UDESC

Compiladores

Prof. Ricardo

ricardo.martins@udesc.br

UDESC

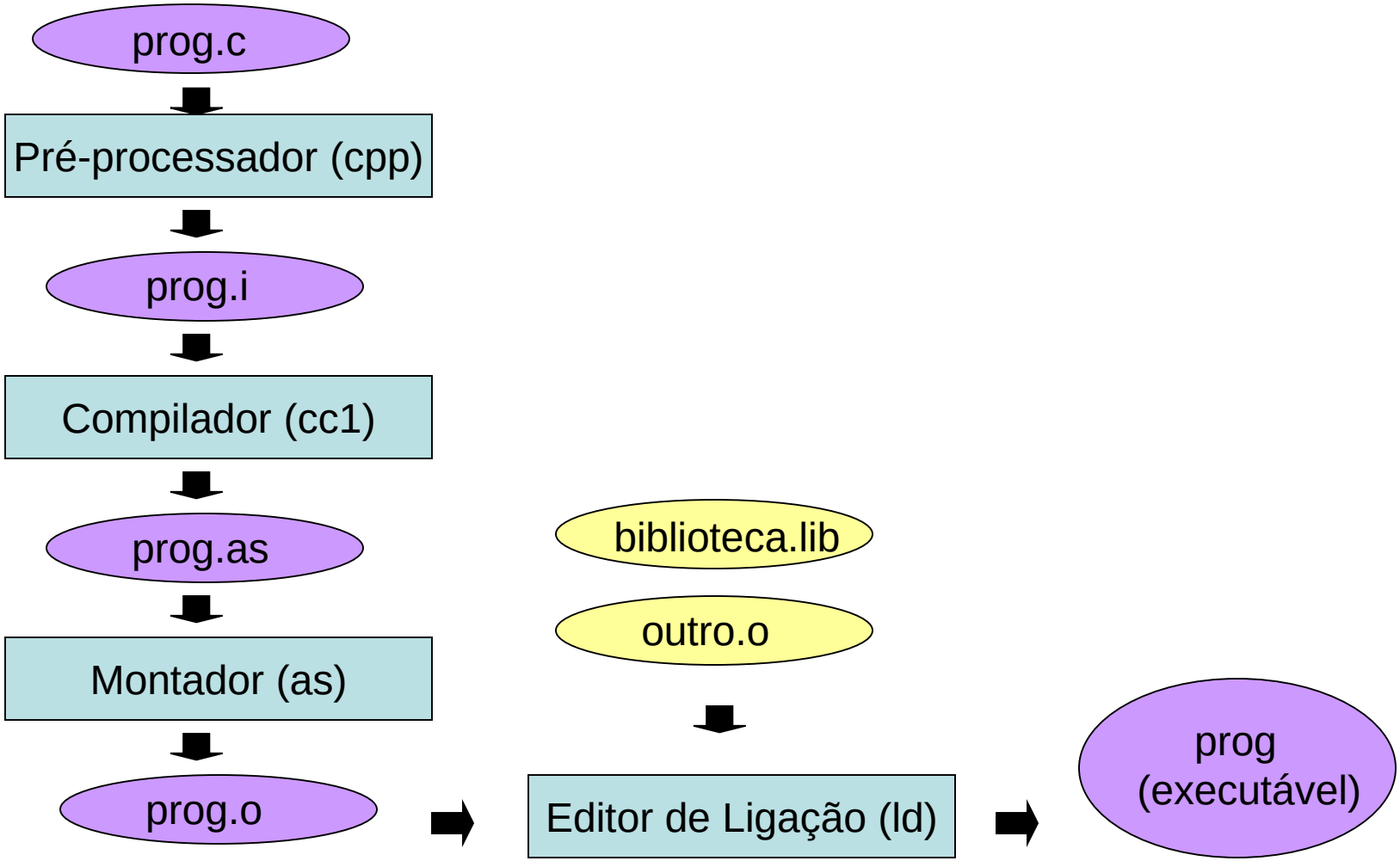
Bibliografia Recomendada

Aho, Alfred V.; Lam, Monica S.; Sethi, Ravi; Ullman, Jeffrey D.; Compiladores: Princípios, Técnicas e Ferramentas. Pearson.

Cooper, Keith D.; Torczon, Linda.; Construindo compiladores. Elsevier.

