

## Capítulo III

# Autômatos Finitos e Conjuntos Regulares

## Geradores X Reconhecedores

Gramáticas Tipo 0	→	Máquinas de Turing
G. Sensíveis ao Contexto	→	Aut. Lim. Lineares
G. Livres de Contexto	→	Autômatos de Pilha
Gramáticas Regulares	→	Autômatos Finitos

## Autômatos Finitos

- São reconhecedores de linguagens regulares
- Tipos de Autômatos Finitos:
  - Autômato Finito Determinístico (AFD)
  - Autômato Finito Não Determinístico (AFND)

## III.1 – Autômatos Finitos Determinísticos (AFD)

**Definição formal** :  $M = (K, \Sigma, \delta, q_0, F)$ , onde:

$K \rightarrow$  É um conjunto finito não vazio de **Estados**;

$\Sigma \rightarrow$  É um **Alfabeto**, finito, de entrada;

$\delta \rightarrow$  **Função de Mapeamento** (ou de transição)  
definida em:  $K \times \Sigma \rightarrow K$

$q_0 \rightarrow \in K$ , é o **Estado Inicial**

$F \rightarrow \subseteq K$ , é o conjunto de **Estados Finais**

**Exemplo**: Seja  $M = (K, \Sigma, \delta, q_0, F)$ , onde:

$K = \{q_0, q_1\}$

$\Sigma = \{a, b\}$

$\delta = \{\delta(q_0, a)=q_0, \delta(q_0, b)=q_1, \delta(q_1, b)=q_1, \delta(q_1, a)= - \}$

$q_0 = q_0$

$F = \{q_1\}$

- Que sentenças são aceitas (reconhecidas) por M?
- Qual a Linguagem aceita (reconhecida) por M?

### Representações de AF

- **Alem da representação formal, um AF pode também ser representado por:**
  - Diagrama de Transição
  - Tabela de Transições

### Sentenças Aceitas (reconhecidas) por um A.F. M:

$$\delta(q_0, x) = p \mid p \in F$$

### Linguagem Aceita por M:

$$T(M) = \{ x \mid \delta(q_0, x) = p \wedge p \in F \}$$

## III.2 - A.F.N.D.

**Definição:**  $M = (K, \Sigma, \delta, q_0, F)$ , onde:

$K, \Sigma, q_0, F \rightarrow$  mesma definição dos A.F.D.

$\delta \rightarrow K \times \Sigma = \rho(K)$ , onde  $\rho(K) \subseteq K$

	Vantagem	Desvantagem
<b>AFD</b>	Implementação Trivial / eficiência	Representação menos natural de algumas L.R.
<b>AFND</b>	Representação mais natural de algumas LR	Implementação complexa / ineficiência

### Exemplos: Construa um AFND M |

a)  $T(M) = \{ (a, b)^*abb \}$

b)  $T(M) = \{ (0, 1)^* (00 \mid 11) (0, 1)^* \}$

c) Construa AFD  $\equiv$  AFND dos itens a) e b)

### III.3 – Transformação de AFND para AFD

**Teorema 3.1:** “Se  $\underline{L}$  é um conjunto aceito por um A.F.N.D., então  $\exists$  um A.F.D. que aceita  $\underline{L}$ ”

**Para provar o Teorema 3.1, precisamos:**

1 - Construir um AFD  $M'$  a partir de um dado AFND  $M$

2 - Mostrar que  $M' \equiv M$

**Prova:**

1 – Dado um AFND  $M = (K, \Sigma, \delta, q_0, F)$ , construir um A.F.D.  $M' = (K', \Sigma, \delta', q_0', F')$  como segue:

1 –  $K' = \{\rho(k)\}$

2 –  $q_0' = [q_0]$

3 –  $F' = \{\rho(K) \mid \rho(K) \cap F \neq \emptyset\}$

4 – Para cada  $\rho(K) \subset K'$

faça  $\delta'(\rho(K), a) = \rho'(K)$ , onde

$\rho'(K) = \{p \mid \text{para algum } q \in \rho(K), \delta(q, a) = p\};$

2 – Para mostrar que  $M' \equiv M$ ,  
basta mostrar que  $T(M') = T(M)$ .

