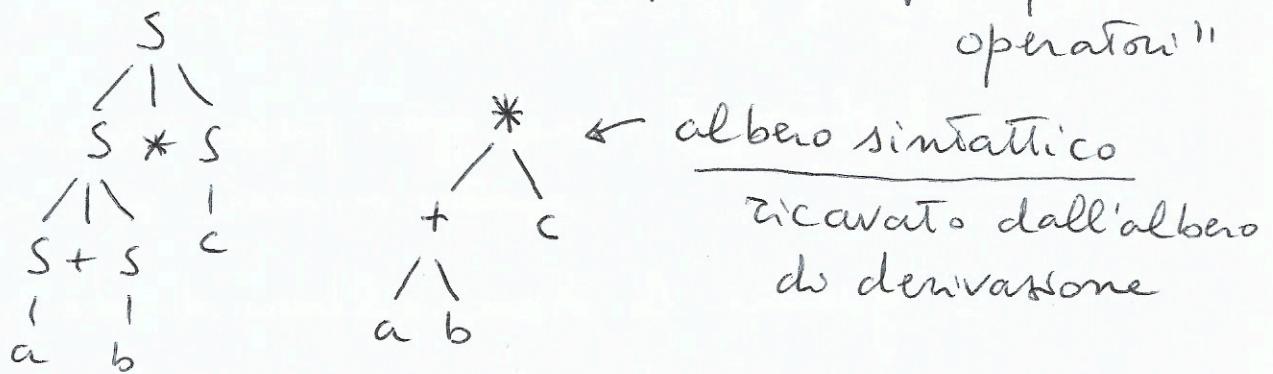
$$S \Rightarrow S*\underline{S} \Rightarrow \underline{S}*c \Rightarrow S+\underline{S*c} \Rightarrow \underline{S+b*c} \Rightarrow a+b*c$$

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

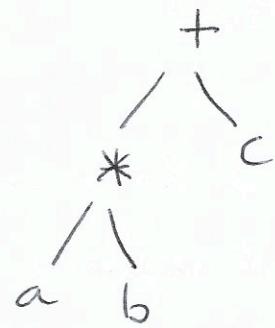
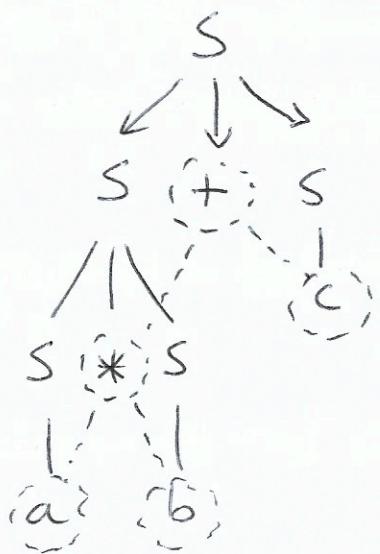
Osservazioni:

- (1) Un albero di derivazione "assume" 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"

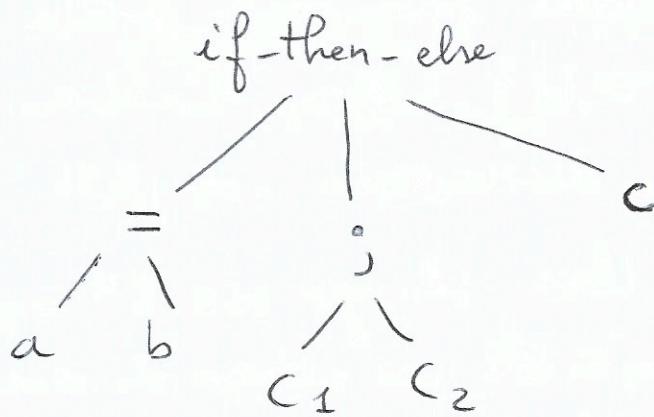
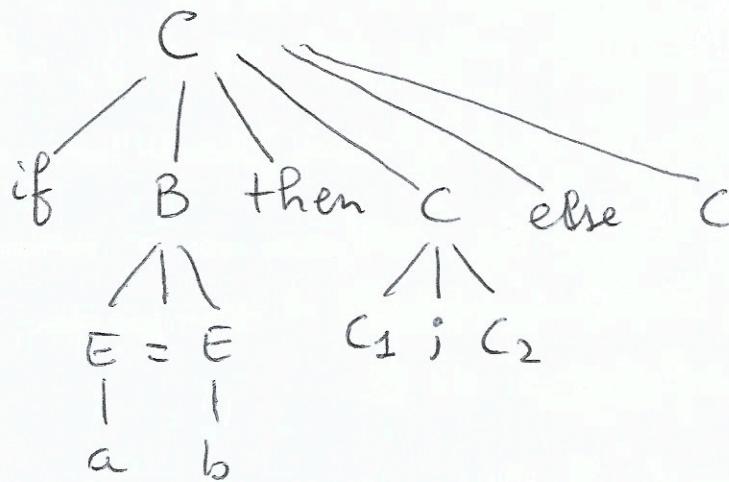


