

Compiladores

Análise Léxica

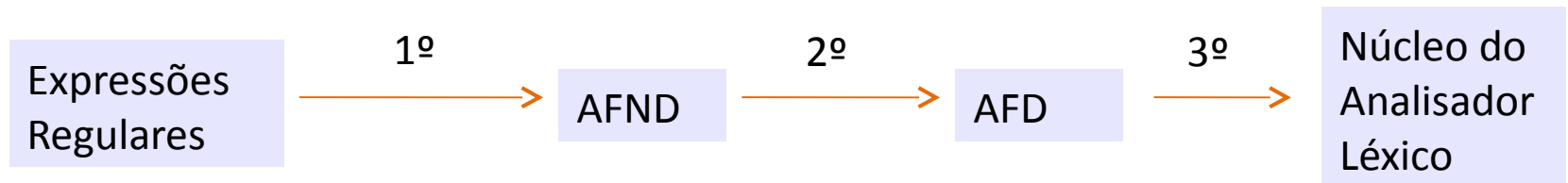
De Expressões Regulares para AFND

Prof. Dr. Luiz Eduardo G. Martins
(adaptado por Profa Dra Ana Carolina Lorena)

UNIFESP

De Expressões Regulares para AFND

- Expressões regulares são compactas, e preferíveis como descrições de *tokens*, se comparadas com AFDs
- O processo de construção do analisador léxico pode ser desenvolvido em três passos:



- Geradores de analisadores léxicos seguem esse modelo

Observações

- Ao discutir **análise léxica**, usamos três termos relacionados:

—Token

- O nome do *token* é uma descrição que representa um tipo de unidade léxica; ou
- Um par consistindo em um nome e um valor de atributo opcional

—Padrão

- É uma descrição da forma que os lexemas de um *token* podem assumir
- É comum o uso de expressões regulares para descrever o padrão

Observações

—Lexema

- Sequência de caracteres no programa fonte que casa com o padrão para um *token*
- Identificado pelo analisador léxico como uma instância de um *token*

TOKEN	DESCRIÇÃO INFORMAL	EXEMPLOS DE LEXEMAS
if	caracteres i, f	if
else	caracteres e, l, s, e	else
comparison	< or > ou <= ou >= ou == ou !=	<=, !=
id	letra seguida por letras e dígitos	pi, score, D2
number	qualquer constante numérica	3.14159, 0, 6.02e23
literal	qualquer caractere diferente de ", cercado por "s	"core dumped"

FIGURA 3.2 Exemplos de tokens.

De Expressões Regulares para AFND

- Algoritmo de Thompson

Compõe um autômato finito não determinístico pela combinação de pequenos autômatos que reconhecem os elementos primitivos de uma expressão regular:

- ▶ Um símbolo do alfabeto da linguagem
- ▶ Concatenação de duas expressões regulares
- ▶ Alternativa de duas expressões regulares
- ▶ Repetição (zero ou mais vezes) de uma expressão regular

ϵ - Expressão regular que representa a *string* vazia

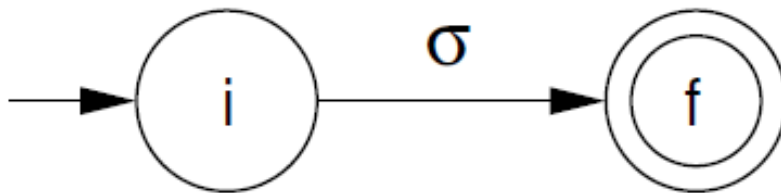
De Expressões Regulares para AFND

- Algoritmo de Thompson
 - É um método indutivo
 - O AFND é construído com base na transição pela string vazia
 - Junção das subexpressões por meio da string vazia
 - O algoritmo é baseado em passos muito simples
 - Facilita o processo de automatização do algoritmo

