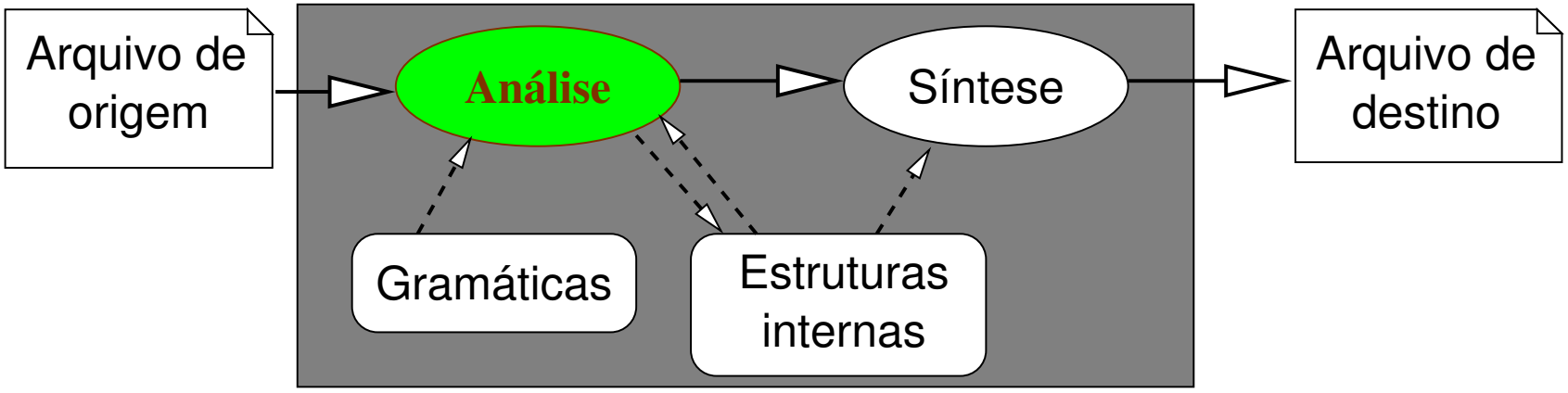
Representação de linguagens

Ivan Ricarte

2008

Sumário

Motivação


Arquivo de origem
Análise
Síntese
Arquivo de destino
Gramáticas
Estruturas internas

Etapa de análise

- Objetivos
  - Extrair do código fonte os elementos básicos do programa
  - Obter estrutura lógica do programa e de suas expressões

Análise léxica reconhecimento, a partir dos caracteres, das palavras básicas do "vocabulário" do programa

- palavras-chaves da linguagem, operadores, identificadores, constantes

Análise sintática reconhecimento da estrutura de comandos e do programa como um todo

- definições, declarações, comandos simples, blocos de comandos, programas

Análise semântica verificar relacionamentos entre fragmentos do código

Etapa de análise

Análise léxica e análise sintática

- Demandam o reconhecimento de seqüências de símbolos válidas

  Léxica seqüência de caracteres que compõe uma palavra

  Sintática seqüência de palavras que compõe uma sentença

- Há necessidade de um mecanismo para especificar quais seqüências são válidas

Mecanismo formal linguagens e gramáticas

Conjuntos

Definição

Um conjunto é um agregado de elementos (sem repetição) tratado como um todo

Representação

- Pela enumeração dos elementos: $\{1, 2, 3, 4, 5, \ldots\}$
- Por uma proposição lógica:
  $\{x | x \text{ é um número inteiro positivo}\}$
- Por um nome: $\mathbb{N}$

Conjunto sem elementos: *conjunto vazio* — $\{\}$ ou $\emptyset$

Relações entre conjuntos e seus elementos

$$A = \{1, 2, 3\}$$

Relação entre elementos e conjuntos: $\in$ (pertinência)

$$1 \in A$$

$$7 \notin A$$

Relação entre conjuntos: $\subset \subseteq$ (continência)

$$\{1, 2, 3\} \subseteq A$$

$$\{1, 2\} \subset A$$

$$\{2, 3, 4\} \not\subset A$$

Operações de conjuntos

Dados dois conjuntos B e C,

União

$$B \cup C = \{x \mid x \in B \vee x \in C\}$$

Interseção

$$B \cap C = \{x \mid x \in B \wedge x \in C\}$$

Diferença

$$B - C = \{x \mid x \in B \wedge x \notin C\}$$

Partição

Definição

Uma partição é um conjunto de subconjuntos tal que

- Todos os elementos do conjunto original pertencem a algum subconjunto da partição; e
- Nenhum elemento do conjunto original pertence a mais de um subconjunto

