Bottom-Up Parsing

- A bottom up parsing corresponds to the construction of a parse tree for a input string.
- The parse tree is constructed beginning at the leaves (i.e. the bottom) and working up towards the root. (i.e. the top)

eg:

$E \rightarrow E + T \mid T$
$T \rightarrow T * F \mid F$
$F \rightarrow id \mid (E)$

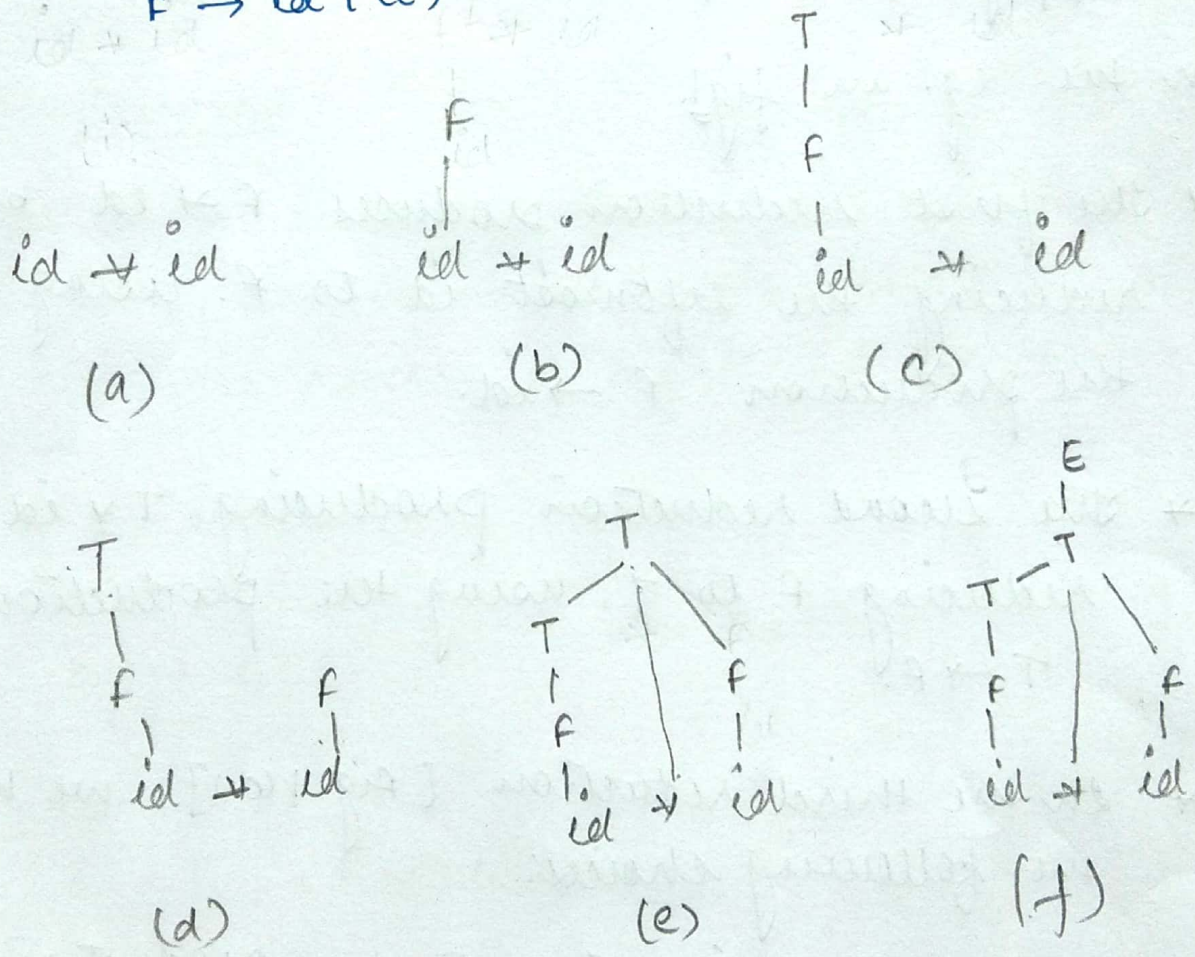


fig: The Bottom-up parse for id * id

Reductions

- Bottom up parsing is the process of "reducing" a string 'w' to the start symbol of the grammar.

- At each reduction step, a specific substring matching the body of a production is replaced by the non-terminal at the head of that production.

- ~~The key dec. decision decisions~~

- The key decisions during bottom-up parsing are about when to reduce and about what productions to apply, as the parse proceed.

- In the eg. in fig:
  - The first reduction produces F * id by reducing the leftmost id to F. using the production $F \rightarrow id$.
  - The second reduction producing T * id by reducing F to T. using the production $T \rightarrow F$.
  - In the third reduction [fig 1(c)], we have the following choices:
    - reducing T to E by production $E \rightarrow T$

      or

      reducing the second id to F by production $F \rightarrow id$.

    The obvious choice is to reduce id to F by production $F \rightarrow id$.

* the string then reduces to T and the parse tree completes with the reduction of T to start symbol E.

