Análise Sintática

Eduardo Ferreira dos Santos

Ciência da Computação
Centro Universitário de Brasília – UniCEUB

Outubro, 2016

Sumário

Fases

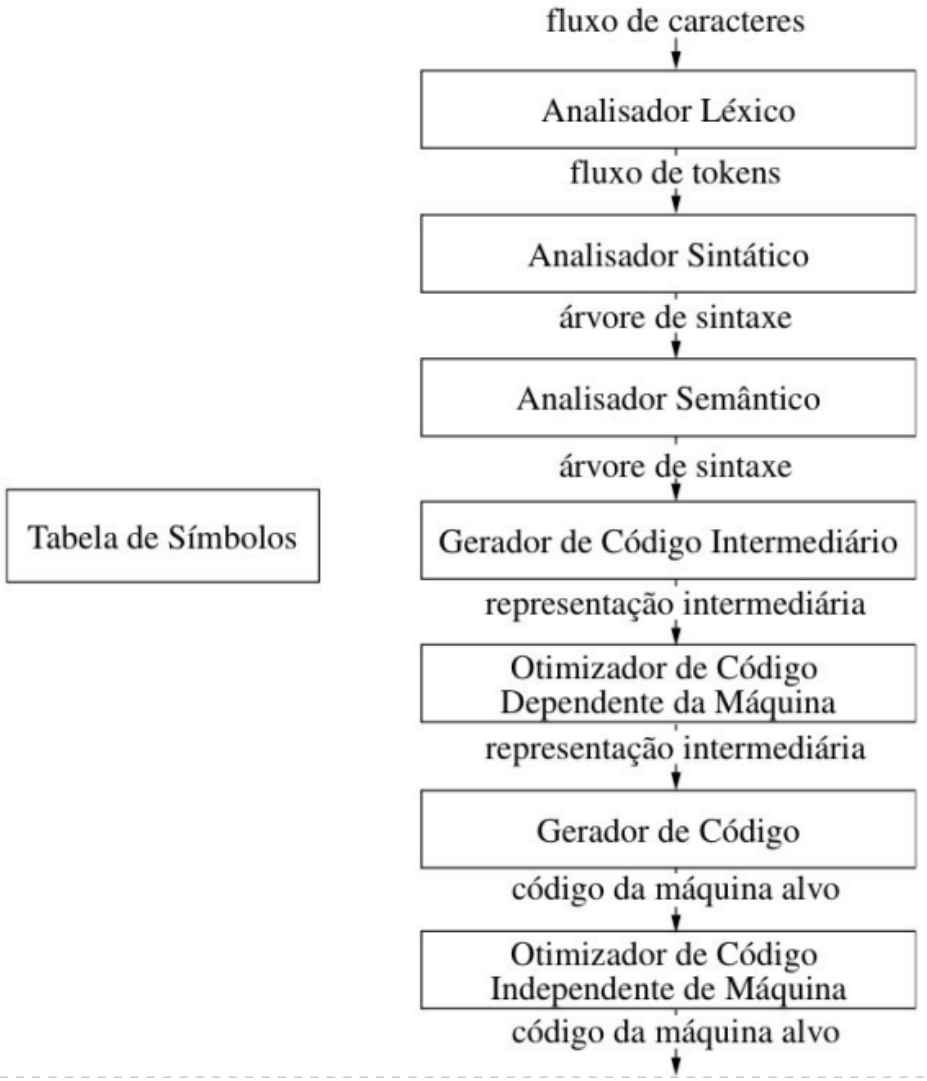

Figura 1.1: Fases do compilador [Aho et al., 2007]

Tokens, padrões e lexemas

Token Dupla formada pelo nome e valor de atributo (opcional)

- Nome do token: símbolo abstrato que representa um tipo de unidade léxica [Aho et al., 2007];
- Também é um símbolo de entrada para o analisador sintático.

Padrões Descrição da forma que os lexemas de um token podem assumir:

- Palavra-chave como token;
- Casamento de sequências de caracteres.

Lexemas Sequência de caracteres no programa fonte que casa com o padrão para um token.

- Identificado como instância desse token.

Papel do analisador sintático

- O analisador sintático recebe do analisador léxico a cadeia de tokens;
- Verifica se a cadeia pertence à linguagem gerada pela gramática;
- Deve emitir mensagens de erro e recuperar-se desses erros;
- Caso ocorra uma falha, deve continuar processando o programa.

Métodos de análise

- Técnicas de análise sintática [Aho et al., 2007]:
  1. Universal;
  2. Ascendente (*bottom up*);
  3. Descendete (*top down*).
- A abordagem universal é considerada muito ineficiente para ser utilizada em compiladores [de Alencar Price and Toscani, 2000];
- Os métodos utilizados em compiladores utilizam a abordagem ascendente ou descendente:

  Métodos descendentes Constroem a árvore de derivação de cima (raiz) para baixo (folhas);

  Métodos ascendentes Realizam a análise no sentido inverso, começando nas folhas e seguindo até a raiz.