Sejam $D_1 \subset D$ e $D_2 \subset D$, então $\{D_1, D_2\}$ é uma partição de D se $D_1 \cup D_2 = D$ e $D_1 \cap D_2 = \emptyset$

Linguagens

Alfabeto

Linguagens definidas a partir de um conjunto de símbolos

Alfabeto é o conjunto de símbolos válidos de uma linguagem

- Alfabeto da língua portuguesa: $\{a, b, c, \ldots, z\}$
- Alfabeto da linguagem de máquina: $\{0, 1\}$

Linguagens

Strings

String é uma seqüência (concatenação) de símbolos de um alfabeto

- A string vazia (nenhum símbolo) é denotada pelo símbolo ε

- Se $\mathbb{A}$ é um alfabeto, então:

  $\mathbb{A}^*$ (clausura de $\mathbb{A}$) é o conjunto de todas as strings que podem ser formadas a partir de símbolos de $\mathbb{A}$

  $\mathbb{A}^+$ é $\mathbb{A}^* - \varepsilon$

Linguagens

Exemplo

Seja $B = \{0, 1\}$ o alfabeto para a linguagem

$$L = \{0^n 1^n \mid n \geq 0\}$$

onde b^n representa uma seqüência de n ocorrências do símbolo b.
Quais das seguintes strings são válidas nesta linguagem?

- 01
- 010
- 0110
- 0011
- ε

Descrição de linguagens: Gramáticas

Uma gramática define regras para a formação de strings válidas em uma linguagem

Produções conjunto de regras de formação de strings

Alfabeto conjunto de símbolos da linguagem e um conjunto de símbolos auxiliares

Símbolos terminais símbolos da linguagem

Símbolos não-terminais símbolos auxiliares

- Um símbolo não-terminal é especial — indica o ponto de partida para formar strings nessa linguagem

Gramáticas

Definição formal

$$G = (V_T, V_N, \mathbb{P}, S_i)$$

- V_T alfabeto dos **símbolos terminais**
- V_N alfabeto dos **símbolos não-terminais**
- $\mathbb{P}$ conjunto de regras ou **produções**, expressos na forma $\alpha \rightarrow \beta$, onde $\alpha \in V^+$ e $\beta \in V^*$
- $S_i \in V_N$ **símbolo sentencial**, **símbolo não-terminal inicial** ou **axioma**: o ponto de partida na produção de qualquer sentença na linguagem

Produções

Uma produção

$$E \rightarrow D$$

representa que o símbolo ou seqüência de símbolos E (o lado esquerdo) pode ser substituído pelo símbolo ou seqüência de símbolos D (o lado direito) no processo de formação de uma string a partir do símbolo sentencial.

Produções

Exemplo

Na gramática

$$G_1 = (\{0, 1\}, \{Z\}, \{Z \to 0Z1, Z \to \varepsilon\}, Z)$$

as duas produções são:

- $Z \to 0Z1$ onde ocorre o símbolo Z é possível substituí-lo pela seqüência $0Z1$
- $Z \to \varepsilon$ onde ocorre o símbolo Z é possível substituí-lo pelo símbolo ε (ou seja, eliminá-lo)

Derivações

Derivação

É a substituição do lado esquerdo de uma produção de uma gramática pelos símbolos do lado direito

$$Z \Rightarrow 0Z1 \Rightarrow 01$$

Resultado de derivação pode ser

Forma sentencial Qualquer seqüência de símbolos, terminais ou não-terminais, que pode ser derivada a partir do símbolo sentencial de uma gramática

