UPE – Caruaru – Sistemas de Informação
Disciplina: Compiladores
Prof.: Paulemir G. Campos

Parsing LR

Analisadores Sintáticos Ascendentes
(*Bottom-Up***)**

- Um *analisador sintático ascendente* tenta construir a árvore de derivação sintática (ADS) para uma sentença *w* a partir dos símbolos de *w* (folhas), fazendo reduções (substituindo o lado direito de uma regra pelo seu lado esquerdo) até obter o símbolo inicial *S* (raiz).

- A idéia basicamente consiste em encontrar e reduzir sucessivamente o *handle* até que seja obtido o símbolo inicial da gramática ou uma ocorrência de erro.

Adaptado de notas de aulas do prof. Giuseppe Mongiovi do DI/UFPB.

Exemplo:

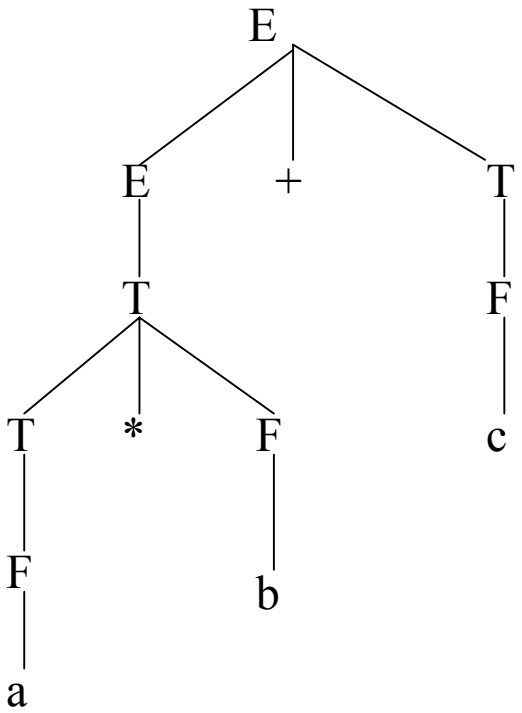
Seja a seguinte gramática G(E) para gerar expressões aritméticas:

E ::= E + T | T
T ::= T * F | F
F ::= id | (E)

Reconhecer a sentença $w = a * b + c$

$E \Rightarrow E + \underline{T} \Rightarrow E + \underline{F} \Rightarrow \underline{E} + id(c) \Rightarrow \underline{T} + id(c) \Rightarrow T*\underline{F} + id(c)$
$\Rightarrow \underline{T}* id(b) + id(c) \Rightarrow \underline{F}* id(b) + id(c) \Rightarrow id(a) * id(b) + id(c)$

forma sentencial	**handle**	**redução**
a * b + c	a	F→ id
F * b + c	F	T→ F
T * b + c	b	F → id
T * F + c	T*F	T→ T*F
T + c	T	E → T
E + c	c	F → id
E + F	F	T → F
E + T	E + T	E → E+T



De posse da idéia de funcionamento do analisador ascendente, passemos a considerar os problemas que devem ser resolvidos:

1. Como identificar o *handle* (a parte da cadeia w que deve ser reduzida)?

2. Que produção deve ser usada na redução?

Esses dois problemas podem ser facilmente resolvidos para um certo tipo de gramáticas denominadas Gramáticas de Precedência Simples.

GRAMÁTICAS DE PRECEDÊNCIA SIMPLES

Dada uma forma sentencial *w*, como podemos descobrir quem é o *handle*?

Uma forma:

Observando a relação de precedência entre dois símbolos adjacentes das formas sentenciais.

Para algumas formas sentenciais do exemplo anterior temos:

1) a * b + c
a→*←b→+←c
handle = a

Antes de efetuar o produto, a e b devem ser conhecidos. Portanto,
a tem precedência em relação a *
b tem precedência em relação a *

2) T + c
T →+ ←c
handle = T

Antes de efetuar a adição, T e c devem ser conhecidos. Portanto,
T tem precedência em relação a +
c tem precedência em relação a +

De 1) e 2) concluímos:

$id \rightarrow *,\ * \leftarrow id,\ id \rightarrow +,\ + \leftarrow id,\ T \rightarrow +,\ + \leftarrow T$