Alberi Sintattici da Alberi di Denr. (34)



- 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

(35)

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 $NT \cup \{S\}$
- 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
$$A \rightarrow x_1 \dots x_k$$
 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,
$$A \rightarrow \epsilon$$
 è una produzione di R
- se molte ognimodo foglia è etichettato su $T \cup \{\epsilon\}$, allora l'albero corrisponde ad una derivazione completa.

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

cisè visita in ordine anticipato trascurando i non terminali

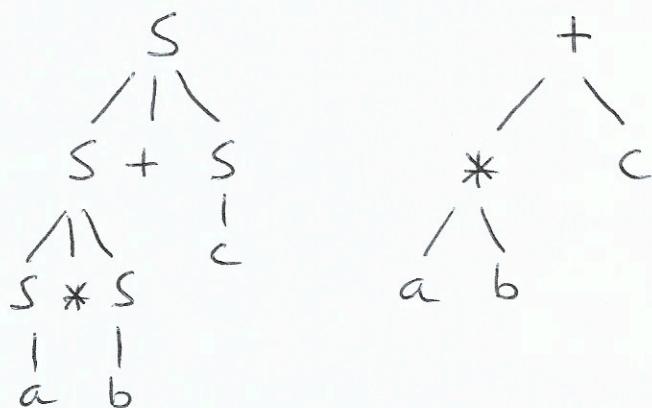
Ambiguità

- Consideriamo la grammatica

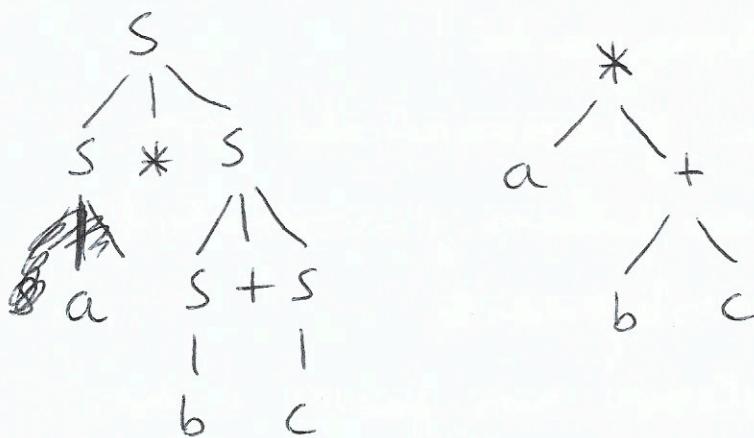
$$S \rightarrow a \mid b \mid c \mid S+S \mid S*S$$