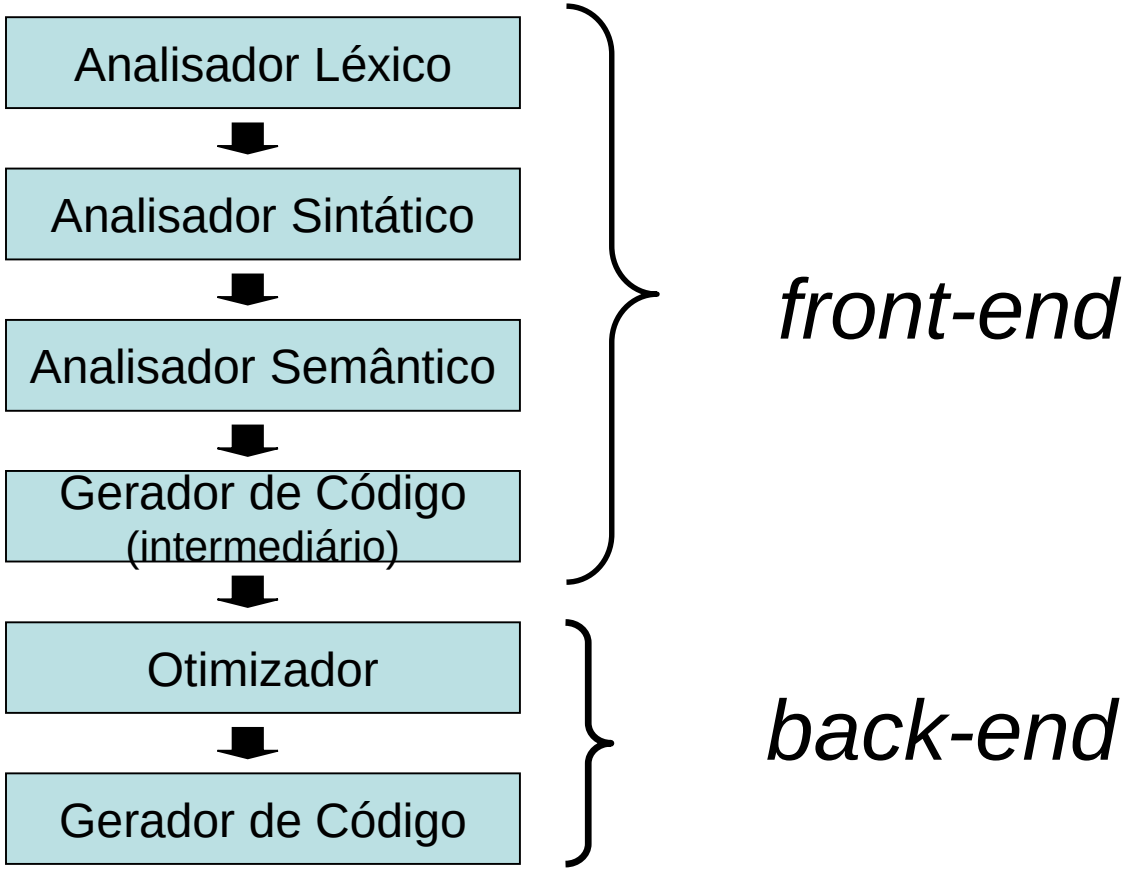
Bryant, Randal E.; O'Hallaron, David R.; Computer Systems: A Programmer's Perspective. Prentice Hall.

Introdução
(Construção de um Executável)



UDESC

Introdução
(Fases de um Compilador)



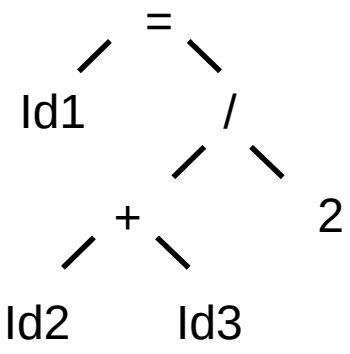
Introdução

final = (nota1 + nota2) / 2;

Analisador Léxico

Id1 = (Id2 + Id3) / 2

Analisador Sintático



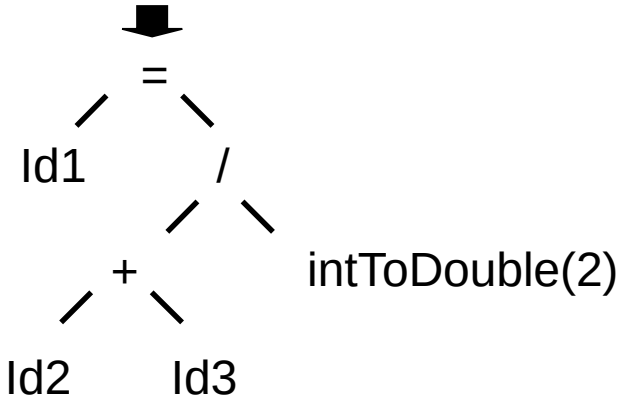
Árvore Sintática

Tabela de Símbolos

Id1	final	double	...
Id2	nota1	double	...
Id3	nota2	double	...
...			