**Exemplo:** Seja  $M$  um A.F.N.D. definido por:

$\delta$	a	b
$\rightarrow q_0$	$q_0, q_1$	$q_0$
$q_1$	---	$q_2$
$q_2$	---	$q_3$
$*q_3$	---	---

### III.4 - Relação entre GR e AF

**Teorema 3.2:** “Se  $G = (V_n, V_T, P, S)$  é uma G.R., então  $\exists$  um A.F.  $M = (K, \Sigma, \delta, q_0, F) \mid T(M) = L(G)$ ”.

**Prova:** a – Mostrar que  $M$  existe  
b – Mostrar que  $T(M) = L(G)$

a) Defina  $M$ , como segue:

1 –  $K = V_n \cup \{A\}$ , onde  $A$  é um símbolo novo

2 –  $\Sigma = V_T$

3 –  $q_0 = S$

4 –  $F = \{A, S\}$  se  $S \rightarrow \varepsilon \in P$   
 $\{A\}$  se  $S \rightarrow \varepsilon \notin P$

5 – Construa  $\delta$  de acordo com as regras a, b e c.

a) Se  $B \rightarrow a \in P \Rightarrow \delta(B, a) = A$

b) Se  $B \rightarrow a C \in P \Rightarrow \delta(B, a) = C$

c) Para todo  $a \in V_T, \delta(A, a) = -$  (indefinido)

b) Para mostrar que  $T(M)=L(G)$ , deve-se mostrar:

1

–  $L(G) \subseteq T(M)$

2 –  $T(M) \subseteq L(G)$

**Exemplos:**

1) $S \rightarrow a S \mid b B$ $B \rightarrow b B \mid c$	2) $S \rightarrow b A \mid a B \mid b \mid \varepsilon$ $A \rightarrow b A \mid a B \mid b$ $B \rightarrow b B \mid a C$ $C \rightarrow b C \mid a A \mid a$
---	---

**Teorema 3.3:** “Se  $M = (K, \Sigma, \delta, q_0, F)$  é um A. F., então  $\exists$  uma G.R.  $G = (V_n, V_t, P, S) \mid L(G) = T(M)$ ”

**Prova:** a – Mostrar que G existe

b – Mostrar que  $L(G) = T(M)$

a) Seja  $M = (K, \Sigma, \delta, q_0, F)$  um A.F.D..

Construa uma G.R.  $G=(V_n, V_T, P, S)$ , como segue:

1 –  $V_n = K$

2 –  $V_t = \Sigma$

3 –  $S = q_0$

4 – Defina P, como segue:

a) Se  $\delta(B, a) = C$  então adicione  $\underline{B \rightarrow aC}$  em P

b) Se  $\delta(B, a) = C \wedge C \in F$ , adicione  $\underline{B \rightarrow a}$  em P

c) Se  $q_0 \in F$ ,  
então  $\varepsilon \in T(M)$ .

Neste caso,  $L(G) = T(M) - \{\varepsilon\}$ , portanto,

construa uma GR  $G_1 \mid L(G_1) = L(G) \cup \{\varepsilon\}$

Senão  $\varepsilon \notin T(M)$  e  $L(G) = T(M)$

b) Para mostrar que  $L(G) = T(M)$ , devemos mostrar que:

1 –  $T(M) \subseteq L(G)$

2 –  $L(G) \subseteq T(M)$

**Exemplos :**

$\delta$	a	b
$* \rightarrow S$	A	B
A	S	C
B	C	S
C	B	A

$\delta$	a	b	b
$\rightarrow S$	S	B	-
B	-	B	A
$*A$	-	-	-

## III.5 - Minimização de Autômatos Finitos

**Definição:** Um AFD  $M = (K, \Sigma, \delta, q_0, F)$  é mínimo se:

- 1 – Não possui estados inacessíveis (inalcançáveis);
- 2 – Não possui estados mortos;
- 3 – Não possui estados equivalentes.