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

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

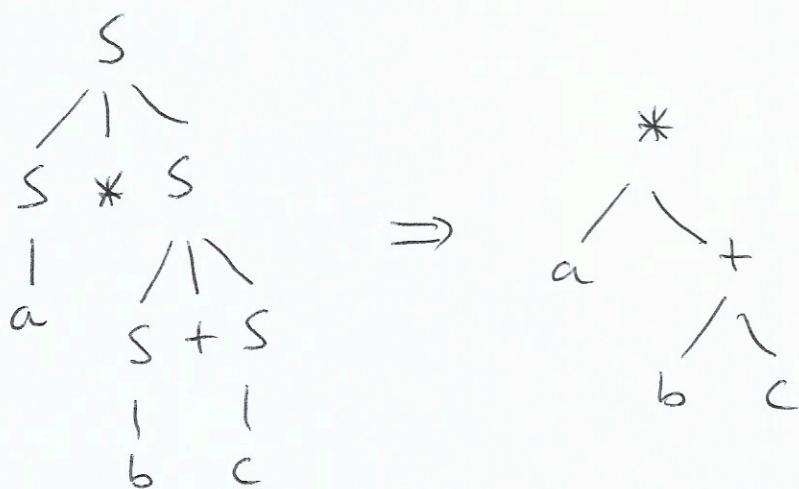
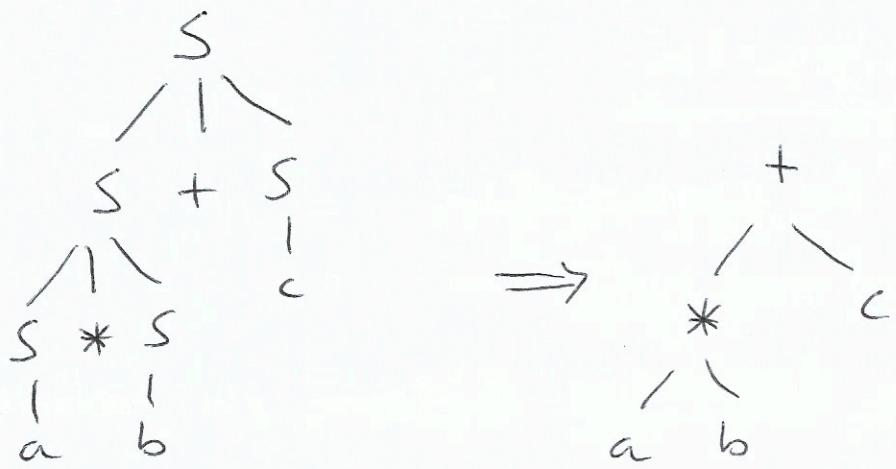


$$(2) \quad S \Rightarrow \underline{S*S} \Rightarrow a*\underline{S} \Rightarrow a*\underline{S+S} \Rightarrow a*b+\underline{S} \Rightarrow a*b+c$$



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 disambiguare
(problema non facile e non sempre risolvibile)

Grammatica Ambigua

(38)

Def: Una grammatica libera G è ambigua se $\exists \text{WL}(G)$ 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

$$\begin{array}{l} S \rightarrow A | \epsilon \\ A \rightarrow \epsilon \end{array} \quad] \quad G \text{ è ambigua perché}$$

$\begin{array}{ccc} S & & S \\ | & & | \\ E & & A \\ | & & | \\ E & & A \end{array}$

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

$$G' [S \rightarrow \epsilon$$

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

\Rightarrow il linguaggio generato da queste grammatiche è ambiguo!

Esempio di un lmp. ambiguo

(39)

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 lmp. liberi
è un lmp. 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 lmp. L_2
- e un'altra attraverso le produzioni che generano il lmp. L_3

(dimostrazione tecnicamente difficile)

N.B. $G_1 \quad S_1 \rightarrow aS_1b \mid ab$

$G_2 \quad S_2 \rightarrow S_1S_1' \quad S_1' \rightarrow cS_1'd \mid cd$

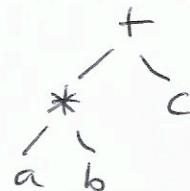
$G_3 \quad S_3 \rightarrow aS_3d \mid aS_3'd \quad S_3' \rightarrow bS_3'c \mid bc$

$G \quad S_4 \rightarrow S_2 \mid S_3$

Rimuovere l'ambiguità (1)

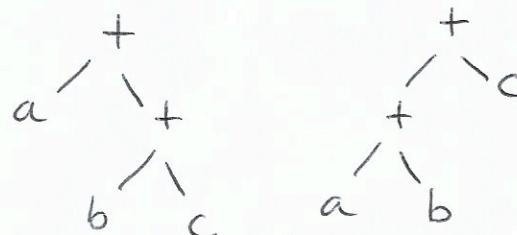
$S \rightarrow a \mid b \mid c \mid S+S \mid S*S$ è ambigua!