Introdução

Analisador Semântico



Gerador de Código (intermediário)

temp1 = Id2 + Id3

temp2 = temp1 / 2.0

Id1 = temp2

Tabela de Símbolos

Id1	final	double	**...**
Id2	nota1	double	**...**
Id3	nota2	double	**...**
...			

UDESC

Análise Léxica

O Analisador Léxico (*scanner*) examina o programa fonte caractere por caractere agrupando-os em conjuntos com um significado coletivo (*tokens*):

- palavras chave (*if, else, while, int, etc*),
- operadores (+, -, *, /, ^, &&, *etc*),
- constantes (*1, 1.0, ‘a’, 1.0f, etc*),
- literais (*“Alo Mundo”, etc*),
- símbolos de pontuação (*; , {, }, etc*),
- labels.

UDESC

Análise Léxica

constanteInt $\rightarrow$ digito digito*
constanteDouble $\rightarrow$ digito digito*. digito*

digito $\in$ {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

X Representa uma sequência de zero ou mais X.*

UDESC

Análise Sintática

Verifica se as frases obedecem as regras sintáticas da linguagem:

Por exemplo, uma expressão pode ser definida como:

expressão + expressão

expressão – expressão

(expressão)

constante

UDESC

Gramáticas Livres de Contexto

Definidas por uma quádrupla (V_N, V_T, S, P), sendo:

V_N é um conjunto de símbolos não terminais (representam as construções sintáticas da linguagem).

V_T é um conjunto de símbolos terminais (*tokens* da linguagem).

$S \in V_N$ é o símbolo inicial da gramática.

P é um conjunto de regras de produção, pares ordenados representados na forma $\alpha \rightarrow \beta$, sendo $\alpha \in V_N$ e $\beta \in (V_N \cup V_T)^*$.

Gramáticas Livres de Contexto

<expr> → <expr> + <expr>
| <expr> – <expr>
| (<expr>)
| <const>

<const> → <const><const>
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 9

UDESC

Derivação

Verificar se uma frase faz parte da linguagem gerada por uma gramática, envolve sucessivas substituições dos símbolos que ocorrem do lado esquerdo da produção pela sua construção sintática correspondente.

Essa substituição é chamada derivação sendo normalmente denotada pelo símbolo $\Rightarrow$. E deve iniciar a partir do símbolo inicial da gramática.