**Alg. para Construção das Classes de Equivalência**

- 1 – Crie, um estado  $\phi$  para representar as indefinições;
- 2 – Divida  $K$  em duas CE :  $F$  e  $K-F$ ;
- 3 – Aplique a **lei de formação de CE**, até que nenhuma nova CE seja formada

**Algoritmo para construção do A.F. Mínimo**

**Entrada:** Um A.F.D.  $M = (K, \Sigma, \delta, q_0, F)$ ;

**Saída:** Um AFD Mínimo  $M' = (K', \Sigma, \delta', q_0', F') \mid M' \equiv M$ ;

**Método:**

- 1 – Elimine os estados Inacessíveis;
- 2 – Elimine os estados Mortos;
- 3 – Construa todas as CE de  $M$ .
- 4 – Construa  $M'$ , como segue:
  - a)  $K' = \{ CE \}$
  - b)  $q_0' = CE$  que contiver  $q_0$ ;
  - c)  $F' = \{ [q] \mid \exists p \in F \text{ em } [q] \}$
  - d)  $\delta'([p], a) = [q] \Leftrightarrow \delta(p_1, a) = q_1 \text{ está em } M \wedge p_1 \in [p] \wedge q_1 \in [q]$

Exemplo: Minimize o seguinte A.F.D.

$\delta$	a	b
* $\rightarrow A$	G	B
B	F	E
C	C	G
* D	A	H
E	E	A
F	B	C
* G	G	F
H	H	D

**Exercícios:**

1)

$\delta$	a	b	c
* $\rightarrow S$	A	B,F	S,F
A	S,F	C	A
B	A	-	B,S,F
C	S,F	-	A,C
*F	-	-	-

2)

$\delta$	0	1
$\rightarrow S$	A , D	E
A	A , B	C , E
B	B	-
C	A , B	-
D	B , D	C
*E	E	E

3)

$\delta$	a	b
$\rightarrow q0$	q1	q2
q1	q3	-
q2	-	q4
*q3	q3	q3
*q4	q4	q4

4)

$\delta$	a	b	c
* $\rightarrow S$	A , C	A , D	B , C
*A	A	A	B
*B	A	A	-
*C	C	D	C
*D	C	-	C



## III.6 – Conjuntos e Expressões Regulares

### Conjuntos Regulares (C.R.)

#### 1 – (\* definição matemática (primitiva) \*)

Seja  $\Sigma$  um alfabeto qualquer.

Definimos um C.R. sobre  $\Sigma$ , como segue:

a –  $\phi$  é um C.R. sobre  $\Sigma$ ;

b –  $\{\epsilon\}$  é um C.R. sobre  $\Sigma$ ;

c –  $\{a\}$ , para todo  $a \in \Sigma$ , é um C.R. sobre  $\Sigma$ ;

d – Se P e Q são C.R. sobre  $\Sigma$ , então:

1 –  $P \cup Q$  (união),

2 –  $P.Q$  (ou  $PQ$ ) (concatenação),

3 –  $P^*$  (fechamento).

Também são C.R. sobre  $\Sigma$ ;

e – Nada mais é C.R.