De Expressões Regulares para AFND

- Algoritmo de Thompson

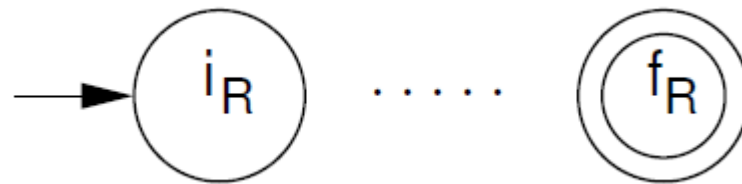
- Autômato que reconhece um símbolo do alfabeto



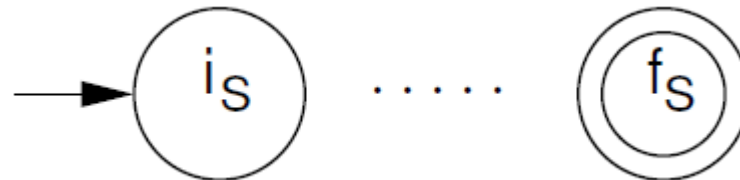
De Expressões Regulares para AFND

- Algoritmo de Thompson

- Autômato para a expressão regular R

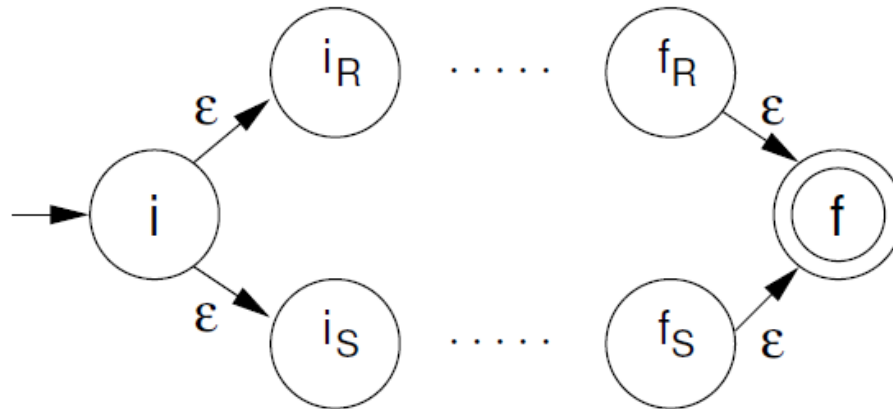


- Autômato para a expressão regular S



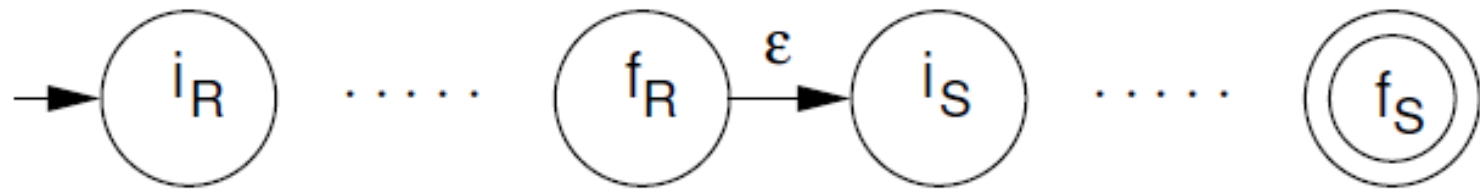