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

$E \rightarrow (\textcircled{E})^+ T \mid T$

$T \rightarrow A * (\textcircled{T}) \mid A$

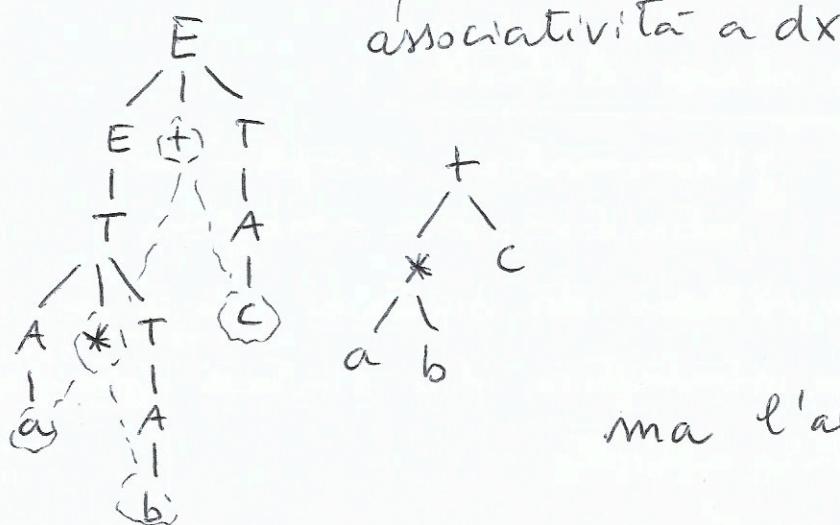
$A \rightarrow a \mid b \mid c$

associatività a sx

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

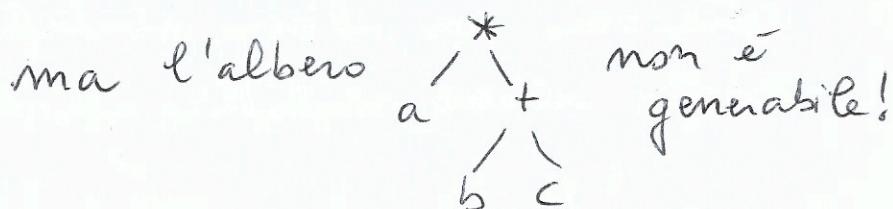
$$L(S) = L(E)$$

\Rightarrow la nuova gr.
è non ambigua
ed equivalente



ma l'albero

non è
generabile!



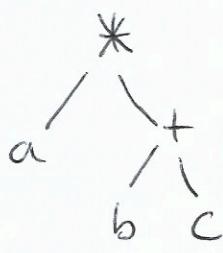
Per ottenere (quasi) lo stesso linguaggio,⁽⁴¹⁾
ma sicuramente gli stessi alberi sintattici,
modifichiamo la grammatica così:

$$E \rightarrow E + T \mid T$$

$$T \rightarrow A * T \mid A$$

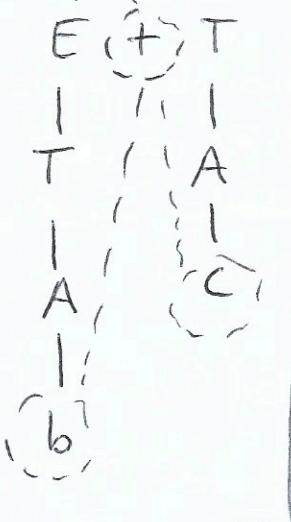
$$A \rightarrow a \mid b \mid c \mid (E)$$

$$a * (b + c)$$



albero sintattico astratto
da sintassi "concreta" che
prevede anche terminali
ausiliari ("(" e ")" semantica-
mente 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.

