

# Parsing bottom up

Corso di Fondamenti di Informatica - modulo 1  
Corso di Laurea in Informatica  
Università di Roma "Tor Vergata"  
a.a. 2020-2021

Giorgio Gambosi

## Parsing bottom up

Costruzione dell'albero sintattico dal basso verso l'alto.

Equivalentemente, costruzione di una derivazione destra (in ordine inverso rispetto alla derivazione stessa).

Parsing LR( $k$ ):

- Left-to-right: la derivazione è calcolata da sinistra a destra (dalla prima produzione applicata all'ultima)
- Rightmost derivation: la derivazione calcolata è destra
- $k$  simboli (di look-ahead) da considerare

## Riduzioni

Operazioni base del parsing LR:

Una sottostringa (*handle*) della forma di frase attuale  $\alpha$ , corrispondente alla parte destra di una produzione, viene sostituita dalla parte sinistra.

- Forma di frase attuale  $\alpha = \delta\eta\zeta$ , con  $\delta, \zeta \in (V_T \cup V_N)^*$ , e  $\eta \in (V_T \cup V_N)^* V_N (V_T \cup V_N)^*$
- Produzione  $A \rightarrow \eta \in P$

Riduzione: la nuova forma di frase è  $\delta A \zeta$

## Esempio di parsing bottom up

Consideriamo ancora la grammatica

$$\begin{aligned} E &\rightarrow TE' \\ E' &\rightarrow +TE' \mid \varepsilon \\ T &\rightarrow FT' \\ T' &\rightarrow *FT' \mid \varepsilon \\ F &\rightarrow (E) \mid \text{id} \end{aligned}$$

e la stringa  $\text{id} + \text{id} * \text{id}$ .

## Esempio di parsing bottom up

Assumiamo di poter individuare sempre la prima handle da sinistra nella forma di frase attuale.

Handle	Produzione	Result
$\text{id} + \text{id} * \text{id} \$$	$F \rightarrow \text{id}$	$F + \text{id} * \text{id}$
$F + \text{id} * \text{id} \$$	$T' \rightarrow \varepsilon$	$FT' + \text{id} * \text{id}$
$FT' + \text{id} * \text{id} \$$	$T \rightarrow FT'$	$T + \text{id} * \text{id}$
$T + \text{id} * \text{id} \$$	$F \rightarrow \text{id}$	$T + F * \text{id}$
$T + F * \text{id} \$$	$F \rightarrow \text{id}$	$T + F * F$
$T + F * F \$$	$T' \rightarrow \varepsilon$	$T + F * FT'$
$T + F * FT' \$$	$T' \rightarrow *FT'$	$T + FT'$
$T + FT' \$$	$T \rightarrow FT'$	$T + T$
$T + T \$$	$E' \rightarrow \varepsilon$	$T + TE'$
$T + TE' \$$	$E' \rightarrow +TE'$	$TE'$
$TE' \$$	$E \rightarrow TE'$	$E$

### Esempio di parsing bottom up

La sequenza di produzioni individuate, lette al contrario (dal basso in alto), forniscono la derivazione destra della stringa

$$E \Rightarrow TE' \Rightarrow T + TE' \Rightarrow T + T \Rightarrow T + FT' \Rightarrow T + F * FT' \Rightarrow T + F * F \Rightarrow T + F * id \Rightarrow T + id * id \Rightarrow FT' + id * id \Rightarrow F + id * id \Rightarrow id + id * id$$

Problema: come individuare handle e riduzione?

### Parsing shift-reduce

Un parser *shift-reduce* effettua parsing bottom up utilizzando:

- una pila di simboli della grammatica
- un buffer di input, in cui è contenuta la parte dell'input ancora da leggere

Il carattere  $\$ \notin V_T$  è utilizzato per marcare il fondo della pila e la fine della stringa di input, per cui inizialmente, se  $w$  è la stringa:

- la pila contiene  $\$$
- il buffer contiene  $w\$$

Pila  
\$

Input  
w\$

### Parsing shift-reduce: operazioni

Reduce: i simboli in cima alla pila corrispondono ad una handle per una produzione  $A \rightarrow \alpha$ . Viene effettuata una reduce, passando da

Pila                      Input                      a                      Pila                      Input  
\$βα                      w\$                      \$βA                      w\$

Shift: i simboli in cima alla pila non corrispondono ad una handle. Il prossimo simbolo in input viene posto sulla pila, passando da

Pila                      Input                      a                      Pila                      Input  
\$α                      aw\$                      \$αa                      w\$

### Parsing shift-reduce: operazioni

Accept: la pila contiene  $\$S$ , dove  $S$  è l'assioma della grammatica, il buffer di input contiene  $\$$  (la stringa è terminata). La stringa è accettata

Error: non ci sono altre azioni eseguibili nella configurazione attuale di pila e buffer. La stringa è rifiutata

### Parsing shift-reduce

Pila	Input	Azione
\$	id+id*id\$	shift
\$id	+id*id\$	reduce $F \rightarrow id$
\$F	+id*id\$	reduce $T' \rightarrow \varepsilon$
\$FT'	+id*id\$	reduce $T \rightarrow FT'$
\$T	+id*id\$	shift
\$T+	id*id\$	shift
\$T + id	*id\$	reduce $F \rightarrow id$
\$T + F	*id\$	shift
\$T + F*	id\$	shift
\$T + F * id	\$	reduce $F \rightarrow id$
\$T + F * F	\$	reduce $T' \rightarrow \varepsilon$
\$T + F * FT'	\$	reduce $T' \rightarrow *FT'$

### Parsing shift-reduce

Pila	Input	Azione
\$T + FT'	\$	reduce $T \rightarrow FT'$
\$T + T	\$	reduce $E' \rightarrow \varepsilon$
\$T + TE'	\$	reduce $E' \rightarrow +TE'$
\$TE'	\$	reduce $E \rightarrow TE'$
\$E	\$	accept

Si può mostrare che l'handle, in un parsing bottom up, apparirà sempre in cima alla pila.

#### Riconoscimento di handle

Definizione di un automa a stati finiti  $\mathcal{A}$  che riconosce handle.

- Una operazione *shift* corrisponde ad una transizione di  $\mathcal{A}$  da uno stato al successivo
- Una operazione *reduce* corrisponde al riconoscimento di una handle

Come definire l'automa  $\mathcal{A}$ ?

- Alfabeto?
- Stati?
- Funzione di transizione?