$E :\rightarrow E+E \mid E*E \mid$

$(E) \mid id$

$(id * (id + id))$

$(id + id), id * id + id$

1) To check membership

$E \rightarrow ($, $\rightarrow P2$

$id - id + id$ ✗

2) errors which are graceful

✓ '-' does not exist

✗ invalid string

3) parse trees → ASTs

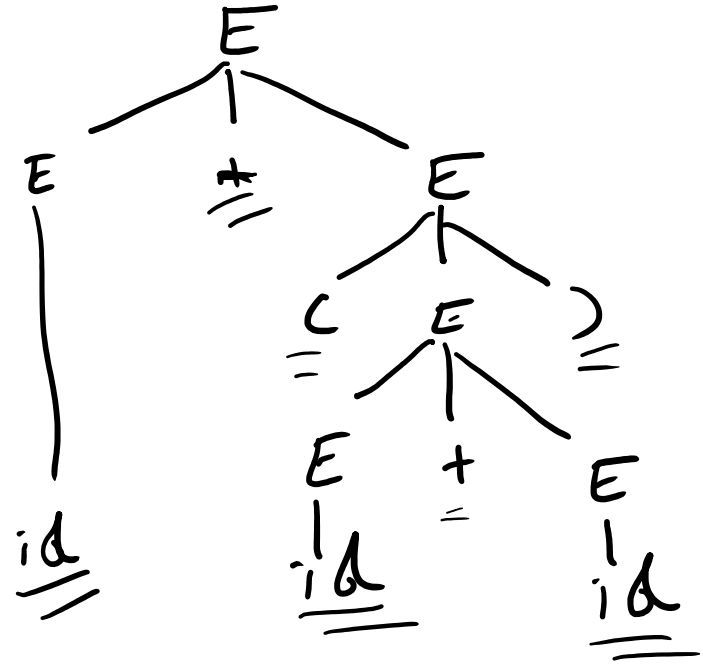
They depict the steps of derivation

Abstract syntax trees

$E :\to E+E \mid E*E \mid (E) \mid id$

id * (id + id)

$E \to E * E \to E * (E)$
$\to E * (E + E)$
$\to id * (E + E)$
$\to id * (id + E)$
$\to id * (id + id)$

