MÁQUINAS, AUTÔMATOS E LINGUAGENS DE MEMÓRIA FINITA: ALGUMAS PROPRIEDADES

Ana Cristina Costa Drummond

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS (M.Sc.)

Aprovada por:

Prof. Paulo Augusto Silva Veloso
(Presidente)

Prof. Nelson Maculan Filho

Prof. Roberto Lins de Carvalho

RIO DE JANEIRO, RJ - BRASIL
JULHO DE 1979

DRUMMOND, ANA CRISTINA COSTA

Máquinas, Autômatos e Linguagens de Memória Finita: Algumas Propriedades |Rio de Janeiro| 1979.

29,7 cm (COPPE-UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1979).

Tese - Univ. Fed. Rio de Janeiro, Fac. Engenharia.

I. Máquinas, Autômatos e Linguagens de de Memória Finita: Algumas Propriedades.
I. COPPE/UFRJ II. Título (série).

DEDICATÓRIA

Para

Armando e André

AGRADECIMENTOS

Ao terminar este trabalho, agradeço a todos aqueles que de algum modo me ajudaram, especialmente,

- Paulo Augusto Silva Veloso pela orientação, pelas sugestões valiosas e pela amizade que demonstrou no decorrer do trabalho.

- Nelson Maculan Filho e Roberto Lins de Carvalho por participarem da banca.

- Angela Maria Schwartz pelo trabalho de datilografia.

- Ao CNPq pela ajuda financeira.

- E a Armando, meus pais e irmãos que me incentivaram a chegar ao final deste trabalho.

RESUMO

O presente trabalho estuda algumas propriedades das linguagens reconhecidas pelo autômatos de memória finita, que são máquinas de memória finita usadas como reconhecedoras. A importância desta classe se deve à sua relação com problemas de codificação/decodificação.

Após caracterizar as linguagens de memória finita por seus autômatos, mostra-se que esta classe de linguagens é fechada sob as operações de complemento e derivada, para as demais operações apresentando casos de fechamento.

A B S T R A C T

This work examines some properties of the languages recognized by the finite - recall automata. Such automata amount to finite-recall machines used as recognizers. The importance of this class stems from its connections with problems in information coding/decoding.

First the finite-recall languages are characterized by means of their automata, then this class of languages is shown to be closed under complementation and derivatives. For other operations some special cases yielding closure are presented.

ÍNDICE

CAPÍTULO I - INTRODUÇÃO 1

CAPÍTULO II - CONCEITOS GERAIS 3

II.1. MÁQUINAS SEQUENCIAIS 3

II.2. AUTÔMATO FINITO 8

II.3. LINGUAGEM REGULAR 12

CAPÍTULO III - MÁQUINAS DE MEMÓRIA FINITA 14

III.1. MMF 14

III.2. MMF (M REDUZIDA) 15

III 3. TABELA DE TESTE 15

III.4. GRAFO DE TESTE 16

III.5. MMF (DEFINIÇÃO KOHAVI[1]) 17

III.6. CIRCUITO DE ENTRADA/SAÍDA 17

III.7. CIRCUITO ENTRADA 17

III.8. MMF (CIRCUITOS E/S) 18

III.9. EXEMPLOS DE MMF's E NÃO MMF's 20

III.10. MÁQUINA DEFINIDA 21

CAPÍTULO IV - AUTÔMATO DE MEMÓRIA FINITA 23

IV.1. AMF 23

IV.2. EXEMPLOS AMF's 24

IV.3. EXEMPLOS DE NÃO AMF's 25

IV.4. AUTÔMATO DEFINIDO 25

IV.5. ESTADOS COMPATÍVEIS 26

CAPÍTULO V - LINGUAGEM DE MEMÓRIA FINITA 29

V.1. LMF 29

V.1.2. EXEMPLOS DE LMF's 30

V.1.3. EXEMPLOS DE NÃO LMF's 31

V.2. LINGUAGEM DEFINIDA 31

V.3. LINGUAGEM $\bigcup_{i=1}^{n} A_i B_i^*$ (A,B finitos e B_i distin - tos entre si) 33

V.4. L Σ* PARA L ε LMF 33

V.5. L U {w} PARA L ε LMF e w ε Σ* 34

V.5.1. L U P 36

V.6. L - {w} PARA L ε LMF E w ε Σ* 37

V.6.1. L - P 37

V.6.2. (P - Q) U (Q - P) 38

V.7. Σ* B Σ*, A Σ* (A e B finitos) 38

V.8. (A U B Σ*) DEFINIDAS REVERSAS 38

CAPÍTULO VI - PROPRIEDADES DAS LINGUAGENS DE MEMÓRIA FINITA.