De Expressões Regulares para AFND

- Algoritmo de Thompson
 - Autômato que reconhece $R|S$



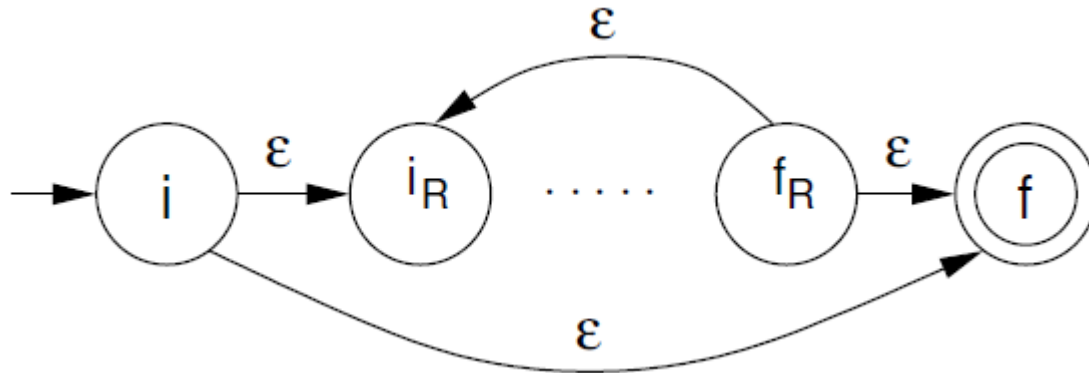
De Expressões Regulares para AFND

- Algoritmo de Thompson
 - Autômato que reconhece concatenação *RS*



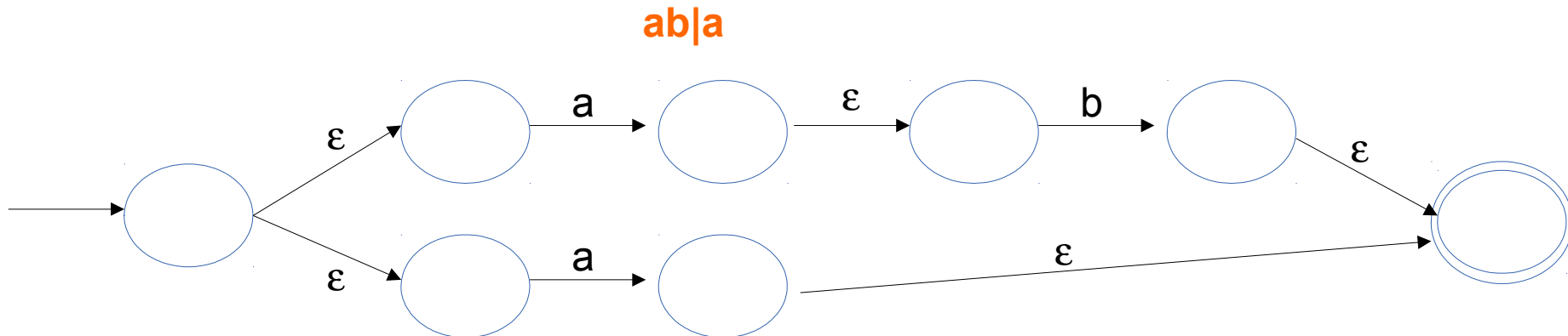
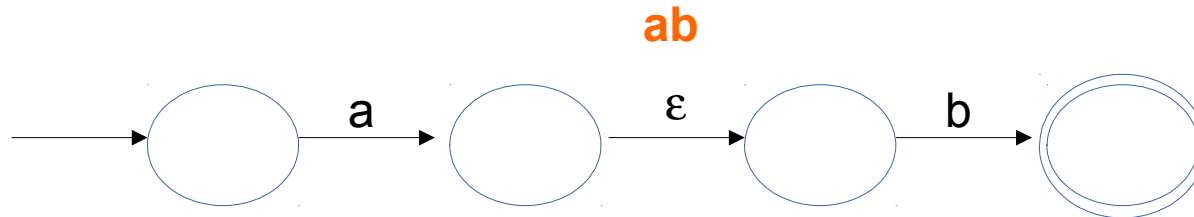
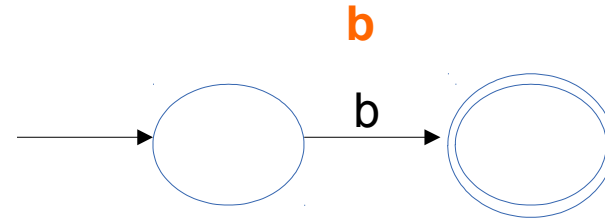
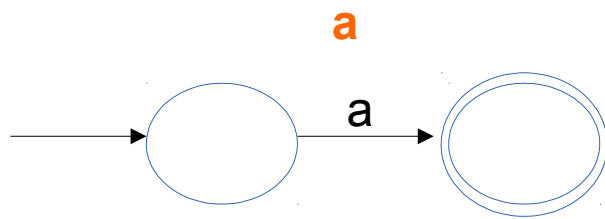
De Expressões Regulares para AFND