#### 2 – Linguagens geradas por Gramáticas Regulares.

#### 3 – Linguagens reconhecidas por Autômatos Finitos.

#### 4 – Linguagens denotados por Expressões Regulares.

## Expressões Regulares (E.R.)

### Definição:

- 1 –  $\phi$  é uma E.R. e denota o C.R.  $\phi$
- 2 –  $\epsilon$  é uma E.R. e denota o C.R.  $\{\epsilon\}$
- 3 –  $\underline{a} \in \Sigma$ , é uma E.R. e denota o C.R.  $\{\underline{a}\}$
- 4 – Se  $\underline{p}$  e  $\underline{q}$  são E.R. denotando  $P$  e  $Q$ , então:
  - a –  $(p \mid q)$  é uma E.R. denotando  $P \cup Q$
  - b –  $(p.q)$  ou  $(pq)$  é uma E.R. denotando  $PQ$
  - c –  $(p)^*$  é uma E.R. denotando  $P^*$
- 5 – Nada mais é E.R..

### Observações:

1 – ordem de precedência: 1)  $*$  2)  $.$  3)  $\mid$

2 – abreviaturas usuais:

$$\begin{aligned} p^+ &= pp^* \\ p^? &= p \mid \epsilon \\ p^{\text{f}}q &= p(qp)^* \end{aligned}$$

### Relação entre E.R. e C.R.

- 1 – Para todo C.R.  $\exists$  uma E.R. que o denota
- 2 – Para toda E.R. é possível construir seu C.R.
- 3 –  $E1 \equiv E2 \Leftrightarrow$  elas denotam o mesmo C.R..

### III.6.1 – Autômatos Finitos com $\varepsilon$ -transições

AFND- $\varepsilon$  :  $M = (K, \Sigma, \delta, q_0, F)$ , onde:

$K, \Sigma, q_0, F \rightarrow$  mesma definição dos A.F.D.

$\delta \rightarrow K \times \Sigma \cup \{ \varepsilon \} = \rho(K)$ , onde  $\rho(K) \subseteq K$

#### Observações:

- $\varepsilon$ -transições permitem movimentos independentes da entrada;
- O uso de  $\varepsilon$ -transições não incrementa a expressividade dos AF;
- Todo AFND- $\varepsilon$  possui um AFND equivalente;

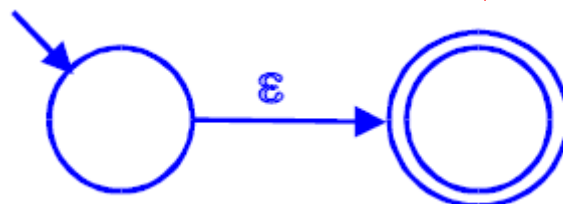
### III.6.2 – Correspondência entre ER e AF

Para mostrarmos que toda ER possui um AF correspondente, é suficiente mostrarmos que toda ER básica ( $\Phi$ ,  $\varepsilon$ ,  $a$ ,  $(\alpha \mid \beta)$ ,  $(\alpha \cdot \beta)$  e  $\alpha^*$  - onde  $\alpha, \beta$  são ERs quaisquer) possui um AF correspondente:

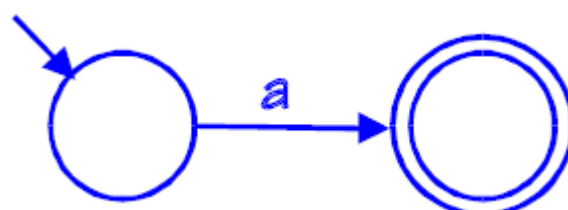
1 – AF representando a ER “ $\phi$ ” ( $M|T(M) = \phi$ )