→Relações de precedência

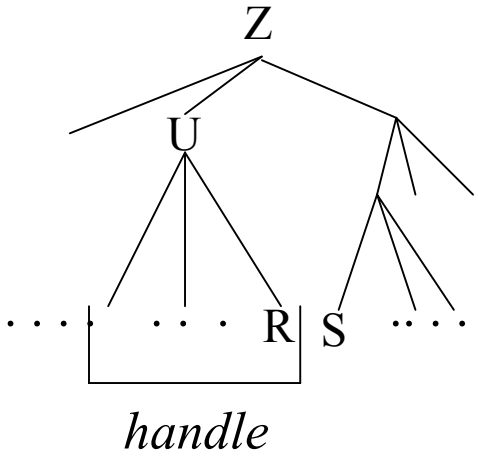
Seja a seguinte forma sentencial canônica

$w = \ldots RS \ldots \qquad R, S \in V$

Em algum ponto da análise sintática, R e/ou S farão parte de um *handle*:

Existem três possibilidades:

1) Apenas R faz parte do *handle*

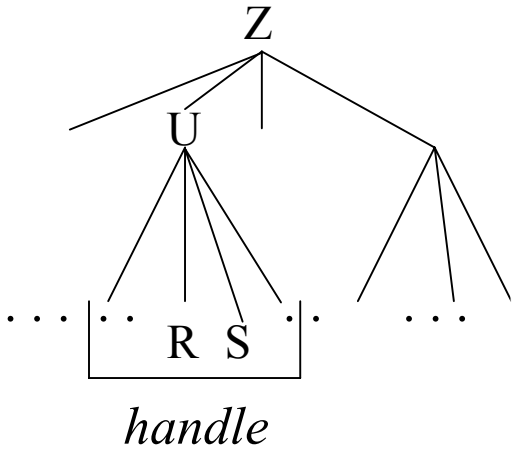


- R tem precedência em relação a S
  (R deve ser reduzido antes de S)

$$R \rightarrow S$$

- existe uma produção do tipo U→ . . . R

2) R e S fazem parte do *handle*

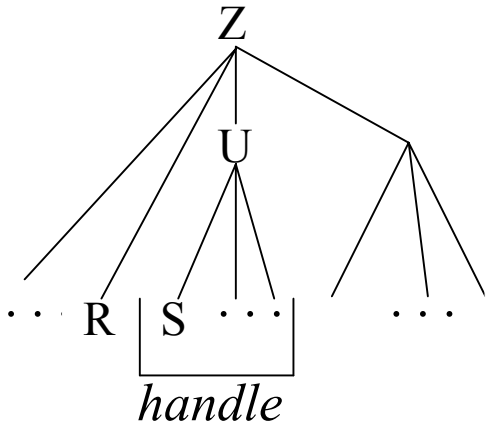


- R e S têm a mesma precedência
  (devem ser reduzidos ao mesmo tempo)

$$R \pm S$$

- existe uma produção do tipo U→ . .RS. .

3) Apenas S faz parte do *handle*



- S tem precedência em relação a R
  (S deve ser reduzido antes de R)

$$R \leftarrow S$$

- existe uma produção do tipo U→S . . .

Se não existir nenhuma forma sentencial canônica . . . RS . . .,
Então não haverá relação entre R e S (situação de ERRO)

OBS: As relações de precedência (←, ±, →) **não** são simétricas,
i. e., R → S não implica S ← R

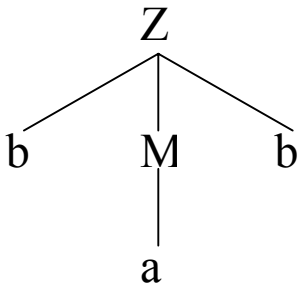
Exemplo – Seja a gramática G(Z)

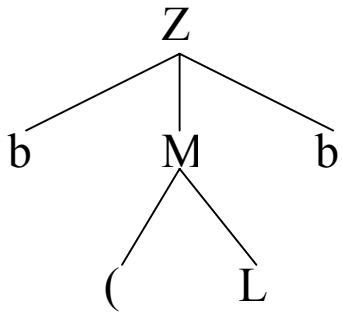
Z ::= bMb
M ::= (L|a
L ::= Ma)

$L(G(Z)) = \{b(^{n}a[a)]^{n}b \mid n \geq 0\}$

Forma sentencial: b a b b (L b

Árvore sintática:





Handle: a (L

Relações fornecidas pela árvore sintática:

b ← a
a → b

b ← (
(± L
L → b

Matriz de Precedência – representa as relações de precedência entre todos os símbolos de uma gramática.