Execução

A saída do analisador sintático é alguma representação da árvore de derivação para a cadeia de tokens reconhecidos pelo analisador léxico. [Aho et al., 2007]

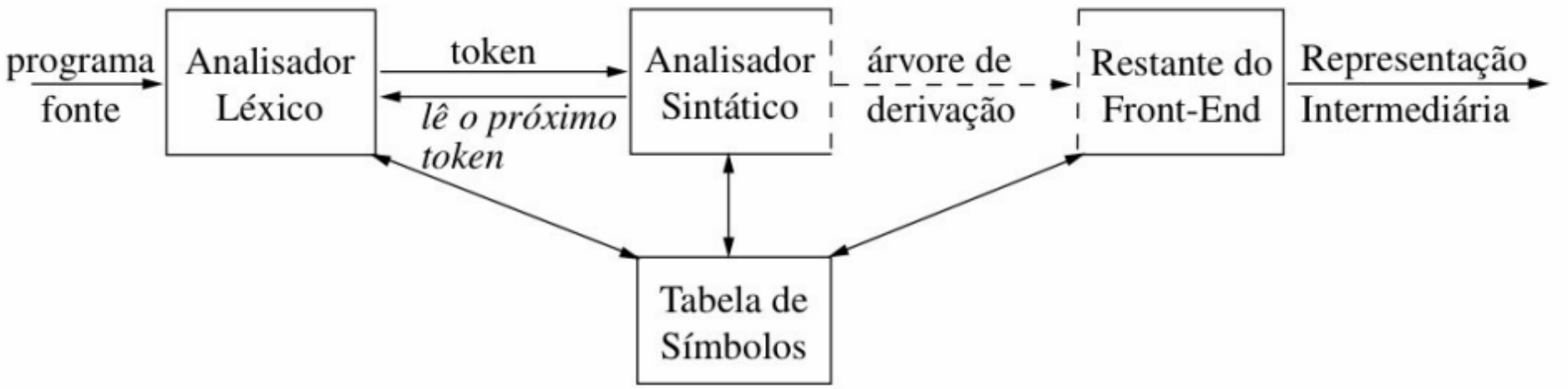

Figura 1.2: Posição do analisador sintático no modelo de compilador [Aho et al., 2007]

Gramáticas representativas

- As palavras-chave, como **while** ou **if** são fáceis de analisar;
- Precisamos representar claramente as expressões;
- A gramática da Figura 9 representa análise sintática descendente, pois possui recursividade à esquerda.

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

Figura 1.3: Gramática que possui associatividade e precedência [Aho et al., 2007]

Análise descendente

- É possível remover a recursividade do exemplo da Figura 9;
- Ao remover a recursividade, aplicamos a análise sintática descendente.

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

Figura 1.4: Variação da gramática da Figura 9 sem recursividade à esquerda [Aho et al., 2007]

Tratamento de erros

- O compilador deve auxiliar o rastreio de erros;
- Alguns tipos de erro mais comuns:

Erro léxico Erro de ortografia em palavras-chave, identificadores ou operadores;

Erro sintático Incluem ponto e vírgula mal colocado e chave faltando. Geram uma árvore de parsing incorreta;

Erro semântico Erro entre operador e operando, por exemplo;

Erro lógico Mais difícil de detectar, acontece por erro do programador. Um exemplo é a utilização de $=$ no lugar de $==$.

Recuperação de erro

- Algumas estratégias para recuperação de erros:

  Modo pânico Descarta um símbolo de entrada de cada vez até que um delimitador de sincronismo seja encontrado;

  Nível de frase Aplica uma correção local ao restante da entrada, como por exemplo adição de ; ou substituição de , por ;

  Global Tenta alterar a árvore de parsing até obter uma entrada correta.

Definição

- Definição: representação gráfica de uma derivação que filtra a ordem na qual as produções são aplicadas.
- Cada nó interior da árvore de derivação representa a aplicação de uma produção;
- Pequeno algoritmo:
  1. O nó interior é rotulado como o não terminal A do lado esquerdo da produção;
  2. Dos filhos dos nós são rotulados, da esquerda para a direita, pelos símbolos do corpo da produção pelo qual A foi substituído.