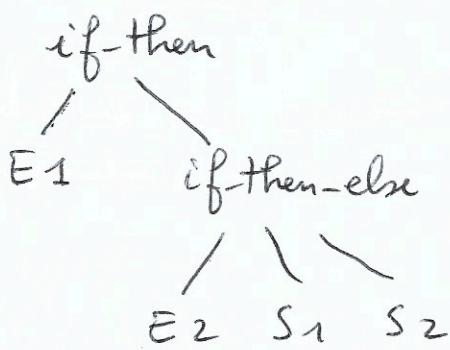
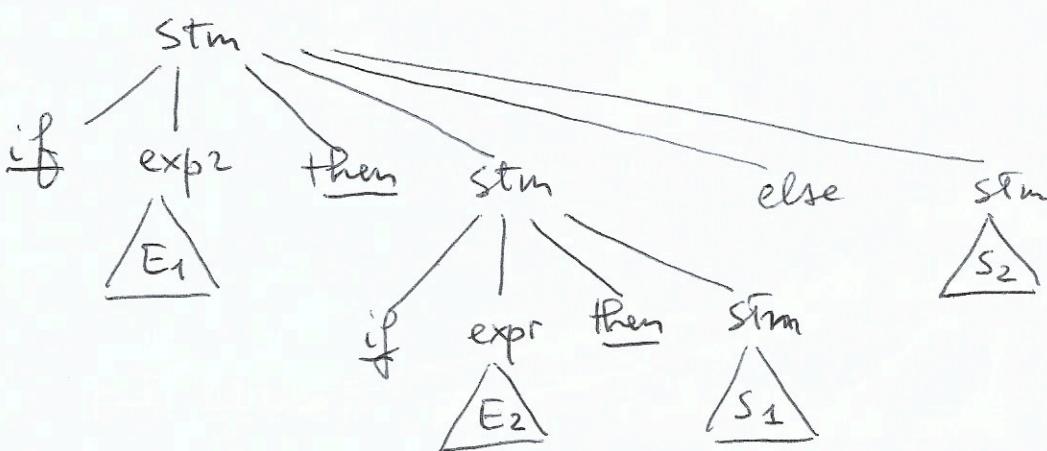
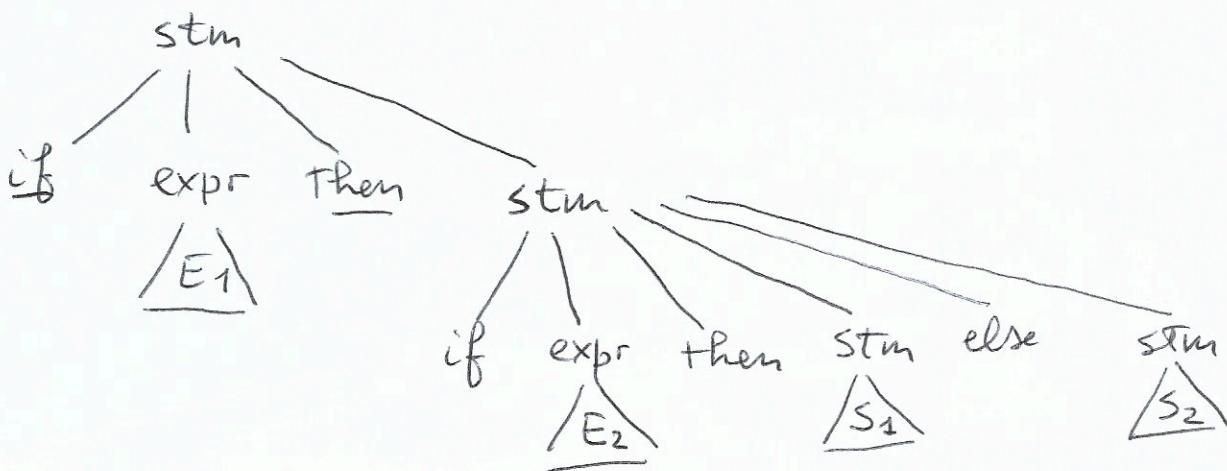
Rimuovere l'ambiguità (2)

(42)

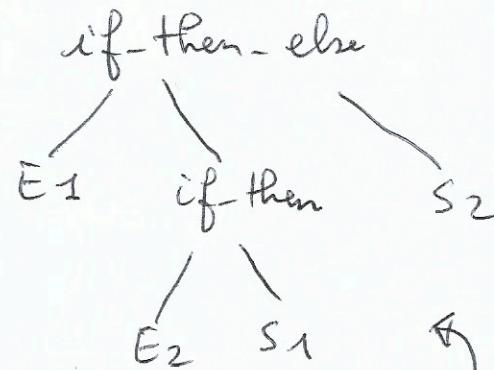
Consideriamo questa porzione di gram. per comandi:

$\langle \text{stmt} \rangle ::= \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{stmt} \rangle \mid$
 $\quad \quad \quad \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{stmt} \rangle \text{ else } \langle \text{stmt} \rangle \mid \dots$

È ambigua. Infatti: $\text{if } E_1 \text{ then if } E_2 \text{ then } S_1 \text{ else } S_2$ ammette due alberi di derivazione



(interpretazione usuale)



(albero
"errato")