UDESC

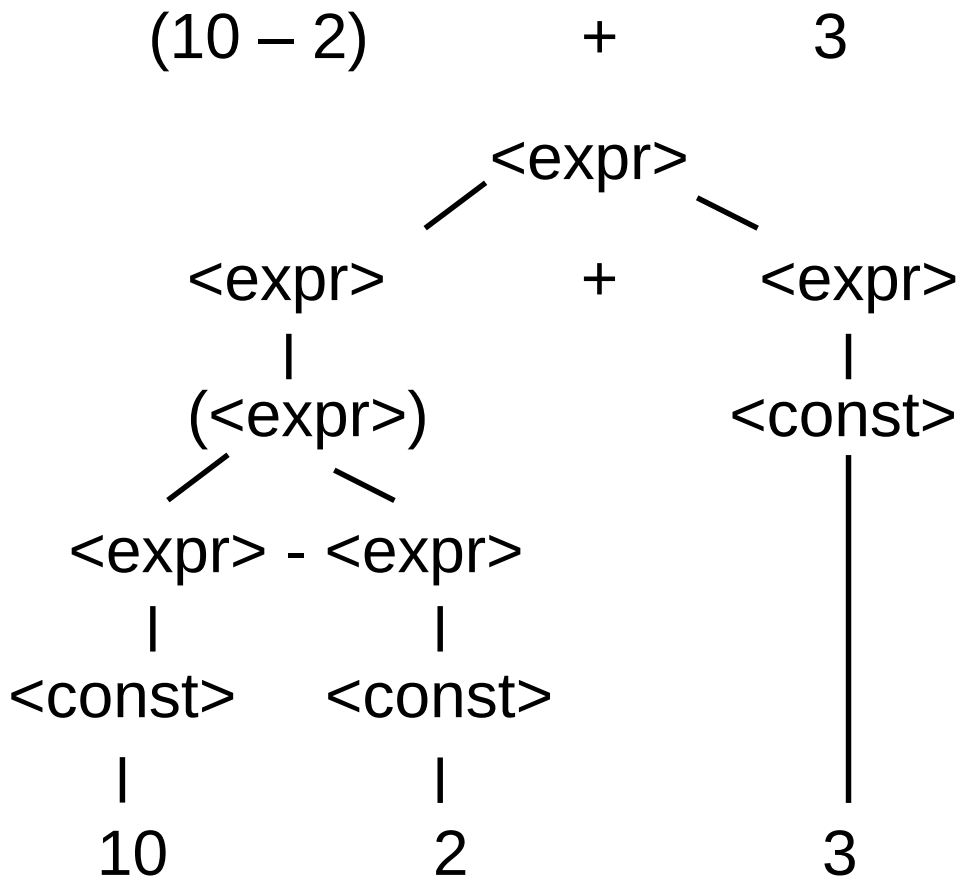
Derivação

<expressão>

⇒ <expr> + <expr>
⇒ (<expr>) + <expr>
⇒ (<expr> - <expr>) + <expr>
⇒ (<const> - <expr>) + <expr>
⇒ (<const><const> - <expr>) + <expr>
⇒ (1<const> - <expr>) + <expr>
⇒ (10 - <expr>) + <expr>
⇒ (10 - <const>) + <expr>

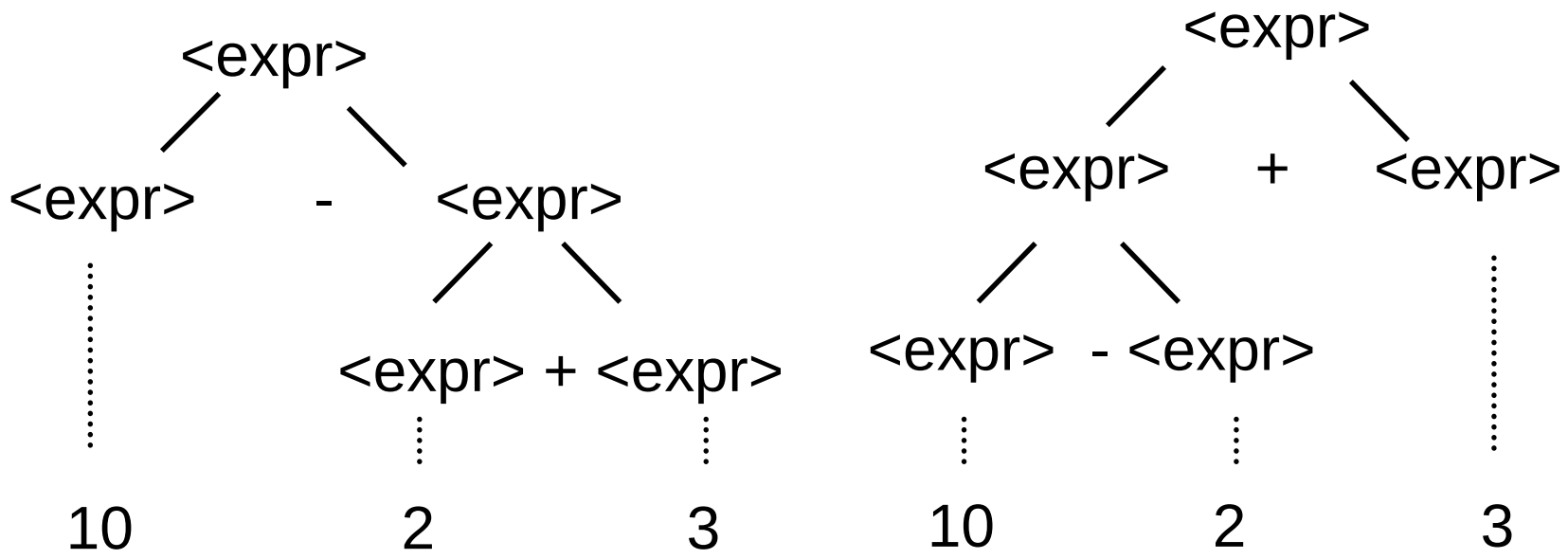
...
⇒ (10 - 2) + 3

Árvore Sintática de Derivação (*Parser Tree*)