Sentença Forma sentencial composta exclusivamente por símbolos terminais

Derivações

Definições

Dadas duas formas sentenciais γ e δ

- $\gamma \Longrightarrow \delta$ δ é imediatamente derivável de γ pela aplicação de uma única produção
- $\gamma \overset{+}{\Longrightarrow} \delta$ δ é derivável de γ pela aplicação de uma ou mais produções

Em G_1

- $0Z1$ é imediatamente derivável de Z
- 01 é imediatamente derivável de $0Z1$
- 01 é derivável de Z

Derivações

Reconhecimento de sentenças válidas

Sentença σ faz parte da linguagem L se há seqüência de derivações a partir do símbolo sentencial S que leve à σ:

$$S \stackrel{+}{\Longrightarrow} \sigma$$

- Na gramática $G_1 = (\{0, 1\}, \{Z\}, \{Z \rightarrow 0Z1, Z \rightarrow \varepsilon\}, Z)$, quais das seguintes strings são válidas?
  - 01
  - 010
  - 0110
  - 0011
  - ε

Reconhecimento de sentenças

Exemplos

Em $G_1 = (\{0,1\}, \{Z\}, \{Z \to 0Z1, Z \to \varepsilon\}, Z)$

01 $Z \Rightarrow 0Z1 \Rightarrow 01$ (válida)

010 $Z \Rightarrow 0Z1 \Rightarrow ?$ (inválida)

0110 $Z \Rightarrow 0Z1 \Rightarrow 00Z11 \Rightarrow 0011$ (válida)

0011 $Z \Rightarrow 0Z1 \Rightarrow ?$ (inválida)

ε $Z \Rightarrow \varepsilon$ (válida)

Classificação de gramáticas

Gramáticas podem ter produções $\alpha \rightarrow \beta$ com distintos graus de complexidade em seu formato:

- Quantos símbolos no lado esquerdo?
- Como os símbolos que aparecem no lado esquerdo podem aparecer no lado direito (recursividade)?

Quanto menos restrições houver ao formato das produções, maior o poder de expressão da gramática

- Pode representar linguagens mais complexas
- Procedimento de reconhecimento de sentenças mais complicado

Classificação de Chomsky

- Chomsky (1928–) estudou gramáticas formais e propôs a seguinte classificação:
  - Tipo 0 Gramáticas recursivamente enumeráveis ou Gramáticas com estrutura de frase
  - Tipo 1 Gramáticas sensíveis ao contexto
  - Tipo 2 Gramáticas livres de contexto
  - Tipo 3 Gramáticas regulares
- A mesma classificação é aplicada às linguagens
  - Linguagem do tipo *n* é descrita por uma gramática do tipo *n*

Classificação de Chomsky

Gramática recursivamente enumerável

- Não impõe nenhuma restrição à forma das produções
- Pelo menos uma das produções da gramática pode ter a forma $\alpha \rightarrow \beta$ na qual a quantidade de símbolos em α é maior que a quantidade de símbolos em β
  - Uma derivação pode reduzir a quantidade de símbolos na forma sentencial

Classificação de Chomsky

Gramática sensível ao contexto

- As produções $\alpha \rightarrow \beta$ devem obedecer à restrição

$$| \alpha | \leq | \beta |$$

- Nenhuma derivação pode reduzir a quantidade de símbolos em uma forma sentencial

Classificação de Chomsky

Gramática livre de contexto

- As produções $\alpha \rightarrow \beta$ devem obedecer à restrição

$$| \alpha | = 1$$

  - O lado esquerdo da produção tem um único símbolo não-terminal
- Poder de expressão suficiente para capturar principais aspectos dos comandos das linguagens de programação

Classificação de Chomsky

Gramática regular

- Se produção é recursiva, não contém auto-incorporação
  - Produções recursivas têm formato

$$A \rightarrow A\beta$$

ou

$$A \rightarrow \beta A$$

para uma seqüência de símbolos β qualquer

Notações alternativas

- Além da descrição formal por meio de gramáticas, linguagens podem ser descritas por meio das seguintes representações:

  Expressões regulares mesmo poder de expressão de gramáticas regulares

  Backus-Naur Form mesmo poder de expressão de gramáticas livres de contexto

  Diagramas sintáticos mesmo poder de expressão de gramáticas livres de contexto

Expressões regulares

Definição

Uma expressão regular é definida pelas seguintes regras:

1. A *string* vazia é uma expressão regular.
2. Dado um alfabeto A, então um elemento de A é uma expressão regular em A.
3. Se P é uma expressão regular em A, então a repetição de 0 ou mais ocorrências de P, denotada P^*, também é uma expressão regular em A.
4. Se P e Q são expressões regulares em A, então a seqüência de P e Q, denotada PQ, também é uma expressão regular em A.
5. Se P e Q são expressões regulares em A, então a alternativa entre P e Q, denotada $P \mid Q$, também é uma expressão regular em A.

Expressões regulares

Exemplos

Para o alfabeto $A = \{a, b\}$, são expressões regulares:

- $aa*$ strings com pelo menos uma ocorrência do símbolo *a*, como *a*, *aa* e *aaaaa*. Como $*$ tem a maior precedência, esta expressão é diferente de $(aa)^*$, que representa strings com zero ou mais pares de símbolos *a*.
- $a(a|b)*b$ strings que começam com o símbolo *a* e terminam com o símbolo *b* e têm qualquer quantidade dos símbolos *a* ou *b* no meio. As strings *ab*, *aab*, *abb*, *aabb* e *abab* são válidas segundo essa expressão.
- $ba|a*b$ a string *ba* ou uma string com qualquer quantidade (até mesmo 0) do símbolo *a* terminada pelo símbolo *b*, como *b*, *ab* ou *aab*.

Expressões regulares

Relação com gramáticas regulares

Como um operador em uma expressão regular relaciona-se às produções de uma gramática regular equivalente?

Concatenação: tem a mesma representação nas duas notações, ou seja, um símbolo após o outro.

Alternativa: o operador $|$ de uma expressão regular corresponde, na gramática, a produções alternativas para um mesmo símbolo não-terminal.

Repetição: o operador de repetição $*$ de uma expressão regular corresponde, na gramática, a uma produção recursiva. Se a expressão permite zero ocorrências do padrão, então a regra que estabelece o fim da recursão leva à string vazia.

Relação com gramáticas regulares

Exemplos

- $aa*$ equivale a uma gramática com produções

$$S \rightarrow aS$$
$$S \rightarrow a$$

- $a(a|b)*b$ equivale a uma gramática com produções

$$S \rightarrow aQb$$
$$Q \rightarrow aQ$$
$$Q \rightarrow Qb$$
$$Q \rightarrow \varepsilon$$

Em ambos os casos, S é o símbolo sentencial

Backus-Naur Form

- Define uma notação textual compacta para as produções de uma gramática livre de contexto
- Notação BNF básica:

$::=$	produção ($\rightarrow$)
$< \cdots >$	símbolo não-terminal
$\mid$	alternativa

- Exemplo:

$$< S > ::= a < Q > b$$
$$< Q > ::= a < Q > \mid < Q > b \mid \text{” ”}$$

BNF estendido

- Extensões que foram propostas à notação BNF padrão:
  - $[\,\cdot\,]$ opcional (zero ou uma ocorrência)
  - $(\cdot \mid \cdot)$ fatoração
  - $\cdot*$ repetição (zero ou mais ocorrências)
- Limitação: símbolos usados como metacaracteres não podem fazer parte do alfabeto da linguagem

Diagramas sintáticos

- Notação gráfica para gramáticas livres de contexto, com um diagrama para cada símbolo não-terminal
- Símbolos não-terminais: retângulos
- Símbolos terminais: círculos ou elipses

Diagrama sintático

Exemplo

- Regra $<\text{expr}>::=(<\text{expr}>)|<\text{expr}>+<\text{expr}>$ em BNF equivale a