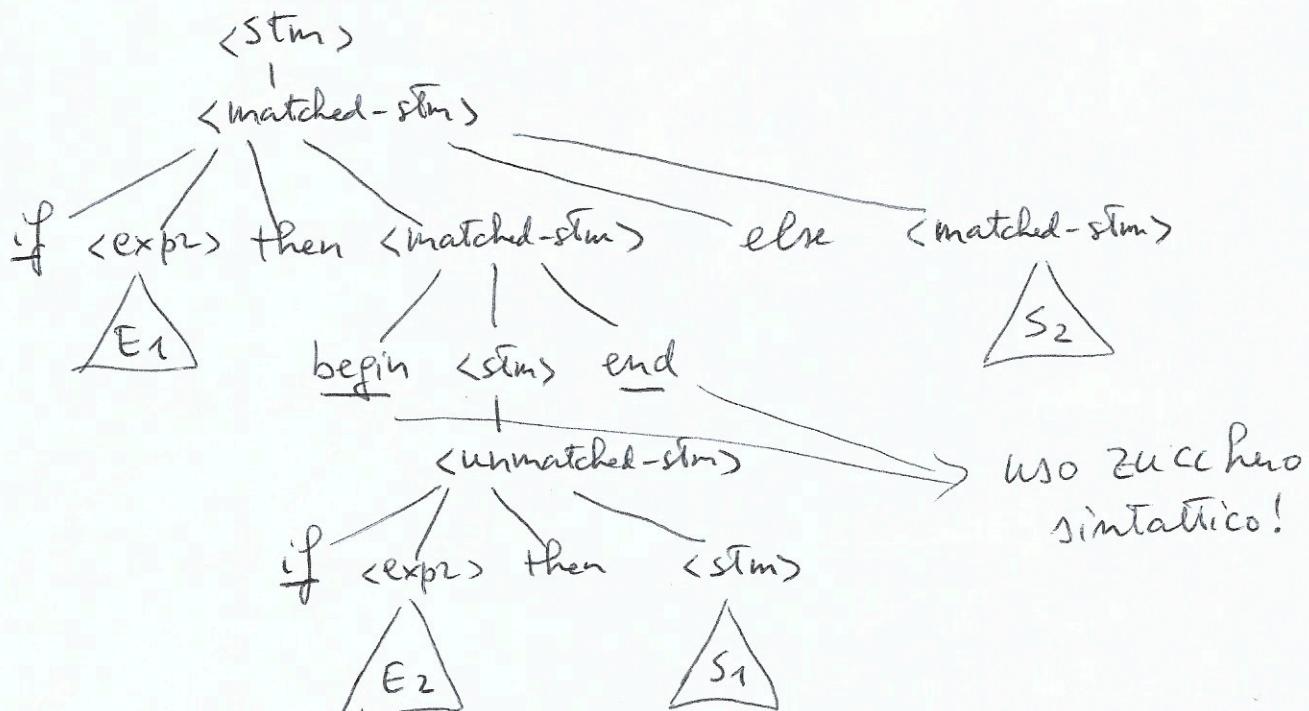
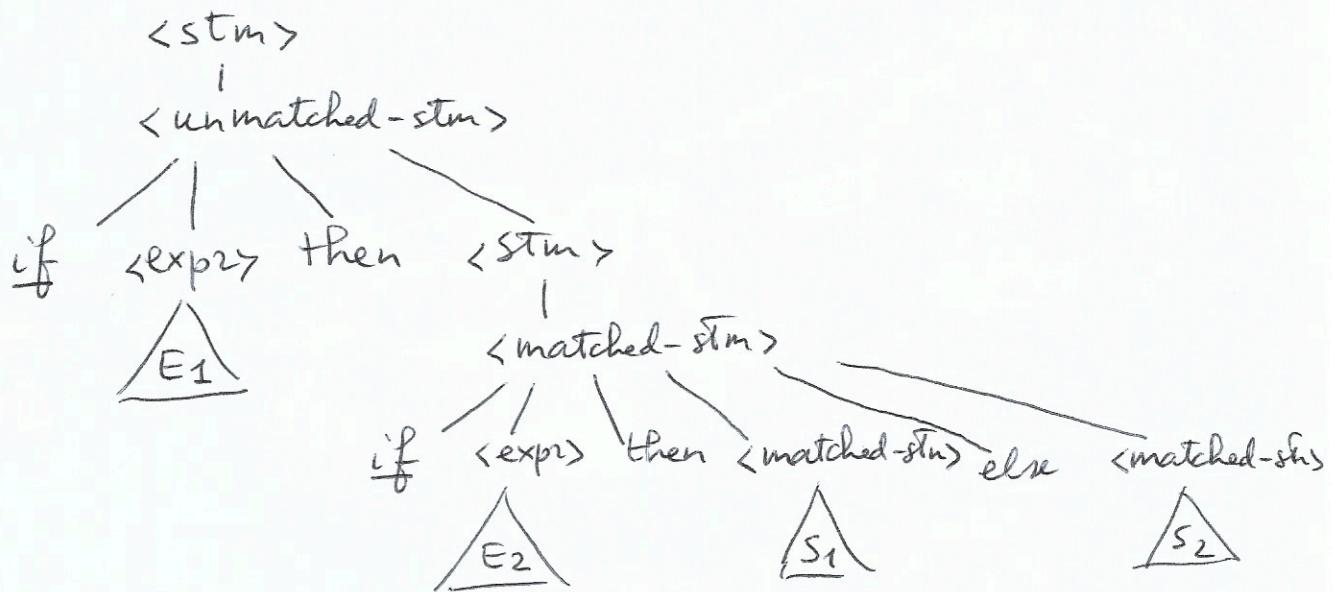
Grammatica Concreta

