Parsing

Eduardo Ferreira dos Santos

Ciência da Computação
Centro Universitário de Brasília – UniCEUB

Abril, 2017

Sumário

1. Introdução
2. Gramáticas
3. Árvore de parsing
4. Exercícios

1 Introdução

2 Gramáticas

3 Árvore de parsing

4 Exercícios

Linguagens e autômatos

- Linguagens formais são muito importantes em Ciência da Computação;
  - Definição de linguagem de programação;
  - Definição de algoritmo.
- Linguagens regulares;
- Linguagens livres de contexto.

Linguagem Regular

Uma linguagem $\mathcal{L} \subseteq \Sigma^*$ é Regular se existe um AFD A tal que $\mathcal{L}(A) = \mathcal{L}$.

Figura 1.1: Definição de linguagem regular [Rezende, 2016]

Linguagens regulares

- Muitas linguagens não são regulares;
- *Strings* com parênteses balanceados não são regulares.

$$\{(^i)^i | i \geq 0\}$$

- Qual a capacidade expressiva das linguagens regulares? [Schwarz et al., 2016]
- **Ex.:** Linguagens que calculam o módulo de um inteiro;
  - Intuição: um autômato finito que seja executado por tempo suficiente deve repetir estados;
  - O autômato finito não é capaz de armazenar a quantidade de vezes que passou por um estado específico.

Funcionalidades do parser

Entrada Sequência de tokens da análise léxica.

Saída Árvore de *parsing* do programa. [1]

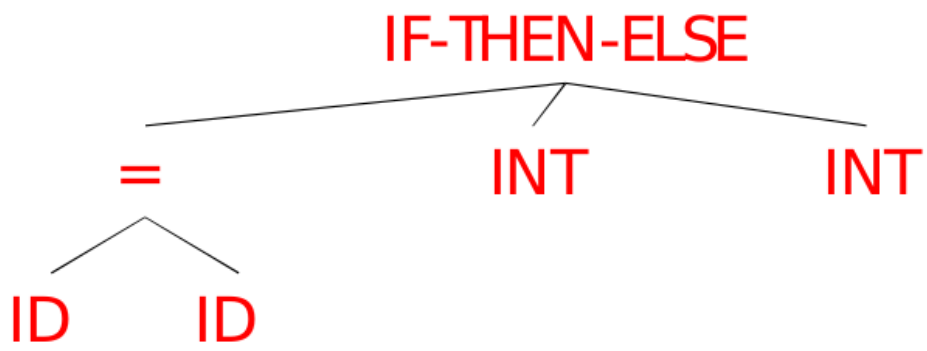
- Exemplo:
  - Decaf:

    if x = y then 1 else 2

  - Entrada do parser:

    IF ID = ID THEN INT ELSE INT

  - Saída do parser:



[1] Alguns *parsers* nunca produzem uma árvore de *parsing*

Comparativo

- A saída da análise léxica representa a entrada do *parser*;
- As duas etapas trabalham em conjunto.

Fase	Entrada	Saída
Lexer	Sequência de caracteres	Sequência de tokens
Parser	Sequência de tokens	Árvore de parsing

Tabela 1.1: Comparação com a análise léxica [Schwarz et al., 2016]

Papel

- Nem todas as sequências de caracteres são válidas para a linguagem;
- O *parser* deve ser capaz de representar a diferença entre as sequências válidas e inválidas;
- Algumas necessidades:
  - Uma linguagem capaz de descrever as sequências de caracteres válidas;
  - Um método para distinguir sequências válidas e inválidas.
- Uma notação natural para as necessidades expressivas do *parser* é a Gramática Livre de Contexto.