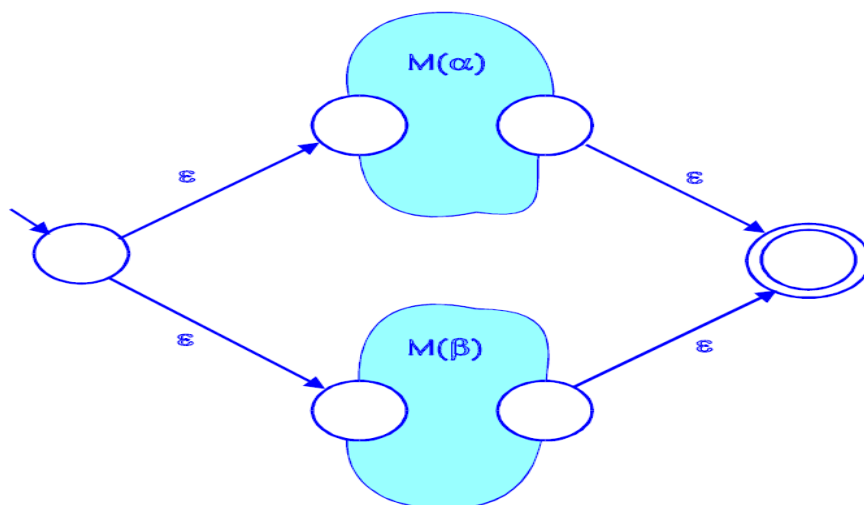
2 – AF representando a ER “ $\varepsilon$ ” ( $M|T(M) = \{\varepsilon\}$ )



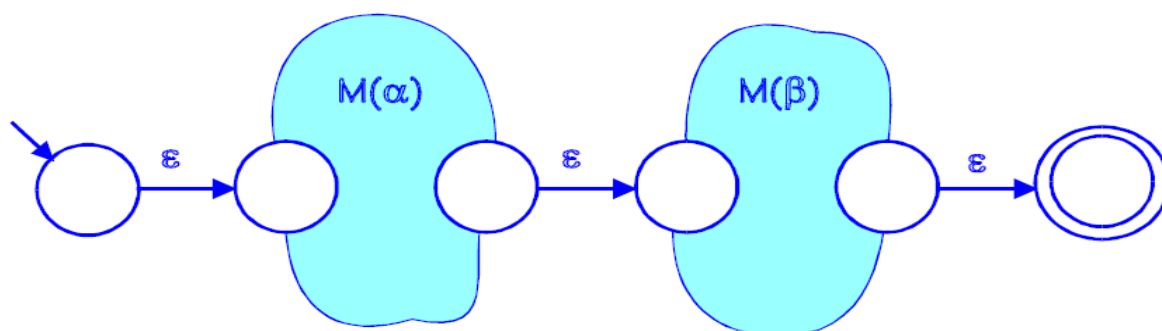
3 – AF representando a ER “ $a$ ” ( $M|T(M) = \{a\}$ )



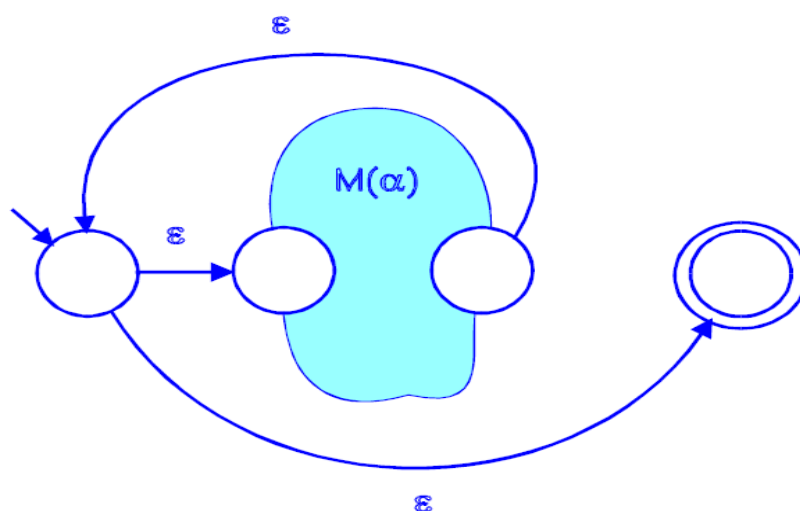
**4 – AF representando a ER “ $\alpha \mid \beta$ ” ( $M|T(M) = \{\alpha|\beta\}$ )**