(43)

$\langle \text{stm} \rangle ::= \langle \text{matched-stm} \rangle \mid \langle \text{unmatched-stm} \rangle$

$\langle \text{matched-stm} \rangle ::= \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{matched-stm} \rangle$
 $\quad \quad \quad \text{else } \langle \text{matched-stm} \rangle \mid \dots \mid \text{begin } \langle \text{stm} \rangle \text{ end}$

$\langle \text{unmatched-stm} \rangle ::= \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{stm} \rangle \mid$
 $\quad \quad \quad \text{if } \langle \text{expr} \rangle \text{ then } \langle \text{matched-stm} \rangle$
 $\quad \quad \quad \text{else } \langle \text{unmatched-stm} \rangle \mid \dots$

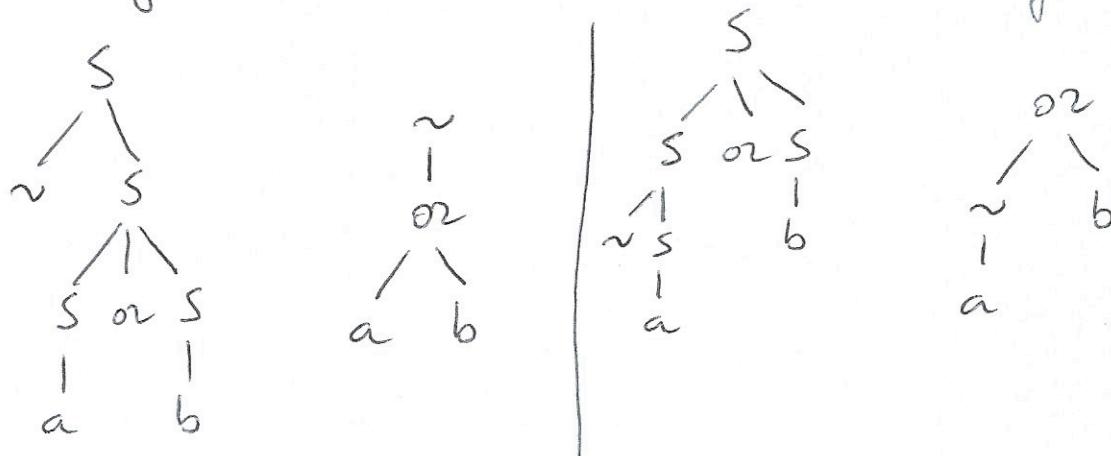


Esercizio : Espressioni booleane

(4h)

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

1) È ambigua? Sì. Consideriamo la stringa ~a ab



2) Grammatica non ambigua equivalente?

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

$$C \rightarrow \sim C \mid A$$

$$A \rightarrow a \mid b$$

- \sim lega più fortemente dell' or
- or associativo a sx

3) La nuova grammatica genera l'albero



4) Modificare la grammatica per generare quell'albero

Sintassi Concrete	$B \rightarrow B \text{ or } C \mid C$ $C \rightarrow \sim C \mid A$ $A \rightarrow a \mid b \mid (B)$
----------------------	--