	Z	**b**	**M**	**L**	**a**	**(**	**)**
Z							
b			±		←	←	
M		±			±		
L		→			→		
a		→			→		±
(			←	±	←	←	
)		→			→		

Matriz de Precedência para a gramática G(Z)

Para que serve a Matriz de Precedência?

Se existir, no máximo, uma relação entre qualquer par de símbolos (R, S),
então as relações indicarão o *handle* de qualquer forma sentencial

Como encontrar o *handle*?

Seja $w = S_1S_2 \ldots S_n$ uma forma sentencial qualquer.
A subcadeia, mais à esquerda, $S_iS_{i+1} \ldots S_j$ ($i \geq 1$ e $j \leq n$) será o *handle* de w se somente se:

$S_{i-1} \leftarrow S_i \qquad S_i \pm S_{i+1} \pm \ldots \pm S_j \qquad S_j \rightarrow S_{j+1}$

onde: $S_0 = S_{n+1} = \$$, $\$ \leftarrow S_i$ e $S_i \rightarrow \$$ p/ $i = 1,2,\ldots n$
($ - símbolo delimitador da cadeia)

Exemplo:

Utilizando a Matriz de Precedência da gramática G(Z), verificar se $w = b(aa)b \in L(G(Z))$

forma sentencial	***handle***	**redução do handle para**
$\leftarrow b\leftarrow(\leftarrow a\rightarrow a\pm)\rightarrow b\rightarrow\$$	a	M
$\leftarrow b\leftarrow(\leftarrow M\pm a\pm)\rightarrow b\rightarrow\$$	Ma)	L
$\leftarrow b\leftarrow(\pm L\rightarrow b\rightarrow\$$	(L	M
$\leftarrow b\pm M\pm b\rightarrow\$$	bMb	Z
$\leftarrow Z\rightarrow\$$	Z	$w \in L(G(Z))$

Definição:

Uma gramática G é chamada de precedência simples se:

1) Existe no máximo uma relação de precedência entre dois símbolos quaisquer de seu vocabulário V;

2) Todas as produções de G tiverem lados direitos únicos.

OBS:

Condição 1 – fornece elementos para encontrar o *handle*
Condição 2 – assegura que a redução do *handle* é única

Teorema

Uma gramática de precedência simples é não ambígua. O único *handle* de qualquer forma sentencial $w = S_1S_2 \ldots S_n$ é a subcadeia, mais à esquerda, $S_iS_{i+1} \ldots S_j$ ($i \geq 1$ e $j \leq n$) tal que:

$S_{i-1} \leftarrow S_i \qquad S_i \pm S_{i+1} \pm \ldots \pm S_j \qquad S_j \rightarrow S_{j+1}$

onde: $S_0 = S_{n+1} = \$$, $\$ \leftarrow S_i$ e $S_i \rightarrow \$$ p/ $i = 1,2,\ldots n$

Implementação do Analisador Ascendente de Precedência Simples

Estrutura de dados

- uma tabela (matriz) de precedência com valores:

  T[i, j] = 0 se não existe relação entre S_i e Sj
  T[i, j] = 1 se $S_i \leftarrow S_j$
  T[i, j] = 2 se $S_i \pm S_j$
  T[i, j] = 3 se $S_i \rightarrow S_j$

- uma tabela que armazene as produções de tal forma que dado um lado direito, possamos localizá-lo na tabela e identificar o correspondente lado esquerdo;
- uma pilha onde são armazenados os símbolos da seqüência de entrada (processados da esquerda para a direita) até que seja identificada uma relação → entre o símbolo do topo da pilha e o símbolo de entrada.

• Referências

Notas de aulas do prof. Giuseppe Mongiovi do DI/UFPB, 2002.

Appel, A. W. **Modern Compiler Implementation in C.** Cambridge University Press, 1998. (Capítulo 3, seção 3.3).