Analisador Sintático parte III

Compiladores

Sumário

- Ambiguidade
- Técnicas de Parsing
- Eliminação de recursividade a esquerda

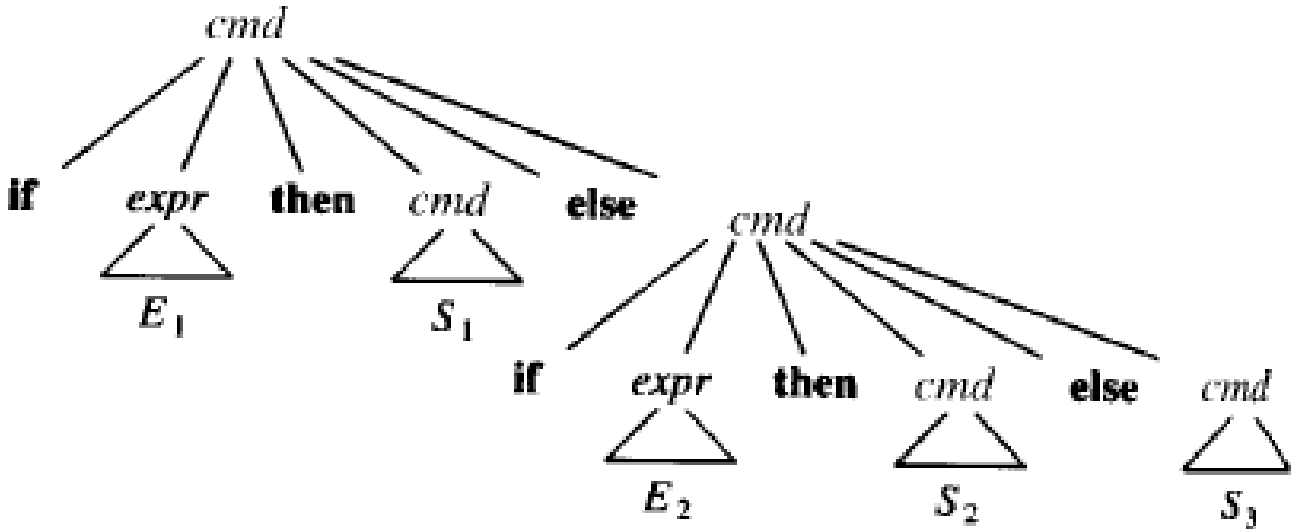
Ambiguidade

- Uma gramática que produza mais de uma árvore gramatical para alguma sentença é dita ambígua.

$$
\begin{aligned}
cmd \rightarrow\ & \textbf{if}\ expr\ \textbf{then}\ cmd \\
\mid\ & \textbf{if}\ expr\ \textbf{then}\ cmd\ \textbf{else}\ cmd \\
\mid\ & \textbf{outro}
\end{aligned}
$$

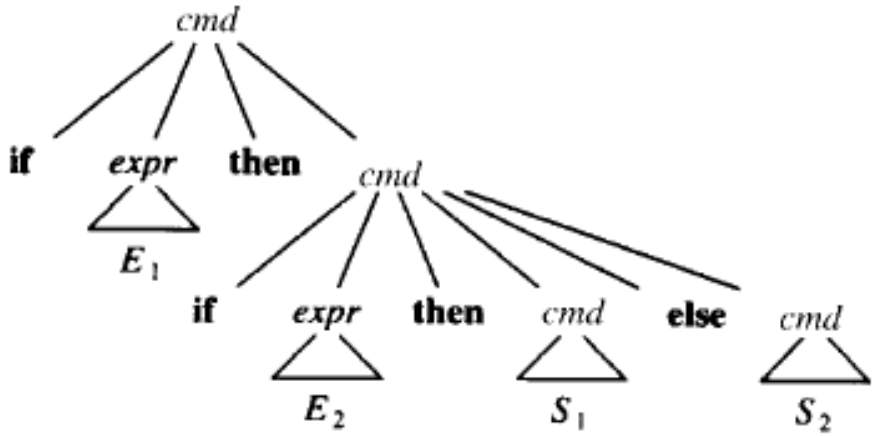
Ambiguidade

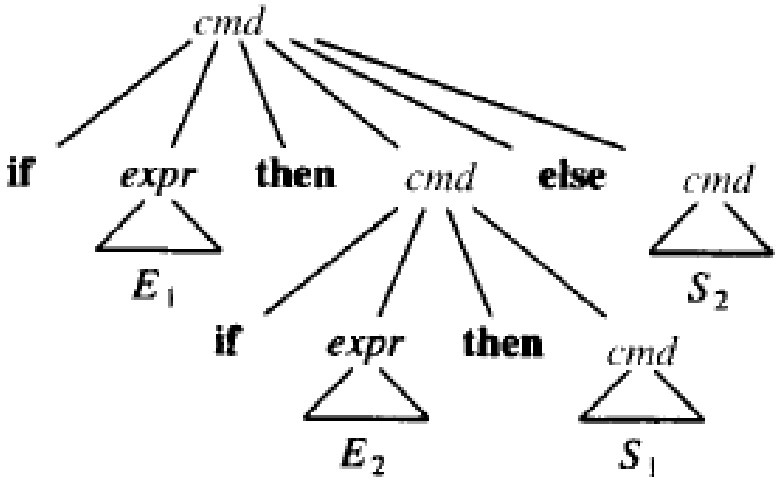
if E_1 **then** S_1 **else if** E_2 **then** S_2 **else** S_3