Gramáticas Ambíguas

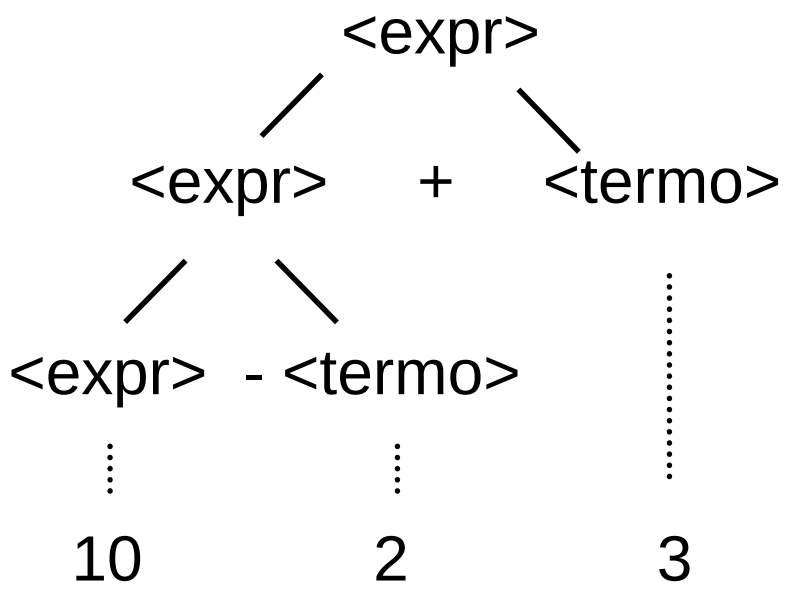
10 – 2 + 3



Gramáticas

<expr> → <expr> + <termo>
| <expr> - <termo>
| <termo>

<termo> → (<expr>)
| <const>



<expr>

⇒ <expr> + <termo>

⇒ <expr> - <termo> + <termo>

⇒ <termo> - <termo> + <termo>

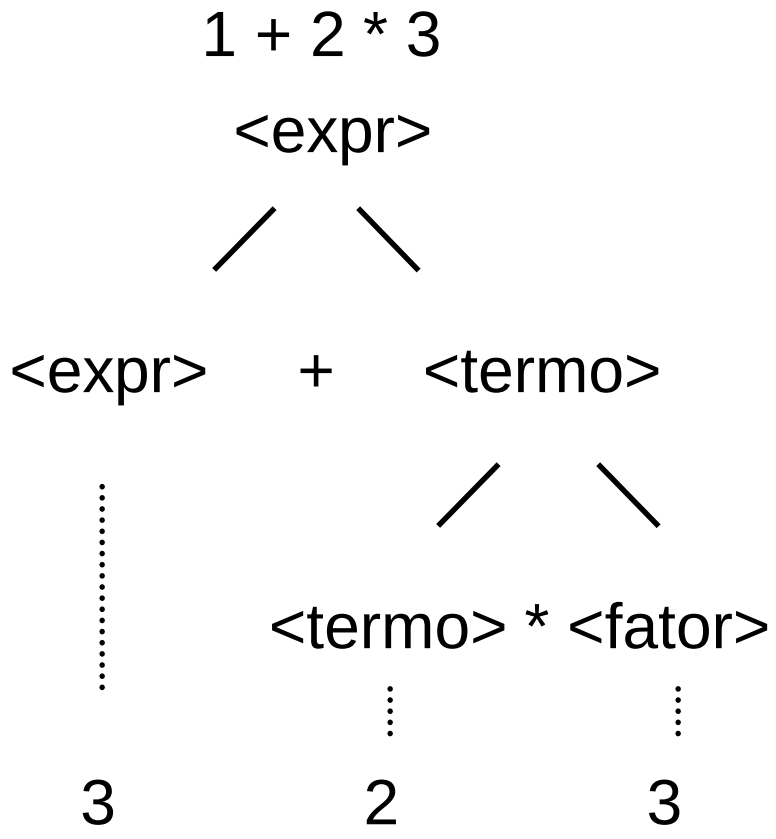
⇒ 10 – 2 + 3

Gramáticas

<expr> → <expr> + <termo>
| <expr> - <termo>
| <termo>

<termo> → <termo> * <fator>
| <termo> / <fator>
| <fator>

<fator> → (<expr>)
| <const>



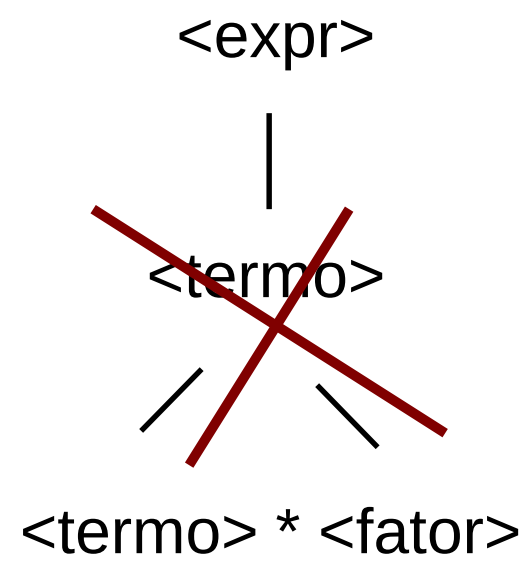
Gramáticas

<expr> → <expr> + <termo>
| <expr> - <termo>
| <termo>

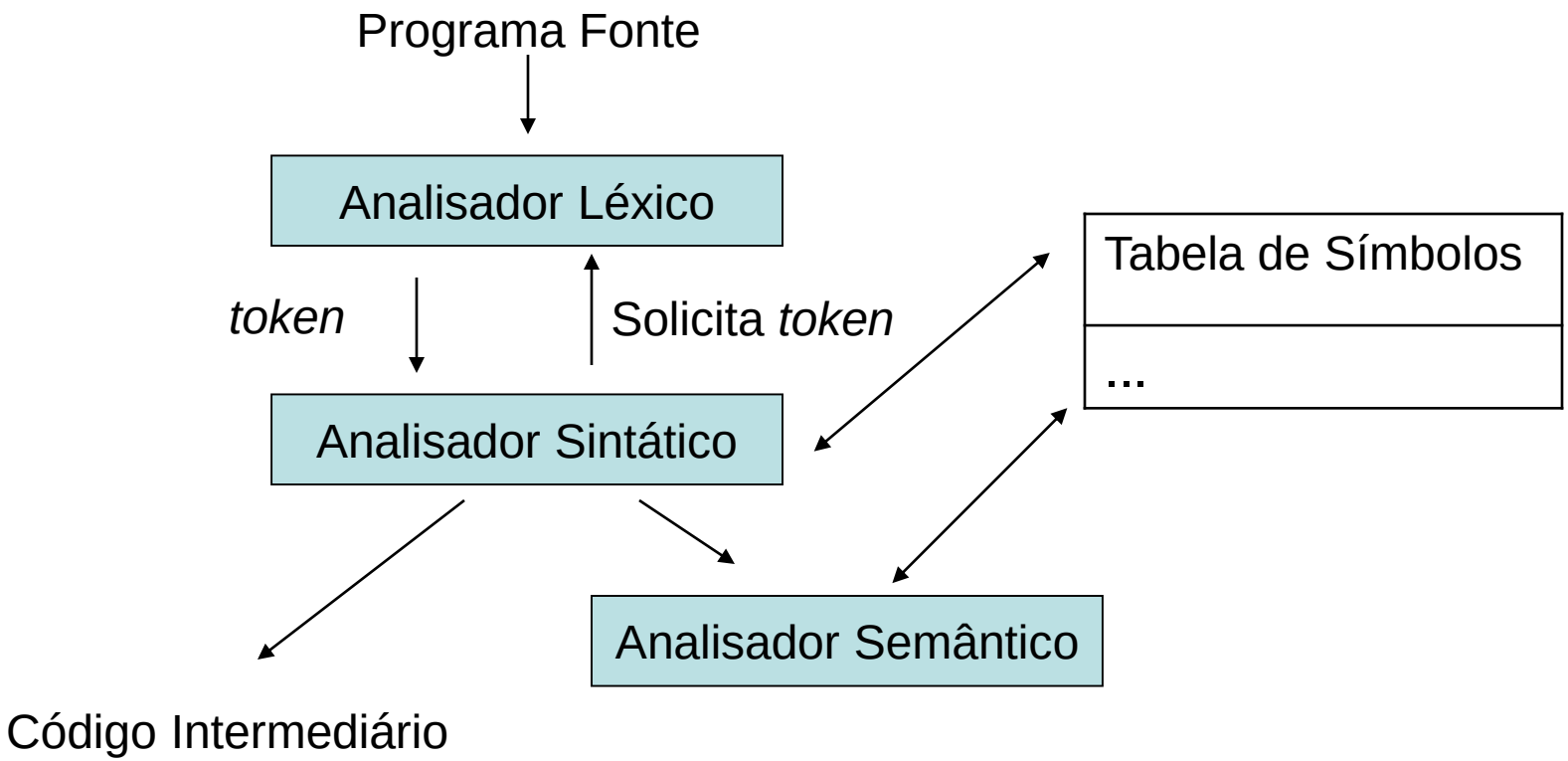
<termo> → <termo> * <fator>
| <termo> / <fator>
| <fator>

<fator> → (<expr>)
| <const>

1 + 2 * 3



Tradução Dirigida pela Sintaxe



UDESC

Gramáticas - Exercícios

1. Considerando a gramática apresentada anteriormente derive as expressões e apresente a árvore sintática correspondente:
   (1 + 2) * 3
   (1 – 2) + 3 * 4
2. Altere a gramática para incluir o operador unário -, esse operador deve ter precedência maior que todos os outros operadores.
3. Altere a gramática para que os operadores de adição, subtração, multiplicação e divisão tenham associatividade da direita para a esquerda.
4. Defina uma gramática para expressões aritméticas (operadores +, -, *, /) pós fixadas.