NOTE: The goal of bottom-up parsing is therefore to construct a ~~derivation~~ right-most derivation in reverse.

$$E \Rightarrow T \Rightarrow T * f \Rightarrow T * id \Rightarrow f * id \Rightarrow id * id$$

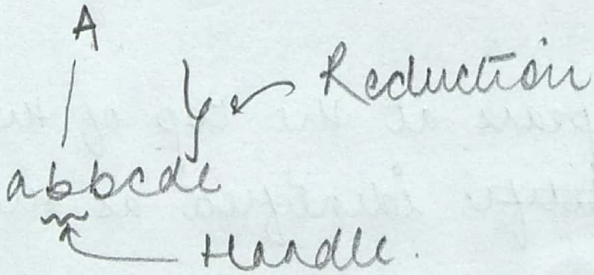
Handle Pruning

- Handle is a substring that matches the body of a production, and whose reduction represents one step along the reverse of right-most derivation.

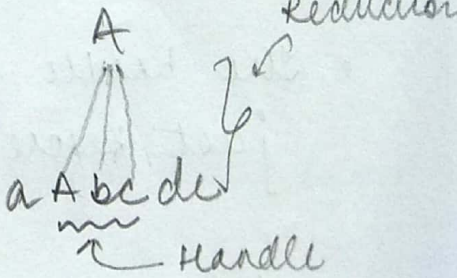
- Eg:
  $S \rightarrow aABe$
  $A \rightarrow Abc \mid b$
  $B \rightarrow d$

  $w = abbcde$

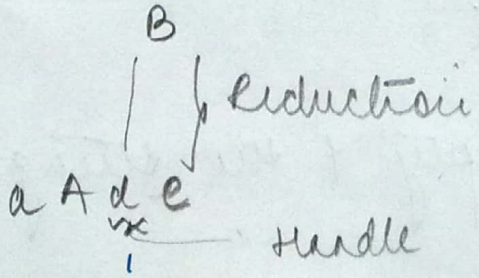
(i)



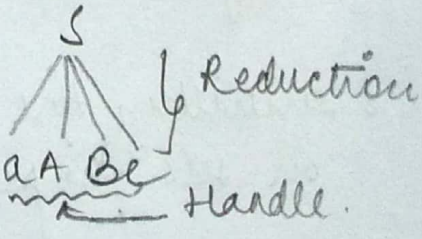
(ii)



(iii)



(iv)



eg $E \to E+T \mid T$
$T \to T*f \mid f$
$f \to (E) \mid id$

Right sentential form	Handle	Reducing Production
id * id	id	$f \to id$
f * id	f	$T \to f$
T * id	id	$f \to id$
T * f	T * f	$T \to T*f$
T	T	$E \to T$

Shift-Reducing Parser

- shift-reduce parsing is a form of Bottom-up parsing.
- In shift-reduce parsing:
  - stack holds grammar symbols.
  - input buffer holds the rest of the string to be parsed.
- The handle always appears at the top of the stack just before it is identified as the handle
- $ marks the bottom of the stack & also the right end of the input.
- Initially, the stack is empty & the string w is on the input:

- During left-to-right scan of the input string, the parser shifts zero or more input symbols onto the stack, until it is ready to reduce a string β of the grammar symbol on the top of the stack.
- It then reduces β to the head of the appropriate production.
- The parser repeats this cycle until it has detected an error or until the stack contains the start symbol and the input is empty:

Stack	Input
$ S	$

- Upon entering this configuration, the parser halts & announces successful completion of parsing.

Actions performed by shift-reduce parser:

a) **Shift**: Shift the next input symbol onto the top of the stack.

b) **Reduce**: The right end of the string to be reduced must be at the top of the stack. Locate the left end of the string within the stack & decide with what nonterminal to replace the string.

c) **Accept**: Announce successful completion of parsing

d) **Error**: Discover a syntax error and call an error recovery routine.

Note: The handle will always appear on the top of the stack, never inside.

Stack	Input	Action
$	id + id $	shift
$ id	+ id $	reduce by F → id
$ F	+ id $	reduce by T → F
$ T	+ id $	shift
$ T +	id $	shift
$ T + id	$	reduce F → id
$ T + F	$	reduce T → T + F
$ T	$	reduce E → T
$ E	$	accept

handle

shift

shift

shift

Eg 2 $S \to (L) \mid a$

$L \to L,S \mid S$ $\quad w = (a,(a,a))$

Stack	Input Buffer	Action
$	(a,(a,a))$	shift
$(	a,(a,a))$	shift
$(a	,(a,a))$	reduce $S \to a$
$(S	,(a,a))$	reduce $L \to S$
$(L	,(a,a))$	shift
$(L,	(a,a))$	shift
$(L,(	a,a))$	shift
$(L,(a	,a))$	reduce $S \to a$
$(L,(S	,a))$	reduce $L \to S$
$(L,(L	,a))$	shift
$(L,(L,	a))$	shift
$(L,(L,a	))$	Reduce $S \to a$
$(L,(L,S	))$	Reduce $L \to L,S$
$(L,(L	))$	shift
$(L,(L)	)$	Reduce $S \to (L)$
$(L,S	)$	Reduce $L \to L,S$
$(L	)$	shift
$(L)	$	$S \to (L)$
$S	$	accept

Q3

$S \rightarrow TL$

$T \rightarrow int \mid float$

$L \rightarrow L, id \mid id$

$w = int\ id, id.$

(1)

$E \rightarrow E - E$

$E \rightarrow E + E$

$E \rightarrow id$

$w = id - id * id$