VI.1. COMPLEMENTO 40

VI.2. UNIÃO E INTERSEÇÃO 41

VI.3. CONCATENAÇÃO 44

VI.4. TRANSPOSTA 45

VI.5. DERIVADA 47

VI.6. ESTRELA 48

VI.7. QUOCIENTES 50

VI.8. IMAGEM HOMOMORFA 53

CAPÍTULO VII - CARACTERÍSTICAS DAS LMF's 60

CAPÍTULO VIII - CONCLUSÃO 61

BIBLIOGRAFIA 63

C A P Í T U L O I

I. INTRODUÇÃO

O objetivo deste trabalho é estudar algumas propriedades das linguagens aceitas por uma classe especial de máquina, as máquinas de memória finita.

A importância desta classe de máquina se deve a sua relação com problemas de codificação/decodificação , isto devido a sua propriedade de ser sua saída final a uma sequência de entrada suficientemente longa independente do estado inicial. Esta saída é determinada pelo último símbolo de entrada e a sequência de entrada/saída precedente.

O estudo é desenvolvido por capítulos da seguinte maneira:

II - apresentamos alguns conceitos básicos e terminologia das máquinas sequenciais, dos autômatos finitos e das linguagens regulares.

III - definimos máquinas de memória finita e máquinas definidas que são caracterizadas com base nos conceitos de circuitos de entrada/saída e de entrada.

IV - introduzimos o conceito de estados compatíveis e a partir deste conceito caracterizamos os autômatos de memória finita, definidos através de MMF's.

V - definimos as linguagens de memória finita como as reconhecidas por um autômato de memória finita. Caracterizamos alguns tipos de linguagens de memória finita.

VI - apresentamos algumas operações sobre as linguagens de memória finita e verificamos que a classe só é fechada para complemento e derivada.

VII - uma primeira caracterização intrínseca das LMF's.

Finalmente, o capítulo de conclusões apresenta um resumo dos principais resultados, bem como, sugestões de problemas para futuras pesquisas.

CAPÍTULO II

II. CONCEITOS GERAIS

Neste capítulo, apresentaremos alguns conceitos básicos e terminologia das máquinas sequenciais, dos autômatos finitos e das linguagens regulares, que servirão de base ao desenvolvimento dos capítulos seguintes. Daremos também alguns exemplos.

II.1. MÁQUINAS

II.1.1. DEFINIÇÃO

Uma máquina sequencial é uma quíntupla $M = \langle\Sigma, S, \Delta, f, g\rangle$ onde:

i) Σ é um conjunto não vazio (de símbolos de entrada);

ii) S é um conjunto não vazio (de estados);

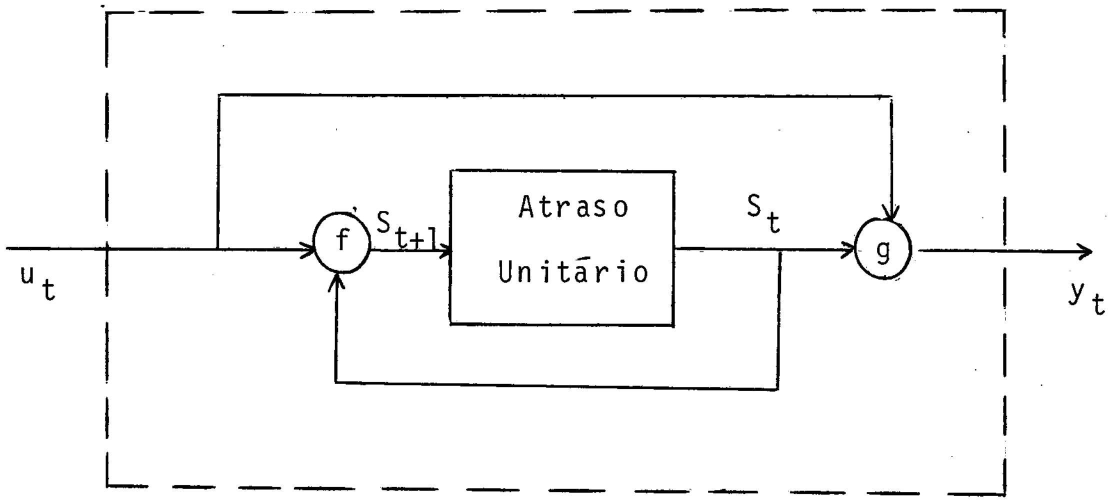
iii) Δ é um conjunto não vazio (de símbolos de saída);

iv) $f: S \times \Sigma \to S$ (função de transição);

v) $g: S \times \Sigma \to \Delta$ (função de saída).

Dizemos que $M = \langle\Sigma, S, \Delta, f, g)$ é uma máquina sequencial finita quando S, Σ e Δ são finitos.

II.1.2. INTERPRETAÇÃO |Booth[3]|



Sendo o estado atual $S_t \varepsilon S$ e recebendo no instante t, o símbolo de entrada $u_t \varepsilon \Sigma$, M da saída $y_t = g(S_t, u_t)$ e no próximo instante (t + 1), estará no estado S_{t+1} = $= f(S_t, u_t)$, que é armazenado no atraso.

Nota: Este modelo é chamado de máquina sequencial de Mealy.