**5 – AF representando a ER “ $\alpha.\beta$ ” ( $M|T(M) = \{\alpha.\beta\}$ )**



**5 – AF representando a ER “ $\alpha^*$ ” ( $M|T(M) = \{\alpha^*\}$ )**



**OBS. Figuras extraídas de J.L.M.Rangel Neto (COPPE/UFRJ-PUC/RJ)**

### III.6.3 - Transformação de ER para AF

#### Diferentes métodos (estratégias):

- Método de Thompson
- Método de De Simoni (Adap. de Rennes/AHO)

#### III.6.3.1 - Método de Thompson

- **Consiste em:**

- 1 - Decompor uma ER em suas sub-expressões constituintes;
- 2 - Construir o AFND $\epsilon$  de cada subexpressão;
- 3 - Compor o AFND $\epsilon$  final (usando  $\epsilon$ -transições)

- **Exemplo:**

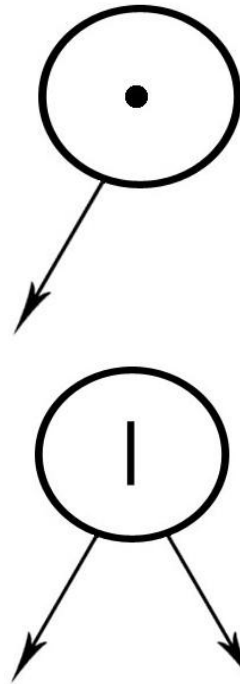
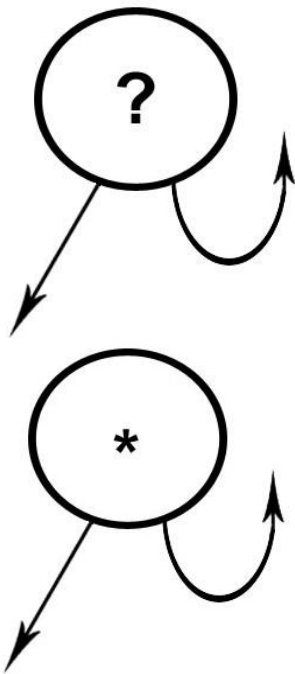
#### III.6.3.2 - Método de De Simone

- **Consiste em:**

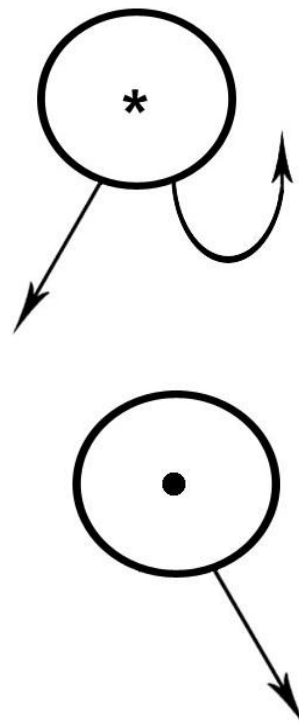
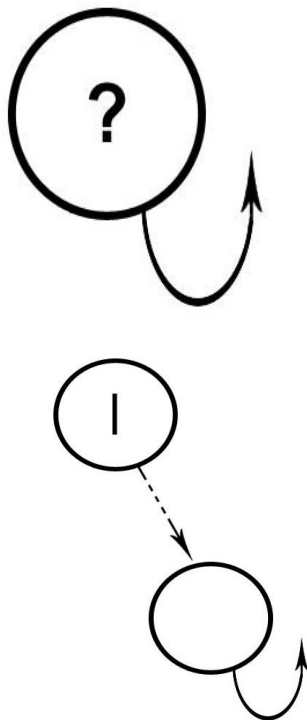
- 1 - Construir uma árvore binária costurada correspondente a ER, onde os nodos internos representam os operadores e os nodos folhas representam os operandos;
- 2 - Numerar os nodos folha de 1 a n;
- 3 - Compor o estado inicial do AF com o número das folhas alcançadas no percurso da Árvore, de acordo com as rotinas “Descer” e “Subir”, a partir do nodo raiz;
  - 3.1 – Caso o percurso atinja o final da Árvore, inclua “ $\lambda$ ” na composição do estado;
- 4 - Definir as transições do estado inicial, com os símbolos dos nodos folha “i” que compõe esse estado, como sendo um estado composto pelos nodos alcançados a partir do caminhar na árvore a partir da costura de cada nodo “i”, usando as rotinas “Descer” e “Subir”;
  - 4.1 – Caso o percurso atinja o final da Árvore, inclua “ $\lambda$ ” na composição do estado;
- 5 – Repetir o passo 4 para todos os estados novos criados;
- 6 – Definir como finais, os estados que contenham “ $\lambda$ ”