- Algoritmo de Thompson
 - Autômato que reconhece fecho R^*



Exemplo

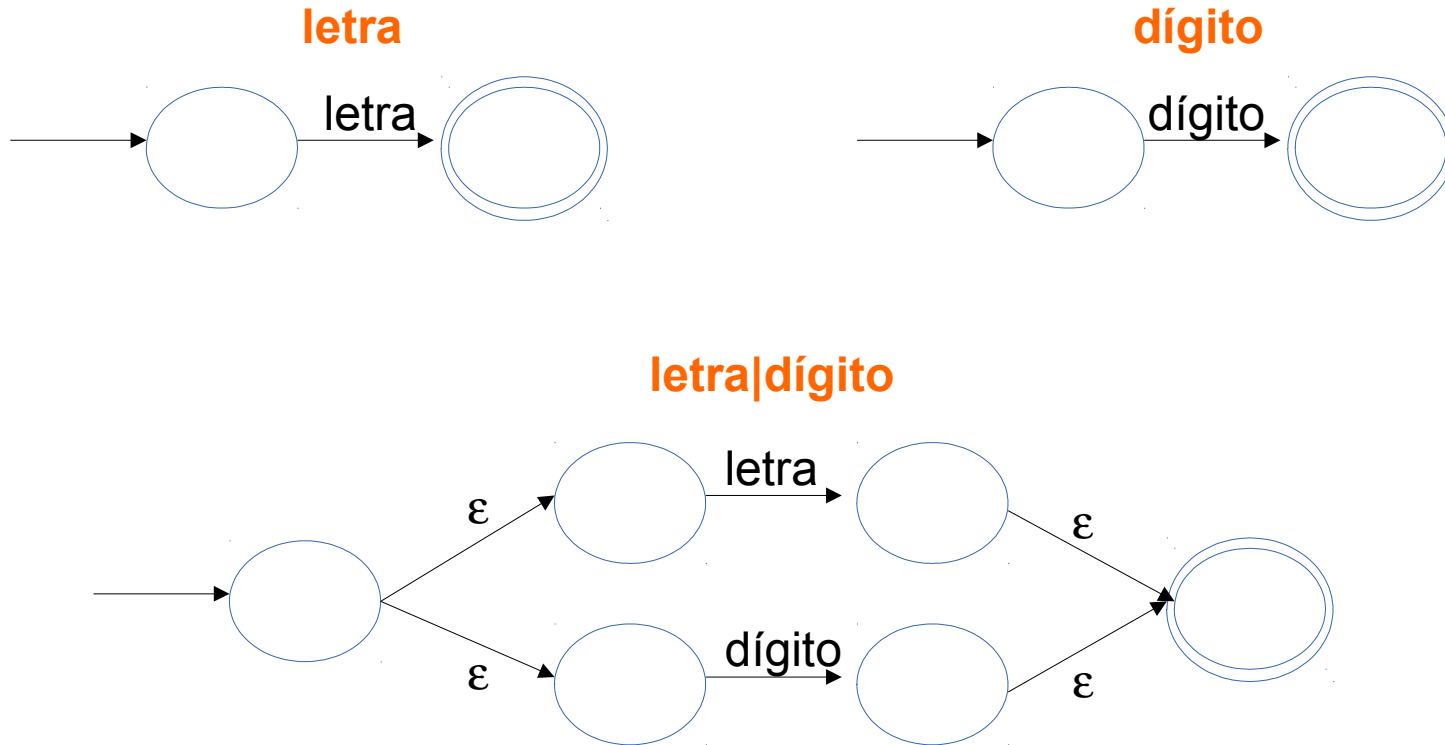
- Algoritmo de Thompson
 - AFND para a expressão regular $ab|a$