Exemplo de derivação

$$E \rightarrow -E \rightarrow -(E) \rightarrow -(E+E) \rightarrow -(id+E) \rightarrow -(id+id) \qquad (1)$$

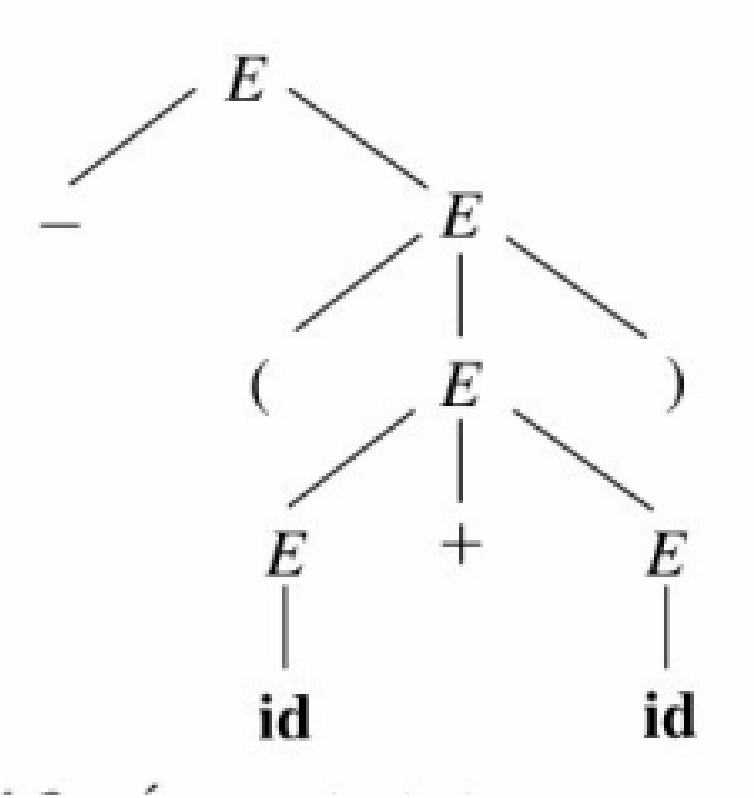

Figura 2.1: Árvore de derivação para a cadeia $-(id+id)$ do exemplo 1 [Aho et al., 2007]

Ambiguidade

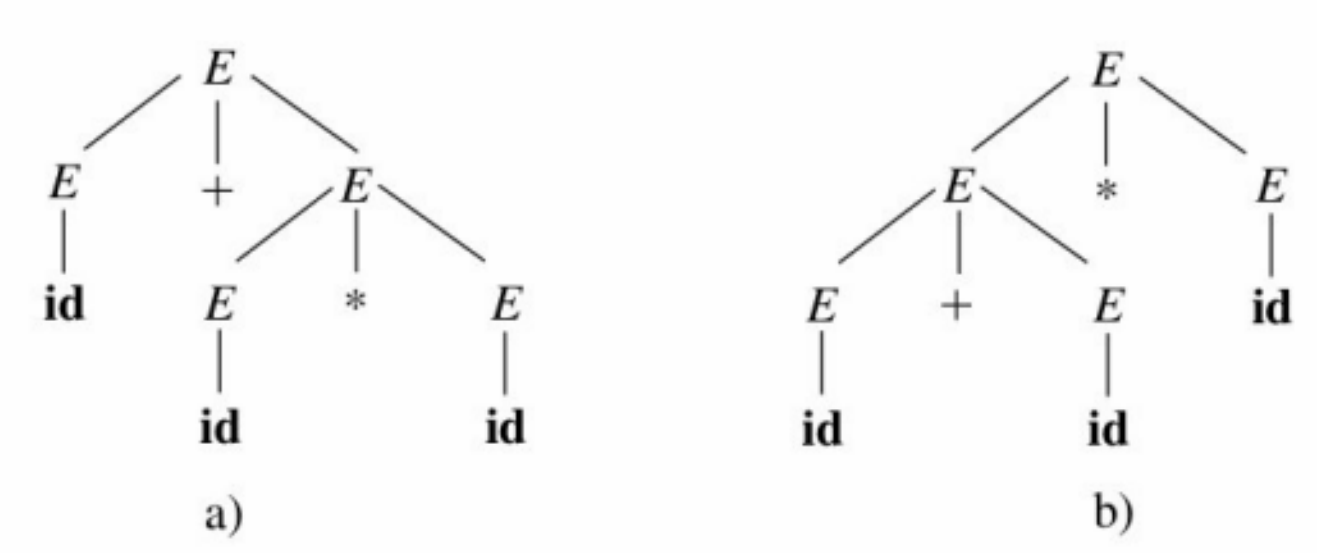

Figura 2.2: Árvore de derivação para a cadeia $id + id * id$ do exemplo da Figura 9 [Aho et al., 2007]

OBRIGADO!!!
PERGUNTAS???


Aho, A., Lam, M., Sethi, R., and Ullman, J. (2007).
Compiladores–Princıpios Técnicas e Ferramentas.
Pearson, 2a. edition.


de Alencar Price, A. M. and Toscani, S. S. (2000).
Implementação de linguagens de programação: compiladores.
Sagra-Luzzatto.