II.1.3. DEFINIÇÃO

Seja M = <Σ, S, Δ, f, g> uma máquina sequencial, podemos definir uma extensão de f, f: S x Σ^* $\rightarrow$ S da se guinte forma; onde $s \in S$:

$f(s, \lambda) = s$ (λ é sequência de comprimento zero).

$f(s, xa) = f(f(s,x),a)$; para $x \in \Sigma^*$, $a \in \Sigma$.

II.1.4. DEFINIÇÃO

Seja $M = \langle\Sigma, S, \Delta, f, g\rangle$ uma máquina sequencial, podemos definir uma extensão de g, $g: S \times \Sigma^* \to \Delta^*$ e também $\hat{g}: S \times \Sigma^* \to \Delta$ como se segue

$g(s, \lambda) = \lambda$

$g(s, xa) = g(s, x)\, g(f(s, x), a)$

$\hat{g}(s, xa) = g(f(s, x), a)$

para todo $s \in S$, $x \in \Sigma^*$, $a \in \Sigma$

$g(s, x)$ dá a sequência de saída e

$\hat{g}(s, x)$ dá o último símbolo de saída |Hopcroft, Ullman[4]|.

II.1.5. EXEMPLOS

Mostraremos algumas maneiras para especificar uma máquina sequencial de Mealy.

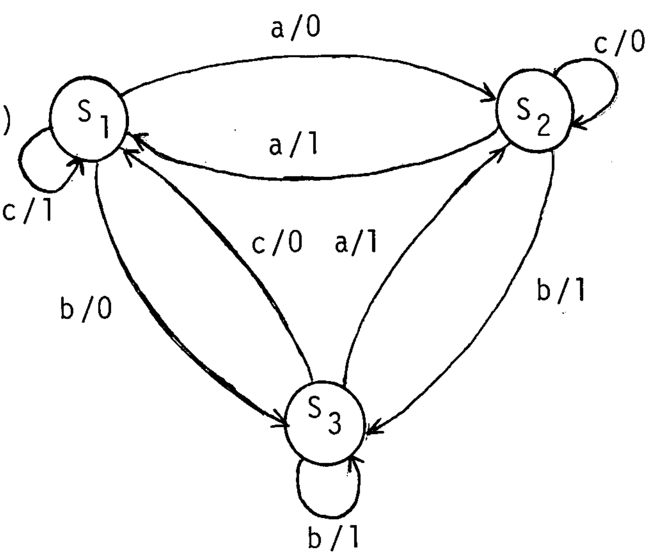
i) $M = <\{a, b, c\}, \{S_1, S_2, S_3\},$

$\{0, 1\}, f, g>$

	a	b	c
S_1	$S_{2/0}$	$S_{3/0}$	$S_{1/1}$
S_2	$S_{1/1}$	$S_{3/1}$	$S_{3/0}$
S_3	$S_{2/1}$	$S_{1/1}$	$S_{1/0}$

(Tabela de Transição)

ii)



(Diagrama de Transição)

iii) $M = <\{a, b\}, \{A, B, C\}, \{0, 1\}, f, g>$

$f(A, a) = B$

$f(A, b) = C$

$f(B, a) = C$

$f(B, b) = B$

$f(C, a) = C$ (função de transição)

$f(C, b) = C$

$g(A, a) = 0$

$g(A, b) = 1$

$g(B, a) = 1$ (função de saída)

$g(B, b) = 0$

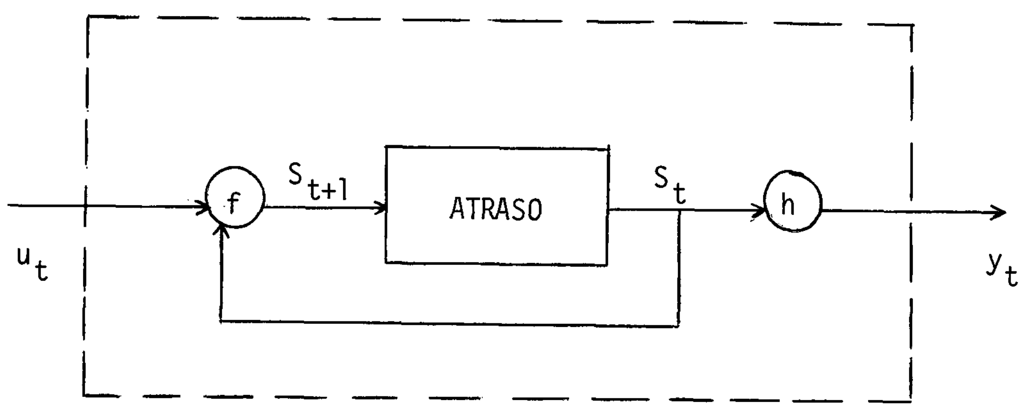
$g(C, a) = 1$

$g(C, b) = 0$

II.1.6. DEFINIÇÃO

Máquina sequencial de Moore é uma quíntupla ordenada $M = <\Sigma, S, \Delta, f, h>$, onde Σ, S, Δ e f tem a mesma definição da máquina de Mealy, e $h: S \to \Delta$ (função de saída)(Booth[3]).