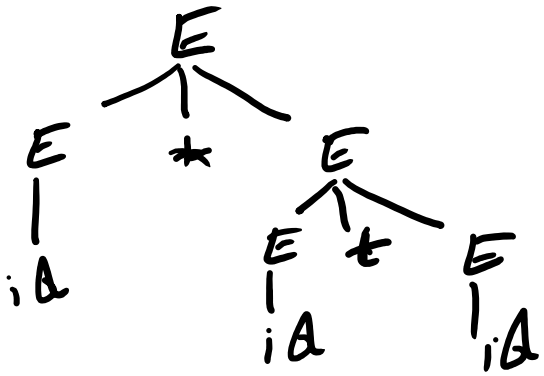


Left most or right most derivation

id * id + id

$E \to E + E \to id + E$
$\to id + E + E \to id * id + E$
$\to id * id + id$

-> id * id + id



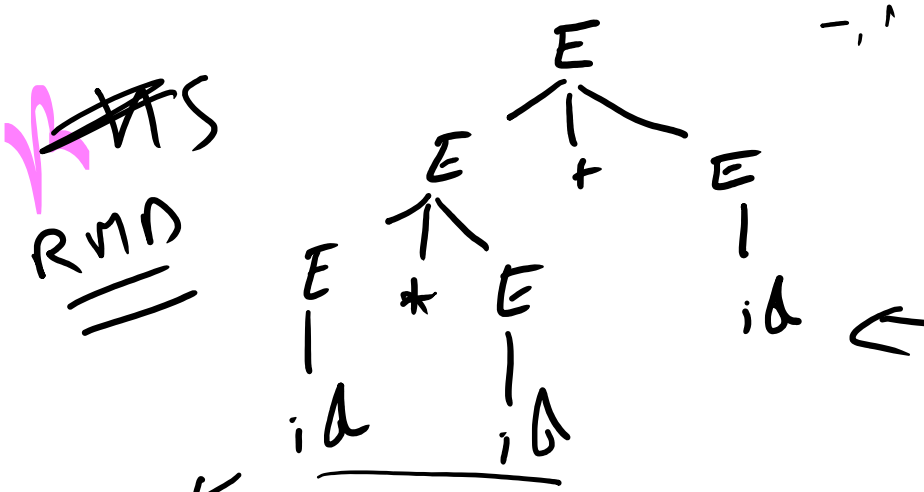
id * id + id

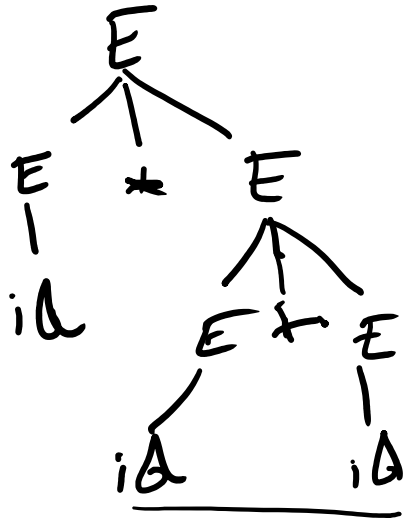
E -> E + E -> ~~E * E + E~~

-> E + id -> E * E + id

-> id * E + id + id

-> id * id + id





5+4+3

20+3=23

id * id + id

5 * 4 + 3

5+4+3

5 + 7

= 35

-> S
E -> E + E' | E

$$\rightarrow \left\{ \begin{array}{l} E \rightarrow E + E \mid E \\ E' \rightarrow id * E' \mid (E) * E' \mid id \mid (E) \end{array} \right\}$$

unambiguous.

Remove ambiguity:

set up rules

1) precedence
2) associativity

} arithem..