Exemplo

- Algoritmo de Thompson

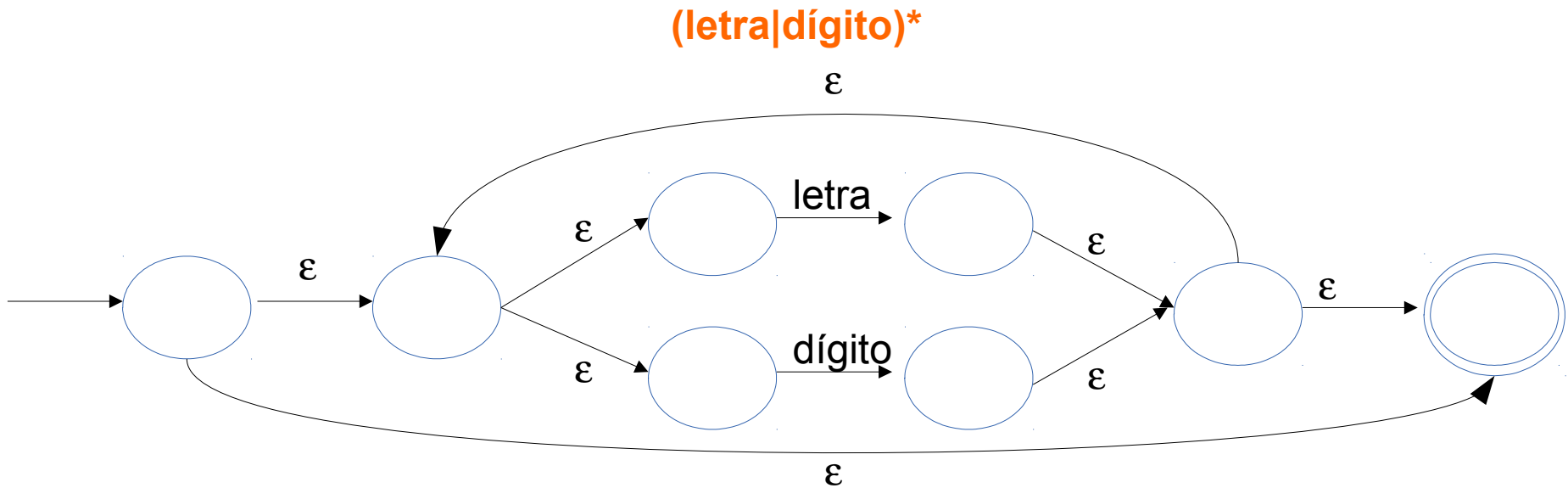
– AFND para a expressão regular $\text{letra}(\text{letra} \mid \text{dígito})^*$



Exemplo

- Algoritmo de Thompson

- AFND para a expressão regular $\text{letra}(\text{letra} \mid \text{dígito})^*$