II.1.7. INTERPRETAÇÃO (Booth[3])



II.1.8. EXEMPLOS

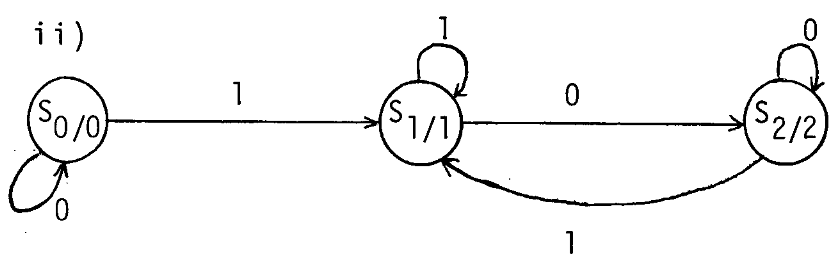
i) $M = <\{a, b, c\}, \{S_1, S_2, S_3\}, \{1, 0\},$

$f, h>$

	a	b	c	
S_1	S_3	S_1	S_2	1
S_2	S_2	S_1	S_2	0
S_3	S_2	S_1	S_3	0

(Tabela de Transição)

ii)



(Diagrama de Transição)

Nota: Uma máquina sequencial de Mealy tem a saída associada a cada transição, isto é, depende do estado e do símbolo de entrada. Já uma máquina sequencial de Moore a saída está associada a cada estado.

II.2.1. DEFINIÇÃO

Um autômato finito (abreviando AF) é uma quin tupla $A = < \Sigma, S, f, s_0, F >$, onde:

$S \neq \phi$ é finito (conjunto de estados)

$\Sigma \neq \phi$ é finito (conjunto de entrada)

$f : S \times \Sigma \rightarrow S$ (função de transição)

$s_0 \in S$ (estado inicial)

$F \quad S$ (conjunto de estados finais)

A função de transição pode ser extendida a $f : S \times \Sigma^* \rightarrow S$ como em II.1.3.

II.2.2. DEFINIÇÃO

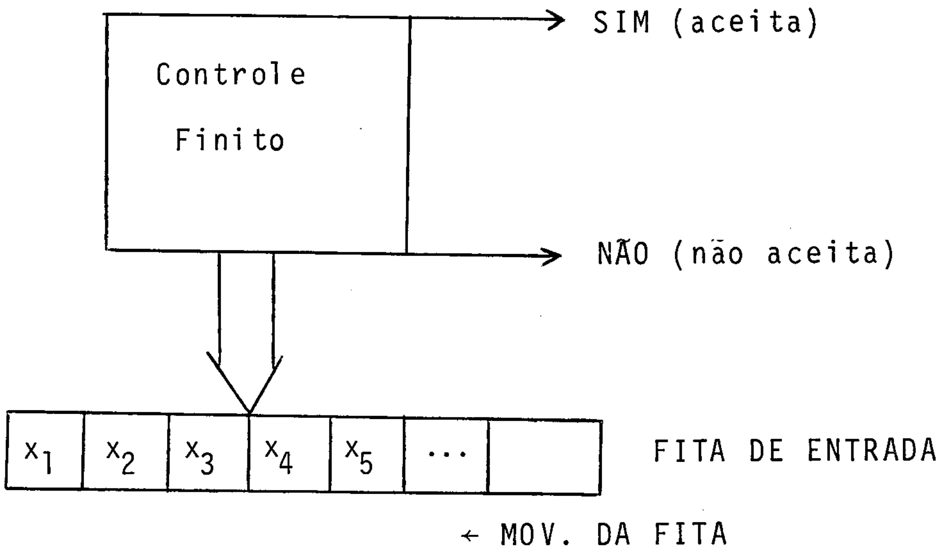
A linguagem reconhecida por um autômato A, consiste de todas as sequências $x \in \Sigma^*$ aceitas por A, o que significa:

$$R(A) = \{x \in \Sigma^* \mid f(s_0,x) \in F\} \qquad \text{(Moore}^5\text{).}$$

NOTA: Uma linguagem sobre Σ é um conjunto L de palavras de símbolos de Σ (alfabeto finito e não vazio). L é um subconjunto de Σ^* (Hopcroft, Ullman[4]).