- **Exemplos:**

## Rotinas Descer:



## Rotinas Subir:



## III.7 – Lema do Bombeamento para Linguagens Regulares - "Pumping Lemma"

**Objetivo:** demonstrar que algumas linguagens não são regulares.

**Lema do Bombeamento :** Se  $L$  é uma LR, então existe uma constante  $n \geq 1$  | para todo  $w \in L$ ,  $|w| \geq n$ , podemos escrever  $w$  como  $x y z$  onde:

- $|xy| \leq n$
- $y \neq \varepsilon$
- $xy^i z \in L$  para qualquer  $i \geq 0$

**Idéia geral:** O "Lema do Bombeamento", ou "Pumping Lemma", nos diz que qualquer sentença  $w$  de uma **linguagem regular** pode ser decomposta em três partes:  $w = xyz$ , de maneira que a repetição (o *bombeamento*) de  $y$ , qualquer número de vezes, resulta em sentenças  $xy^i z$  que também pertencem à linguagem; ou seja, as sequências  $xz$ ,  $xyz$ ,  $xyyz$ , ... , também serão sentenças da linguagem em questão.

Para mostrar que uma linguagem **não é regular**, basta encontrar uma sentença  $w$  qualquer pertencente à linguagem, que não satisfaça o lema do bombeamento – isto é, não possa ser decomposta em  $xyz$  de forma que seja possível *bombear*  $y$  e continuar na linguagem.

**Exemplos:**

## III.8 – Propriedades e Prob. de Decisão de CR

### Propriedades Básicas de C.R.

1 – União

2 – Concatenação

3 - Fechamento

4 – Complemento: Se  $L_1 \subseteq \Sigma^*$  é CR  $\Rightarrow \Sigma^* - L_1$  também é CR

5 – Intersecção: Se  $L_1$  e  $L_2$  são CR  $\Rightarrow L_1 \cap L_2$  também é CR

### Problemas de Decisão sobre C.R.

1 – Membership :  $x \in T(M)$ ?

2 – Emptiness :  $T(M) = \varnothing$ ?

3 – Finiteness :  $T(M)$  é finita?

4 – Containment :  $T(M_1) \subseteq T(M_2)$ ?

5 – Equivalencia :  $T(M_1) = T(M_2)$ ?

6 – Intersecção Vazia :  $T(M_1) \cap T(M_2) = \varnothing$ ?

## III.9 – Implementação de Autômatos Finitos

### Formas básicas para implementação de A.F.:

- Implementação Específica
- Implementação Geral (ou genérica);

## III.10 – AF com saída

A funcionalidade dos AF pode ser estendida (sem alterar a classe de linguagens reconhecida), atribuindo-se ações (significados):

- Às Transições (Máquinas de Mealy);
- Aos estados (Máquinas de Moore)

## III.11 – Aplicações de A.F. e E.R.

1 – Compiladores – Análise Léxica

2 – Editores de texto – busca/substituição

3 – Reconhecimento de padrões

4 – Outras: S.O, Redes, Hipertexto, Robótica, Criptografia, ...