if - then - else

$E \rightarrow$ if E then E |

if E then E else E |

← string

if s_1 then if s_2 then s_3 else s_4

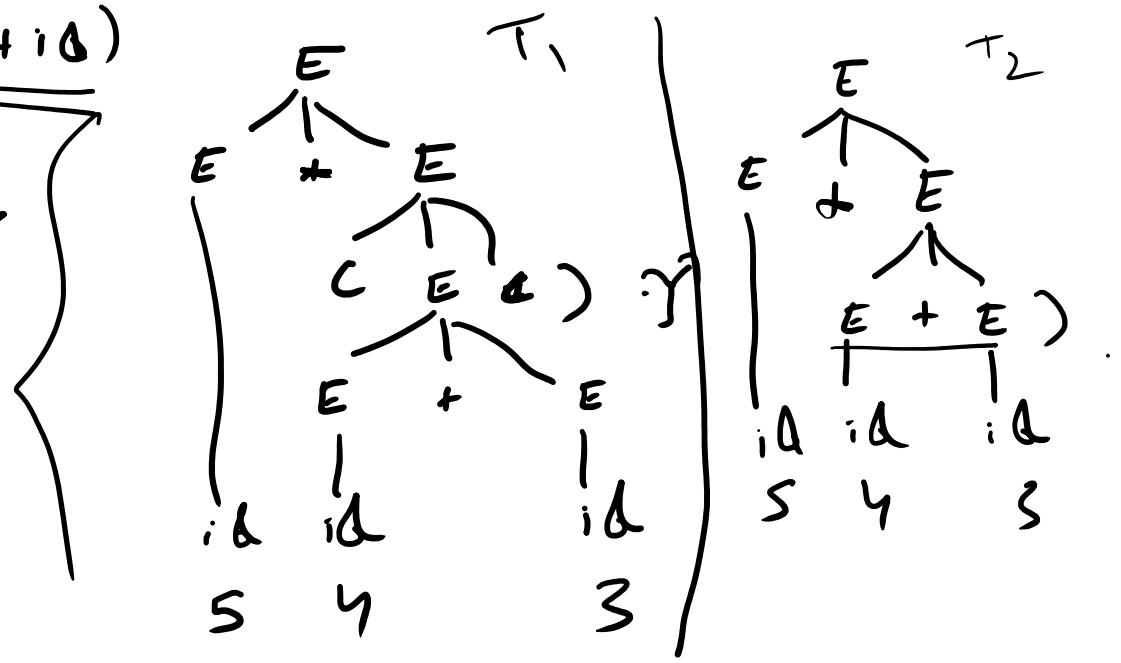
Rule

+, −

$E \rightarrow E + E \mid E * E \mid (E) \mid id$.

$id + (id * id)$

Parse tree

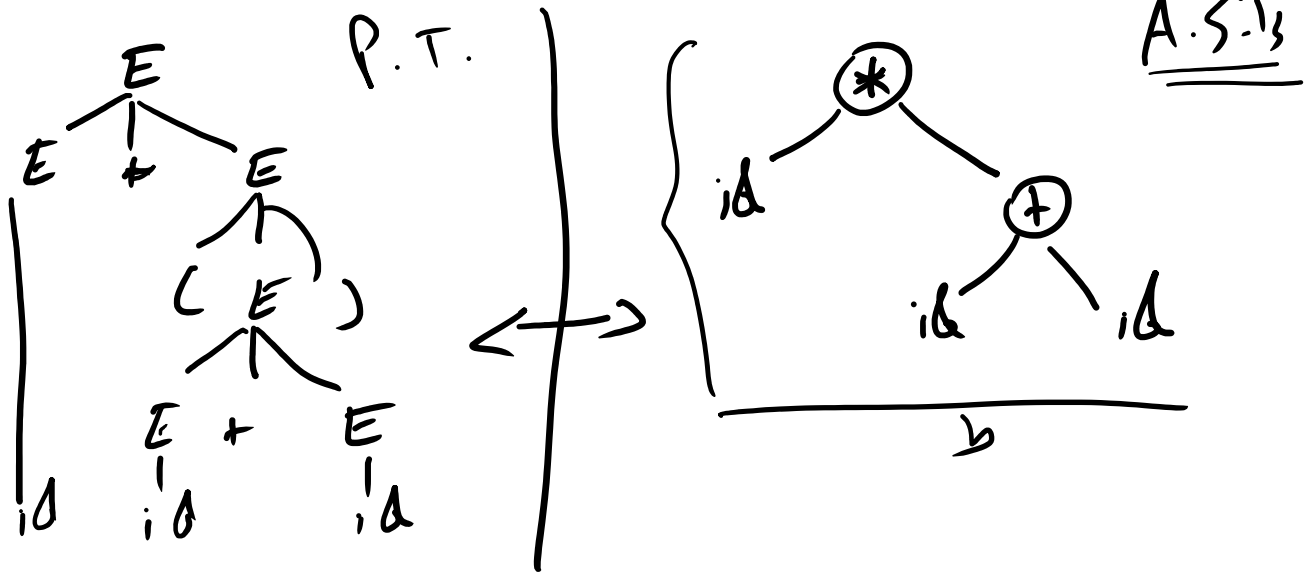


Tree st →

PL. SYNT → ( ) , . . . { }

$5 + (4 * 3)$ →

ASTs



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

$id * id + id$

T.D

$E \to E*E$
$\to E+E*E$
$\to id*id+id$

$\longleftrightarrow$

B.U. $id*id+id$
$\to id*id+E$
$\to id*E+E$
$\to id*E$
$\to E*E$
$\to E$

B.T

id — $E \to E+E \to (E) \to id$

T.D

$id+id$ — $E \to E+E \to id+E \to id+id$

B.U $\longleftrightarrow$ P.D.A

U.B

$id+id$

$E \to a(*)$

$E*E$

$id+$

$E \to E+E$

$id\ id$

P.T $\to$ A.ST

SEMTsδ oPt

iδ+iQ +iδ iQ

(+e e)