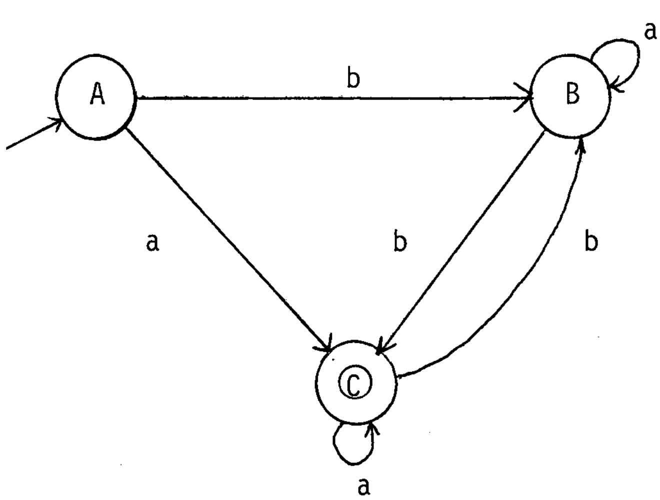
II.2.3. INTERPRETAÇÃO

Inicialmente, o autômato finito está no estado inicial s_0, a cabeça está sobre o primeiro símbolo mais à esquerda da fita. A cada símbolo lido, a máquina muda ou não de estado. Se após o último símbolo o AF para em um estado final, então, a fita é aceita, caso contrário ela é rejeitada.



II.2.4. EXEMPLOS

i) $A_1 = <\{a, b\}, \{A, B, C\}, f, A, \{C\}>$



ii) $A_2 = \langle\{0, 1\}, \{A, B\}, f, A, \{B\}\rangle$

	0	1
→ A	A	B
<u>B</u>	B	B

(Tabela de Transição)

<u>Nota</u>: O estado inicial é indicado pela seta e o final pelos dois círculos ou pelo traço na tabela de transição.

II.2.5. <u>DEFINIÇÃO</u>

Um autômato é <u>reduzido</u> se e somente se não tem estados distintos equivalentes isto é, $\forall$ s, t ε S.

$$s \equiv t \Longrightarrow s = t$$

sendo <u>equivalentes</u> estados que aceitam as mesmas palavras, isto é, $s \equiv t$ se e somente se para todo $x \varepsilon \Sigma^*$ $f(s, x) \varepsilon F \Longleftrightarrow f(t, x) \varepsilon F$.

II.2.6. <u>DEFINIÇÃO</u>

Um autômato é <u>conexo</u> se e somente se é conexo em relação ao seu estado inicial, isto é, $f(s_0, \Sigma^*) = S$

(todos os seus estados são alcançáveis a partir do estado s_0).

II.2.7. DEFINIÇÃO

Um autômato é minimal se é reduzido e conexo.

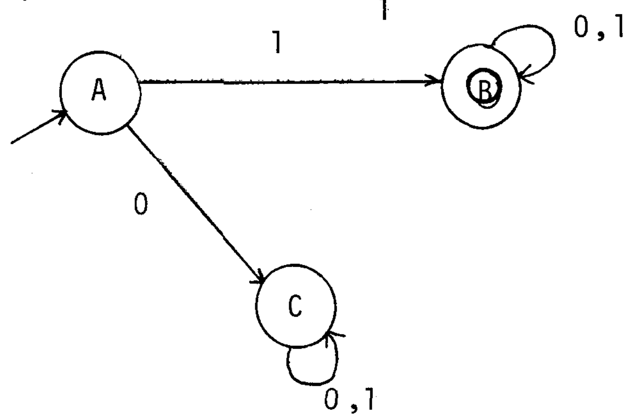
II.3. LINGUAGEM REGULAR

II.3.1. DEFINIÇÃO

Uma linguagem $L \subseteq \Sigma^*$ é regular se e somente se é reconhecida por um autômato finito, isto é, existe um $A = \langle \Sigma, S, f, s_0, F \rangle$, tal que $L = R(A)$ (Hopcroft, Ullman[4]).

II.3.2. EXEMPLOS

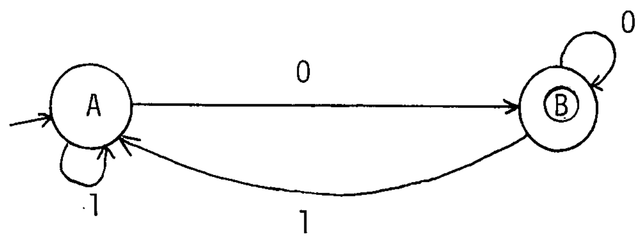
i) Seja o autômato A_1



A linguagem regular reconhecida por A_1 é:

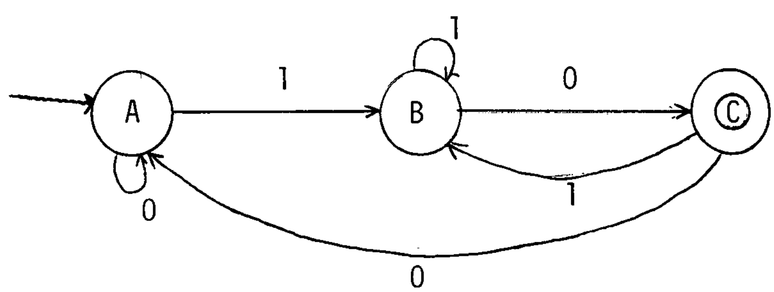
$$L = R(A_1) = 1\ \Sigma^* = 1\ (0 + 1)^*$$

ii) Seja A_2,



$$L = R(A_2) = 1^*\ 0\ (0 + 11^*0)^*$$

iii) Seja A_3,



$$L = R(A_3) = \Sigma^*\ 10$$

II.3.3. <u>DEFINIÇÃO</u>

Uma linguagem é <u>finita</u> se tem um número finito de sequências.