UDESC

Gramáticas

Dados 2 conjuntos independentes de símbolos:

- V_T – Símbolos terminais.
- V_N – Símbolos não terminais.

Uma gramática é definida como a quádrupla:

$$(V_N, V_T, S, P)$$

$S \in V_N$ é o símbolo inicial da gramática.

P é um conjunto de regras de reescrita na forma:

$\alpha \rightarrow \beta$, sendo: $\alpha \in (V_N \cup V_T)^* V_N (V_N \cup V_T)^*$

$$\beta \in (V_N \cup V_T)^*$$

Classificação de Gramáticas

- Irrestritas – nenhuma restrição é imposta
- Sensíveis ao Contexto - $|\alpha| \leq |\beta|$
- Livres de Contexto - $\alpha \in V_N$
  $\beta \in (V_N \cup V_T)^+$
- Regulares - $\alpha \in V_N$
  β tem *a* forma a ou *aB*, sendo
  $a \in V_T$ e $B \in V_N$

UDESC

Gramáticas Regulares

Uma gramática regular gera uma linguagem regular.

$C \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 9$
$\quad \mid 0C \mid 1C \mid 2C \mid 3C \mid 4C \mid 5C \mid 7C \mid 8C \mid 9C$

$C \rightarrow CC \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 9$

Linguagens Regulares

- Geradas a partir de uma gramática regular;
- Podem ser representadas por meio de uma expressão regular;
- Podem ser reconhecidas por Autômatos Finitos.

Considerando linguagens compostas por símbolos 0 e 1 podemos afirmar:

a linguagem $L_1 = \{0^n 1^n \mid n \geq 1\}$ não é regular;
a linguagem $L_2 = \{0^n 1^m \mid n \geq 1, m \geq 1\}$ é regular;

UDESC

Expressões Regulares

Maneira compacta de representar linguagens regulares. É composta de 3 operações. Sendo e_1 e e_2 expressões que geram respectivamente duas linguagens regulares L_1 e L_2:

- Concatenação: $e_1 e_2 = \{ xy \mid x \in L_1 \text{ e } y \in L_2 \}$
- Alternação: $e_1 | e_2 = \{ x \mid x \in L_1 \text{ ou } x \in L_2 \}$
- Fechamento: e_1^* = *zero ou mais ocorrências de* e_1.

É definida a precedência desses operadores como sendo: fechamento, concatenação, alternação (da maior precedência para a menor).

UDESC

Expressões Regulares

Exemplos:

identificador $\rightarrow$ (letra | _) (letra | digito | _)*

letra $\rightarrow$ a | b | ... | z | A | B | ... | Z

digito $\rightarrow$ 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

constInt $\rightarrow$ digito digito*

constDouble $\rightarrow$ digito digito*.digito* | . digito digito*

UDESC

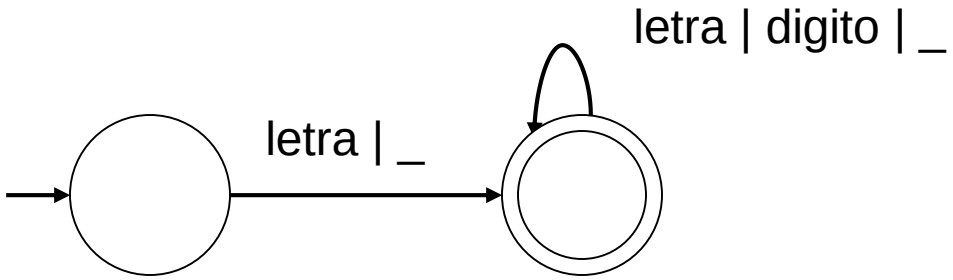
Autômato Finito

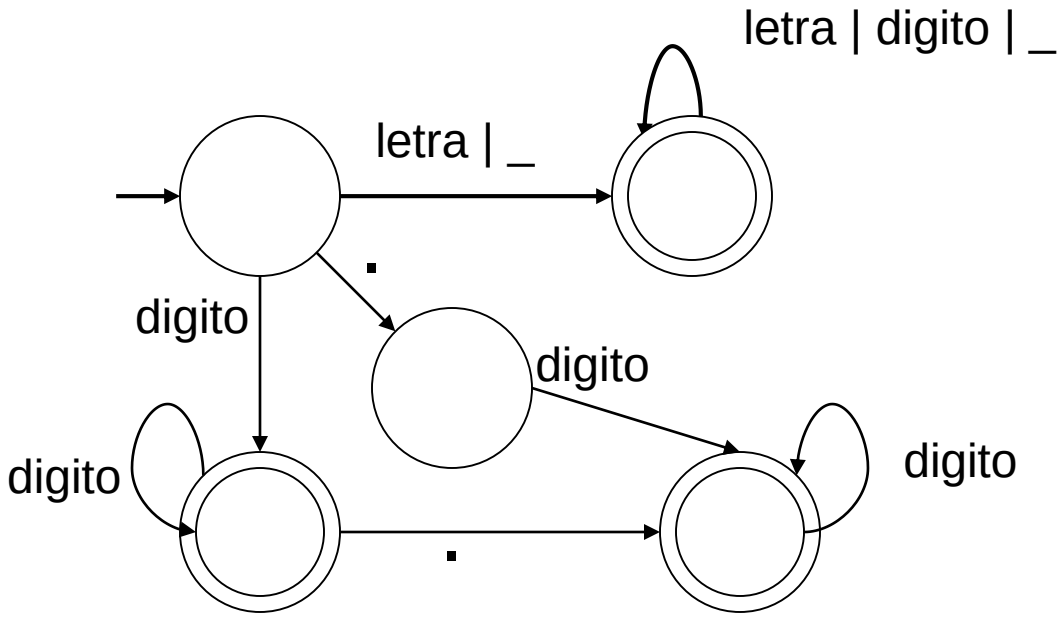
A linguagem gerada por uma gramática regular pode ser reconhecida por um autômato finito.

Um autômato finito consiste em:

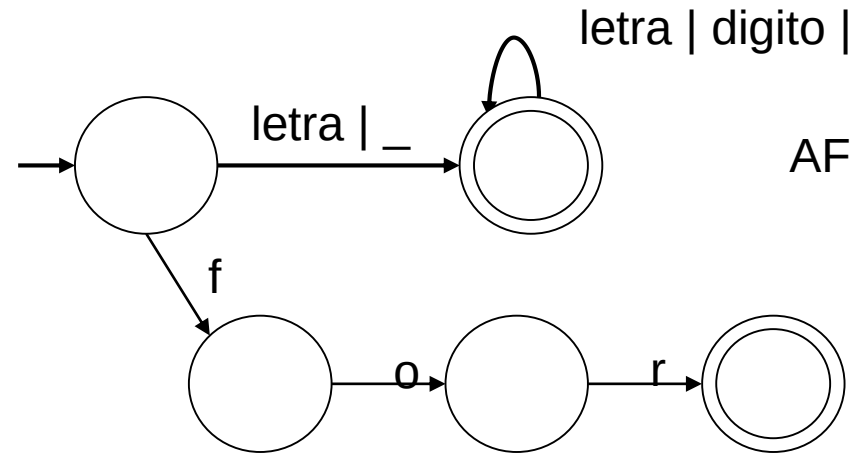
1. Um conjunto finito de estados.
2. Um conjunto finito de símbolos de entrada (alfabeto).
3. Uma função de transição que tem como argumentos um estado e um símbolo de entrada e retorna a um estado.
4. Um estado inicial.
5. Um conjunto de estados finais também chamados estados de aceitação.

Autômato Finito

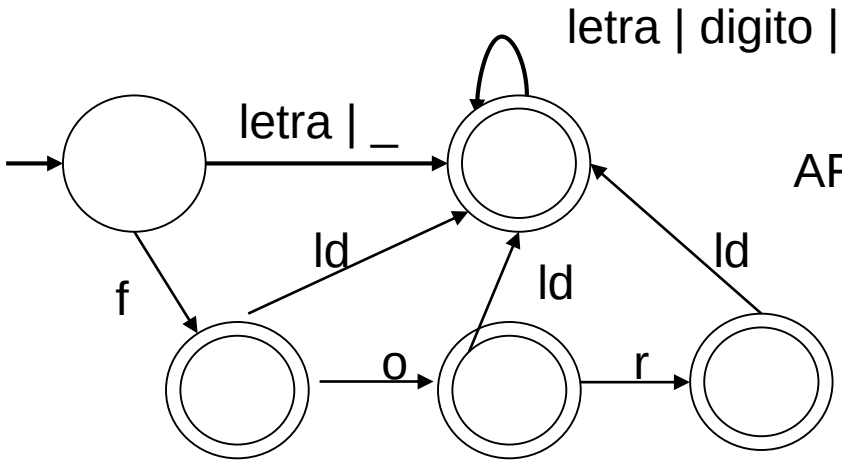




Autômato Finito



AFN – Autômato Finito Não Determinista



AFD – Autômato Finito Determinista

Onde ld representa letra | digito | _ (com exceção das letras que fazem transições para outros estados).

Autômato Finito Implementação