1 Introdução

2 Gramáticas

3 Árvore de parsing

4 Exercícios

Gramáticas

- Informalmente a gramática é um mecanismo de produção de palavras ou sentenças a partir de substituição de variáveis [Rezende, 2016];
- Palavras-chave:
  - Variáveis;
  - Símbolos terminais;
  - Produções (ou regras).

$$\begin{aligned} E &\rightarrow \epsilon \\ E &\rightarrow 0E1 \end{aligned} \tag{1}$$

- O exemplo da Equação 1 apresenta os seguintes elementos:
  - Variáveis: E;
  - Símbolos terminais: $\{0, 1, \epsilon\}$
  - Produções: $\{E \rightarrow \epsilon, E \rightarrow 0E1\}$

Definição [Rezende, 2016]

Uma Gramática Livre de Contexto é uma tupla $G = (V, \Sigma, R, S)$, onde:

V	conjunto finito de variáveis
Σ	alfabeto finito de símbolos terminas $V \cap \Sigma = \emptyset$
$S \in V$	variável inicial
$R \subseteq Vx(\Sigma \cup V)^*$	conjunto finito de produções

Escrevemos uma produção como $A \rightarrow \alpha$, onde $A \in V$ e $\alpha \in (\Sigma \cup V)^*$

Meta-notação

$\langle foo \rangle$	foo não é um terminal, ou seja, é uma variável
foo	(em **negrito**) significa que **foo** é um terminal, ou seja, um token ou parte de um token
$[x]$	zero ou uma ocorrência de x, ou seja, x é opcional. Perceba que o uso de colchetes entre aspas ('[' ']') representa um terminal.
x^*	uma ou mais ocorrências de x
x^+,	lista de ocorrências de x separadas por vírgula
$\{\}$	chaves grandes são utilizadas para agrupar elementos. Chaves entre aspas ('{' '}') são terminais.
$\|$	separa as alternativas (operador ou)

Tabela 2.1: Meta-notação para aplicação na gramática da linguagem Decaf [Amarasinghe and Rinard, 2010]

Exemplos

- O Exemplo da Equação 2 mostra a definição de tipo de dado (*type*) na linguagem Decaf.

$$\langle type \rangle \rightarrow \textbf{\textit{int}} | \textbf{\textit{boolean}} \tag{2}$$

- O exemplo da Equação 3 apresenta a definição de identificador na linguagem Decaf:

$$\langle id \rangle \rightarrow \langle alpha \rangle | \langle alpha\_num \rangle^* \tag{3}$$

Algoritmo

A ideia de realizar produções utilizando uma gramática pode ser representada através de um simples algoritmo [Schwarz et al., 2016]:

1. Comece uma cadeia de caracteres iniciando com o símbolo inicial S;
2. Substitua qualquer variável X na cadeia pelo lado direito de alguma produção:
$$X \rightarrow Y_1 ... Y_n$$
3. Repita o passo 2 até que não haja nenhuma variável na cadeia.

Linguagem da gramática

Considere uma gramática livre de contexto G que possui o símbolo inicial S. Dizemos que a linguagem $\mathcal{L}$ aceita por G é tal que:

$$\mathcal{L} = \mathcal{L}(G) = \{a_1 ... a_n S \rightarrow^* a_1 ... a_n \text{ e todo } a_i \text{ é um terminal}\}$$

- Terminais:
  - Não há nenhum símbolo para substitui-los. Por isso são terminais;
  - Uma vez gerados, os terminais são permanentes;
  - Os terminais devem ser os tokens da linguagem.

Exemplos

- Exemplo aritmético simples:

$$E \rightarrow E + E | E * E | (E) | id$$

- Alguns exemplos da linguagem Decaf

$\langle alpha \rangle \rightarrow$ **a** | **b** ... | **z** | **A** | **B** | ... | **Z** | **_**
$\langle digit \rangle \rightarrow$ **0** | **1** | **2** | ... | **9**
$\langle alpha\_num \rangle \rightarrow \langle alpha \rangle | \langle digit \rangle$
$\langle id \rangle \rightarrow \langle alpha \rangle \langle alpha\_num \rangle^*$

1 Introdução

2 Gramáticas

3 Árvore de parsing

4 Exercícios

Derivação

- Uma derivação é uma sequência de produções [Schwarz et al., 2016]:

$$S \rightarrow ... \rightarrow ... \rightarrow ...$$

- A derivação pode ser desenhada como uma árvore:
  - O primeiro símbolo é a raiz da árvore;
  - Para uma produção do tipo $X \rightarrow Y_1 ... Y_n$ adicione os filhos $Y_1 ... Y_n$ ao nó X.