<u>Nota</u>: As linguagens regulares são fechadas sob união, interseção, complemento, concatenação e estrela |Hopcroft, Ullman[4]|.

C A P Í T U L O III

III. MÁQUINA DE MEMÓRIA FINITA

Neste capítulo definiremos as máquinas de memória finita (abreviando MMF) e alguns exemplos serão apresentados.

III.1. DEFINIÇÃO

M é uma máquina de memória finita de ordem K (K - MMF) se e somente se $\forall$ s, t ε S, $\forall$ w ε Σ^K, $\forall$ σ ε Σ.

$$g\,|f\,(s,\ w),\ \sigma| = g|f\,(t,\ w),\ \sigma| \text{ sempre que:}$$

$$g(s,\ w) = g(t,\ w)$$

sendo K o menor natural para o qual isto se verifica.

Nota: Uma máquina de memória finita é de ordem K se são necessários pelo menos K símbolos de entrada e saída para determinarmos a próxima saída, independente do estado inicial.

Uma máquina é MMF se for K-MMF para algum $K > 0$.

III.2. LEMA

Se uma māquina sequencial ē de memōria finita sua reduzida tambēm ē

Demonstração: Ver |Veloso[7] p. 13|.

III.3. DEFINIÇÃO

Se M, (figura 3.1) ē uma māquina sequencial reduzida, a tabela de teste de M (figura 3.2), ē construīda da seguinte maneira:

A primeira linha contēm todas as combinações de entrada/saīda e para cada linha correspondente a cada estado, preenchemos colocando o estado correspondente a combinação entrada/saīda.

Completamos a tabela acrescentando novas linhas que correspondem a pares de estados, ($S_i\ S_j = S_j\ S_i$ e $i \neq j$).

Cada par gera novos pares que serão inseridos em uma nova linha e esses gerarão novos pares atē não aparecer mais nenhum par diferente.

	0	1
A	B/0	C/1
B	D/0	C/0
C	D/0	B/1
D	C/0	A/0

(figura 3.1)

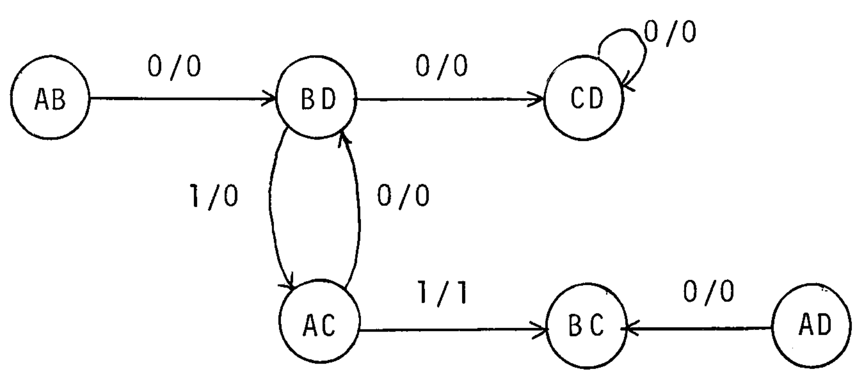
	0/0	0/1	1/0	1/1
A	B	-	-	C
B	D	-	C	-
C	D	-	-	B
D	C	-	A	-
AB	BD	-	-	-
AC	BD	-	-	BC
AD	BC	-	-	-
BD	CD	-	AC	-
CD	CD	-	-	-

(figura 3.2)

III.4. DEFINIÇÃO

O *grafo de teste* de M (figura 3.3) é construído da seguinte maneira:

Cada par da tabela de teste representa um vértice no grafo, que será ligado a outro vértice de acordo com a combinação de entrada/saída de sua coluna.



(figura 3.3)

III.5. TEOREMA

Uma condição necessária e suficiente para que uma máquina sequencial M seja de memória finita é que seu grafo de teste seja livre de circuitos. (Sendo ℓ o comprimento do maior caminho do grafo de teste, então, $K = \ell + 1$).

Demonstração: Ver |Kohavi[1] p. 447|.

III.6. DEFINIÇÃO

Um circuito entrada-saída (abreviando E/S) de um estado é uma sequência não nula de pares ordenados de entrada/saída que leva dele a ele mesmo, isto é:

para $s \in S$

$$E/S\ (s) = \{(u, y) \in (\Sigma \times \Delta)^+ \mid f(s, u) = s,$$
$$g(s, u) = y\}$$

III.7. DEFINIÇÃO

Um circuito de entrada (abreviando E) de um estado é uma sequência não nula de entradas que leva dele a ele mesmo, isto é:

para $s \varepsilon S$

$$E(s) = \{u \ \varepsilon \ \Sigma^+ \ | f\ (s, u) = s\}$$

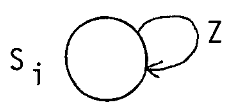
III.8. TEOREMA

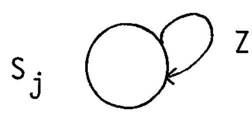
Uma condição necessária e suficiente para que uma máquina sequencial reduzida seja de memória finita é que dois estados distintos S_i e S_j não tenham circuitos E/S iguais.