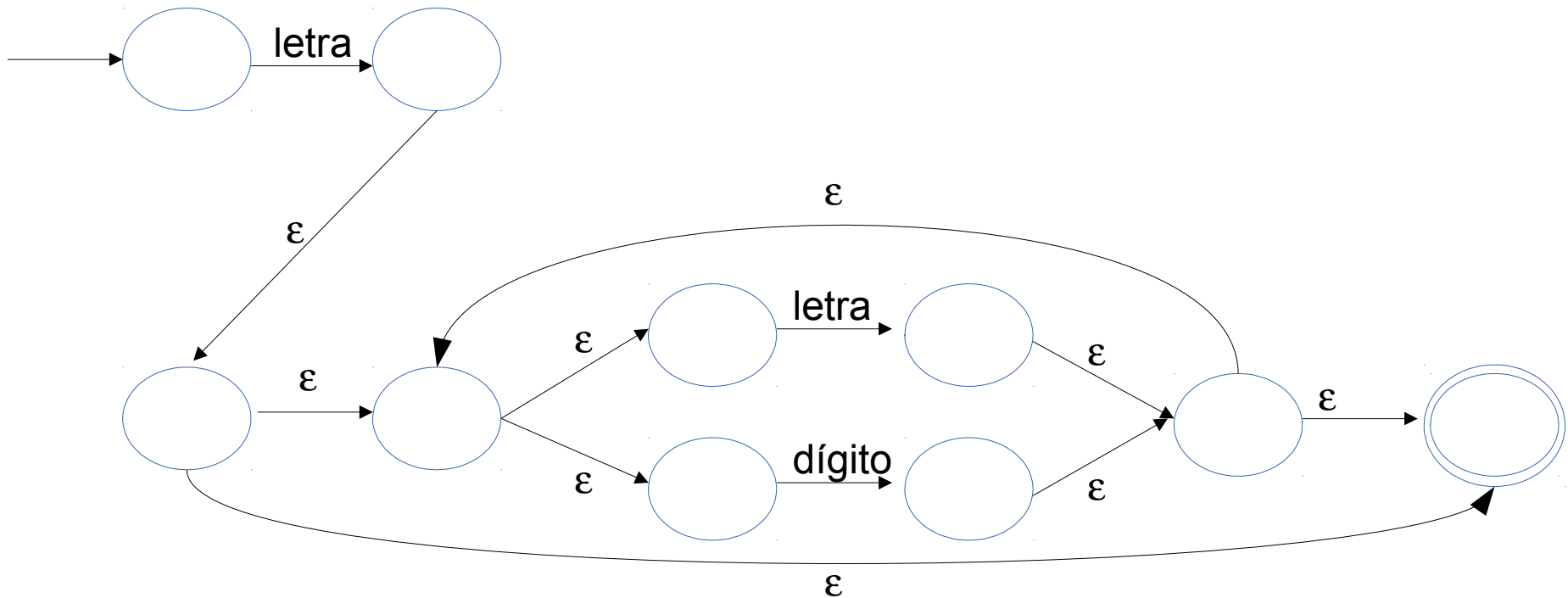


Exemplo

- Algoritmo de Thompson

– AFND para a expressão regular $\text{letra}(\text{letra} \mid \text{dígito})^*$

letra(letra|dígito)*



Exemplo

- Algoritmo de Thompson
 - AFND para a expressão regular $(0|1)^*0$

Exemplos de *strings* válidas:

00	10
010	110
0000	1100

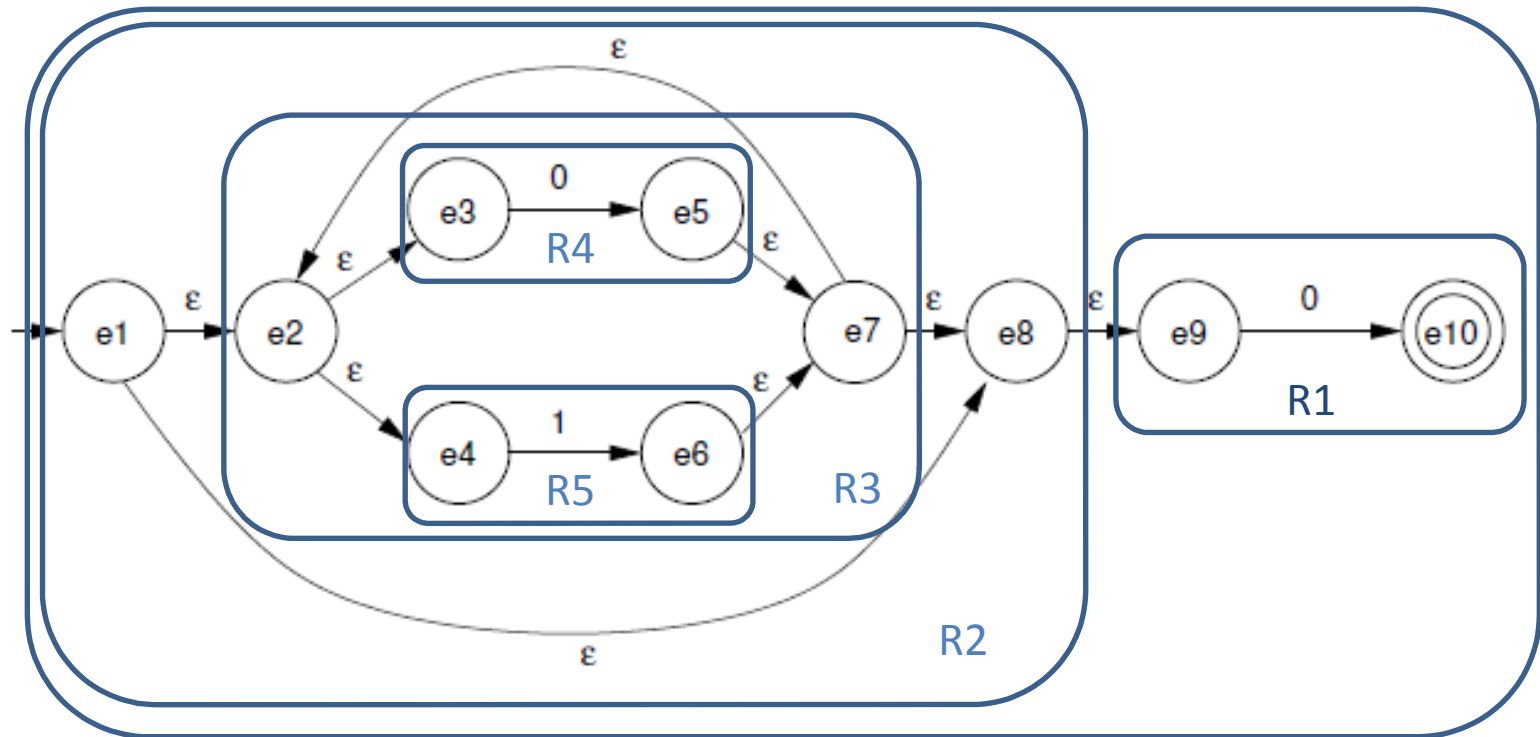
Exemplos de *strings* não válidas:

01	11
011	101
0001	1101

Exemplo

- Algoritmo de Thompson

► Autômato para $(0|1)^*0$



$$R5 = 1$$

$$R4 = 0$$

$$R3 = R4 / R5$$

$$R2 = (0|1)^* = R3^*$$

$$R1 = 0$$

$$R = R2R1$$

Exercício

- 1) Dada a expressão regular $(xx)^*(y/z)z^*$ construa o AFND para reconhecer cadeias dessa linguagem

Exercício

2) Em uma aplicação que aceita cadeias binárias, $\Sigma = \{0, 1\}$, as cadeias aceitas são aquelas que terminam com o mesmo bit que iniciaram

a) Encontre uma expressão regular que descreva essas cadeias

0

b) Desenvolva o AFND para reconhecer essas cadeias

1

Exercício

3) Converter a expressão regular $(a|b)^*a((a|b)|\varepsilon)$ para um AFND

Bibliografia

- Bibliografia consultada

Capítulo 2 LOUDEN, K. C. **Compiladores: princípios e práticas**. São Paulo: Pioneira Thompson Learning, 2004

RICARTE, I. **Introdução à Compilação**. Rio de Janeiro: Editora Campus/Elsevier, 2008.

AHO, A. V.; LAM, M. S.; SETHI, R. e ULLMAN, J. D.

Compiladores: princípios, técnicas e ferramentas. 2ª edição – São Paulo: Pearson Addison-Wesley, 2008