Ambiguidade

if E_1 **then if** E_2 **then** S_1 **else** S_2





Ambiguidade

$$
\begin{aligned}
\textit{cmd} &\rightarrow \textit{cmd\_associado} \\
&\mid \textit{cmd\_não associado} \\
\text{cmd\_associado} &\rightarrow \textbf{if}\ \text{expr}\ \textbf{then}\ \text{cmd\_associado}\ \textbf{else}\ \text{cmd\_associado} \\
&\mid \textbf{outro} \qquad (4.9) \\
\textit{cmd\_não associado} &\rightarrow \textbf{if}\ \textit{expr}\ \textbf{then}\ \textit{cmd} \\
&\mid \textbf{if}\ \textit{expr}\ \textbf{then}\ \textit{cmd\_associado}\ \textbf{else}\ \textit{cmd\_não associado}
\end{aligned}
$$

Tipos de *Parsing*

- Top-down parsers
  - LL(1): L é porque a entrada é da esquerda para a direita, L pois é derivação mais a esquerda, recursivo descendente e o 1 porque avalia um token adiante
  - Inicia na raiz da árvore e cresce em direção as folhas
  - Obtém uma producão e tenta associar a entrada
  - Se escolha errada ⇒ pode tentar retroceder (backtrack)
- Bottom-up parsers
  - LR(1): L é porque a entrada é da esquerda para a direita, R é porque é feito usando derivação mais a direita e o 1 porque avalia um token adiante
  - Inicia nas folhas e cresce até a raiz
  - Inicia em um estado válido para os primeiros tokens

Recursão a esquerda

- Uma gramática é recursiva à esquerda se possui um não-terminal **A** tal que exista uma derivação **A => Aα** para alguma cadeia **α**.

- Os métodos de análise sintática top-down não podem processar gramáticas recursivas à esquerda.

Recursão a esquerda

- O par de produções recursivas à esquerda

A => Aα | β

- Poderia ser substituído por

A => βA'

A' => αA' | ε

Exemplo

$$
\begin{aligned}
E &\rightarrow E + T \mid T \\
T &\rightarrow T * F \mid F \\
F &\rightarrow (E) \mid \mathbf{id}
\end{aligned}
$$

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

A análise Top-Down

- Tentativa de se encontrar uma derivação mais à esquerda para uma cadeia de entrada.

- Equivalentemente, pode ser vista como uma tentativa de construir uma árvore gramatical, para a cadeia de entrada, a partir da raiz.

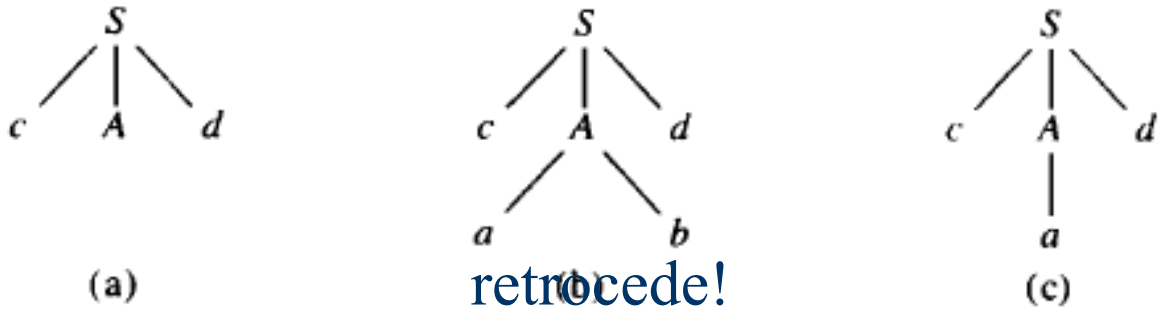
A análise Top-Down

- Considere a gramática:

S → cAd

A → ab | a

- E a cadeia de entrada w = cad.



A análise Bottom-Up

- Análise de empilhar e reduzir;
- Tenta construir uma árvore gramatical para uma cadeia de entrada começando pelas folhas e trabalhando árvore acima em direção à raiz;
- "Reduzir" uma cadeia w ao símbolo de partida de uma gramática.

A análise Bottom-Up

- Considere a gramática:

S -> aABe

A -> Abc | b

B -> d

- A sentença abbcde pode ser reduzida a S pelos seguintes passos:

abbcde

aAbcde

aAde

aABe

S

Pilha da Análise Sintática

- Quatro ações possíveis:
  - Empilhar
  - Reduzir
  - Aceitar
  - Erro.

Reduzir e Empilhar

	PILHA	ENTRADA	AÇÃO
(1)	$	$\mathbf{id}_1 + \mathbf{id}_2 * \mathbf{id}_3\$$	empilhar
(2)	$\$\mathbf{id}_1$	$+ \mathbf{id}_2 * \mathbf{id}_3\$$	reduzir por $E \rightarrow \mathbf{id}$
(3)	$\$E$	$+ \mathbf{id}_2 * \mathbf{id}_3\$$	empilhar
(4)	$\$E +$	$\mathbf{id}_2 * \mathbf{id}_3\$$	empilhar
(5)	$\$E + \mathbf{id}_2$	$* \mathbf{id}_3\$$	reduzir por $E \rightarrow \mathbf{id}$
(6)	$\$E + E$	$* \mathbf{id}_3\$$	empilhar
(7)	$\$E + E *$	$\mathbf{id}_3\$$	empilhar
(8)	$\$E + E * \mathbf{id}_3$	$\$$	reduzir por $E \rightarrow \mathbf{id}$
(9)	$\$E + E * E$	$\$$	reduzir por $E \rightarrow E * E$
(10)	$\$E + E$	$\$$	reduzir por $E \rightarrow E + E$
(11)	$\$E$	$\$$	aceitar

Fig. 4.22. Configurações de um analisador sintático de empilhar e reduzir para a entrada $\mathbf{id}_1 + \mathbf{id}_2 * \mathbf{id}_3$.