Demonstração:

1. necessidade (por contraposição)

Sejam S_i e S_j com um mesmo circuito E/S Z

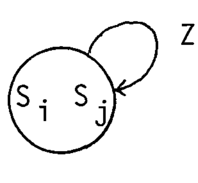




As sequências de entrada/saída $Z^n \ \forall n$, não podem distinguir S_i e S_j, logo a máquina reduzida não é de memória finita.

2. suficiência (por contraposição)

Suponhamos que a máquina não seja de memória finita. Então, seu grafo de teste (def.III.4) tem pelo menos um circuito, isto é, existe um par $\{S_i, S_j\}$ com um circuito Z.



Para isto acontecer só existem duas possibi lidades:

i) $S_i \xrightarrow{Z} S_i$

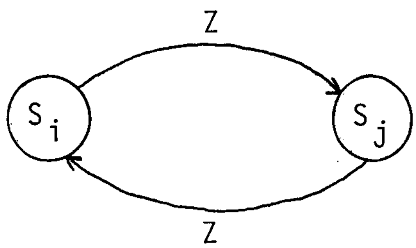
$S_j \xrightarrow{Z} S_j$

tendo assim dois estados com circuitos E/S iguais.

ii) $S_i \xrightarrow{Z} S_j$

$S_j \xrightarrow{Z} S_i$

Neste caso podemos formar o circuito E/S = Z.Z tanto em S_i como em S_j, portanto, teremos dois circuitos iguais.



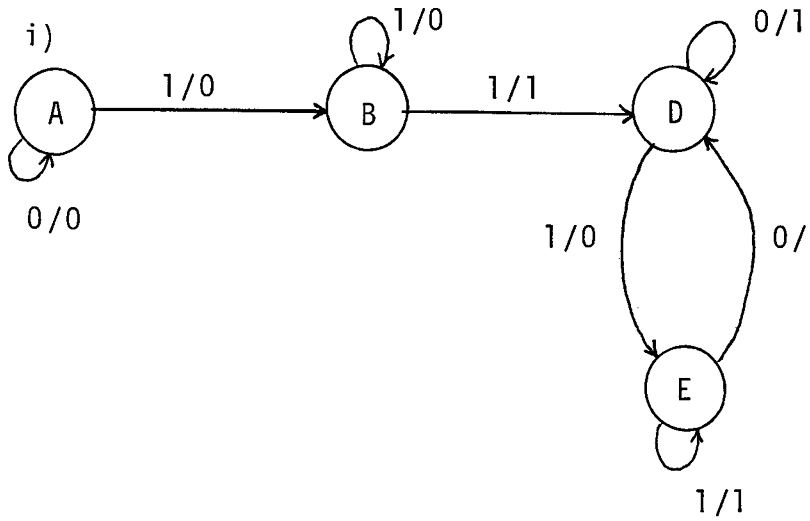
$S_i \xrightarrow{Z.Z} S_i$

$S_j \xrightarrow{Z.Z} S_j$

<u>Nota:</u> No caso de a máquina não ser reduzida, a existência de dois estados com circuitos E/S iguais significa que a máquina não é de memória finita ou que a máquina é de memória finita e os dois estados são equivalentes.

III.9. EXEMPLOS

i)



Esta máquina é de memória finita, estados distintos, não possuem circuitos iguais

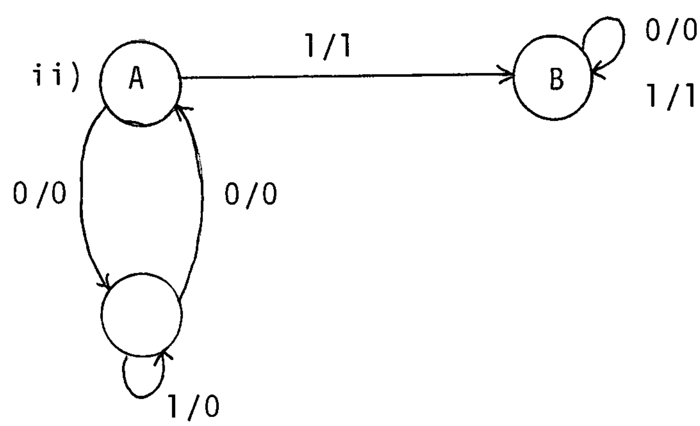
$$E/S\ (A) = (0/0)^+$$

$$E/S\ (B) = (1/0)^+$$

$$E/S\ (D) = |1/0\ (1/1)^*\ 0/1 + (0/1)|^+$$

$$E/S\ (E) = |0/1\ (0/1)^*\ 1/0 + (1/1)|^+$$

ii)



Esta máquina não é uma MMF, pois com o circuito E/S 0/0 0/0 nãodistinguimos o estado A de C nem de B e eles não são equivalentes.

$$E/S\ (A) = |0/0\ (1/0)^*\ 0/0|^*$$

$$E/S\ (B) = |(0/0) + (1/1)|^{+}$$

$$E/S\ (C) = |0/0\ 0/0\ (1/0)^{*}|^{+} + (1/0)^{+}$$

III.10. MÁQUINA DEFINIDA

Máquinas definidas (abreviando MD), são caso especial das máquinas de memória finita, não é necessário conhecermos a sequência de saída para determinarmos a saída final.