Exemplo de derivação

- Gramática:

$$E \rightarrow E + E | E * E | (E) | id$$

- Cadeia:

$$id * id + id$$

Exemplo de derivação (cont.

$$
\begin{aligned}
& E \\
\rightarrow\ & E+E \\
\rightarrow\ & E*E+E \\
\rightarrow\ & id*E+E \\
\rightarrow\ & id*id+E \\
\rightarrow\ & id*id+id
\end{aligned}
$$

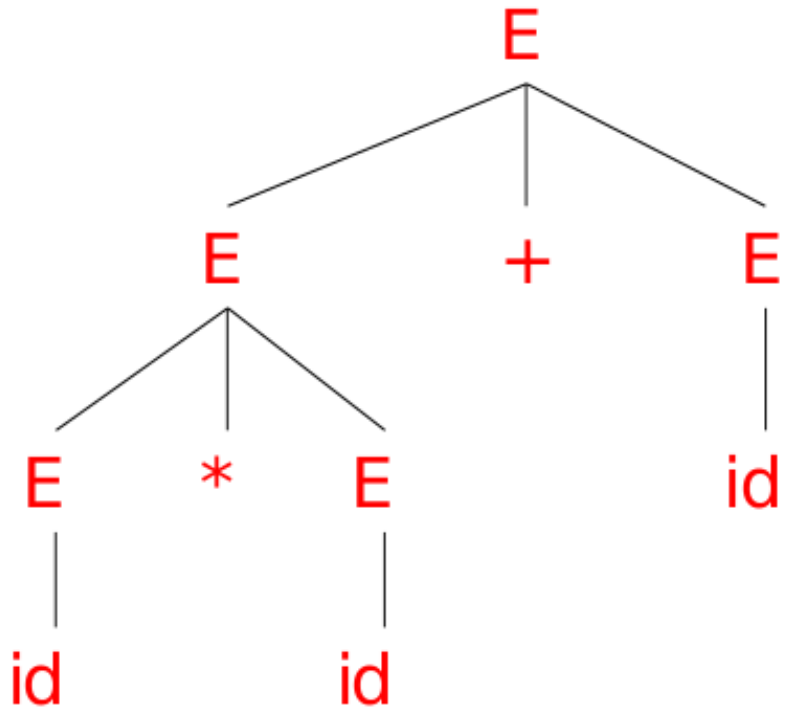


Figura 3.1: Exemplo simples de árvore de derivação aritmética [Schwarz et al., 2016]

Notas sobre o parsing

- Uma árvore de parsing possui:
  - Terminais nas folhas;
  - Não terminais em todos os outros nós interiores.
- Uma leitura em-ordem da árvore traz de volta a cadeira original;
- A árvore de parsing mostra a associação entre as operações. A cadeia original não;
- Problema da ambiguidade;
- Algoritmos preditivos;
- Principais algoritmos de árvore: derivação à direita e à esquerda.

1 Introdução

2 Gramáticas

3 Árvore de parsing

4 Exercícios

Exercício 2

Listing 1: Exemplo do exercício 02 [Aho et al., 2007]

```
Here is a photo of <b>my house</b>:
<p><img src = "house.gif"><br>
Veja <a href = "morePix.html">More Pictures</a> if you
liked that one</p>
```

1. Divida o documento HTML nos lexemas apropriados;
2. Quais lexemas devem ter valores léxicos associados?
3. Quais são esses valores?

Exercício 3

- Considere a linguagem SQL, formada por quatro instruções: SELECT, INSERT, UPDATE e DELETE.

1. Escreva uma gramática para tratar essa linguagem;
2. Descreva os tokens constituintes da linguagem;
3. Implemente um analisador léxico reduzido para a linguagem SQL, considerando somente as quatro instruções básicas.

OBRIGADO!!!
PERGUNTAS???


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


Amarasinghe, S. and Rinard, M. (2010).
Computer language engineering.
Disponível em `http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-035-computer-language-engineering-spring-2010/` Acessado em 02/08/2016.


Rezende, P. (2016).
Notas de aula do Prof. Pedro Rezende.
Disponível em: `http://cic.unb.br/~rezende/tc.html` Acessado em 14/03/2016.


Schwarz, K., Papadakis, H., and Mittal, R. (2016).
Compilers.

Disponível em http://web.stanford.edu/class/cs143/ Acessado em 30/09/2016.