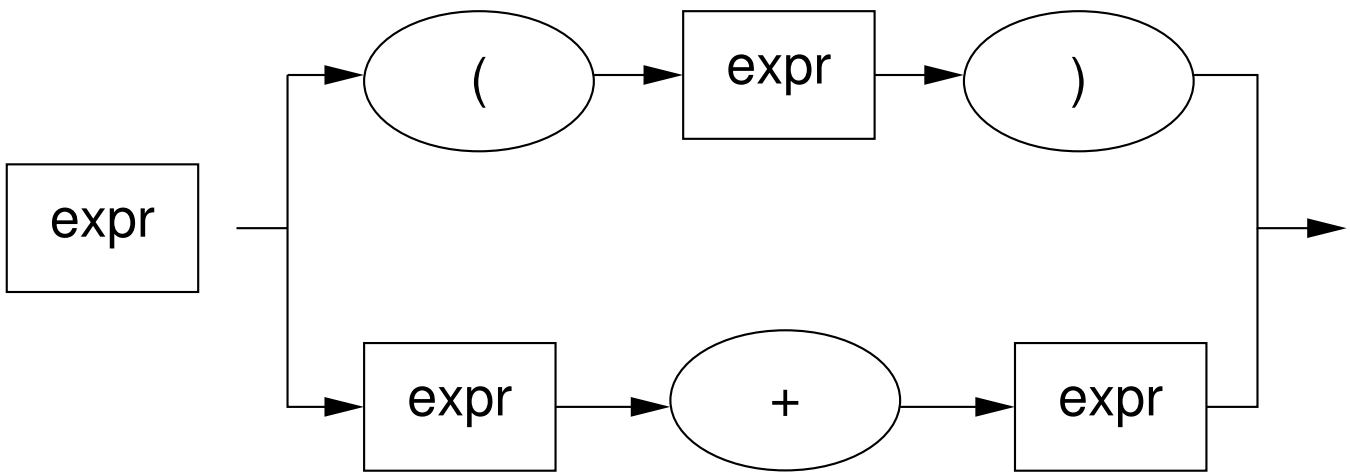
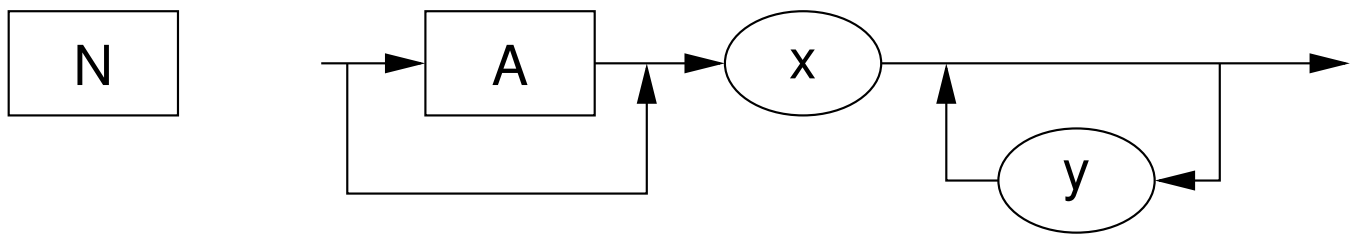


Diagrama sintático

Exemplo

- Regra $<\mathrm{N}>::=[<\mathrm{A}>]\mathrm{xy}*$ em BNF estendido equivale a



Como essas definições são utilizadas

- Associada a uma gramática formal há uma definição de um autômato que reconhece strings na linguagem descrita por essa gramática
- Símbolos elementares de uma linguagem de programação são descritos por gramáticas ou expressões regulares
  - Embasamento formal para a análise léxica
  - Reconhecimento por meio de autômatos finitos
- Sentenças de uma linguagem de programação são descritas por gramáticas livres de contexto
  - Embasamento formal para a análise sintática
  - Reconhecimento por meio de autômatos de pilha

Sugestões de leitura (Web)

- “O conjunto dos números naturais não tem o 0?”
  `http://en.wikipedia.org/wiki/Natural_number`
- Biografias na Wikipedia:

  Noam Chomsky `http://pt.wikipedia.org/wiki/Chomsky`

  John Backus `http://pt.wikipedia.org/wiki/John_Backus`

  Peter Naur `http://pt.wikipedia.org/wiki/Peter_Naur`