III.11. DEFINIÇÃO

M é definida de ordem K (K - MD) se e somente se $\forall s, t \in S, \forall w \in \Sigma^K, \forall \sigma \in \Sigma$

$$g|f(s, w), \sigma| = g|f(t, w), \sigma| \quad e$$

$$\forall i < K \quad \exists \quad s, t \ \exists \quad u \in \Sigma^i \quad \exists \quad \sigma \in \Sigma$$

$$g|f(s, u), \sigma| \neq g|f(t, u), \sigma|$$

Nota: Uma máquina sequencial reduzida é definida de ordem K se K é o menor inteiro que determina o estado presente de M conhecendo os últimos K símbolos de entrada.

III.12. TEOREMA

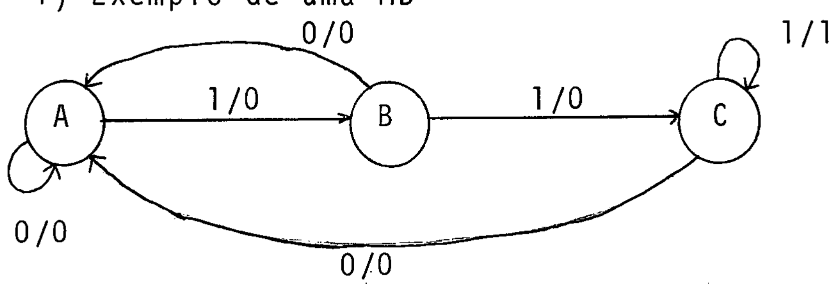
Uma condição necessária e suficiente para que uma máquina sequencial reduzida seja definida é que dois estados distintos S_i e S_j não tenham circuitos E iguais.

Demonstração: Análoga à do Teorema III.8.

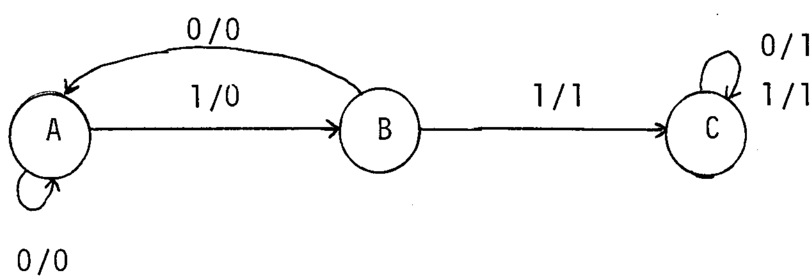
Nota: Se M é uma MD é também uma MMF.

III.13. Exemplos:

i) Exemplo de uma MD



ii) Exemplo de uma MMF não MD



C A P Í T U L O IV

IV. AUTÔMATO DE MEMÓRIA FINITA

Estudaremos agora os autômatos de memória finita (abreviando AMF).

IV.1. DEFINIÇÃO

Um autômato $A = <\Sigma, S, f, s_0, F>$ é de memória finita se e somente se a máquina de Mealy

$M = <\Sigma', S', \Delta, f', g>$ tal que

$\Sigma' = \Sigma$

$S' = S$

$f' = f$

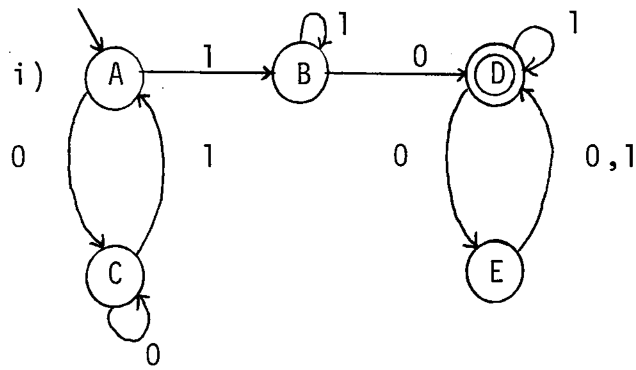
$\Delta = \{0,1\}$

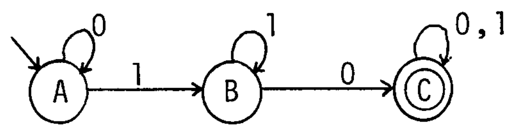
$\forall s \in S$ e $\forall w \in \Sigma'$

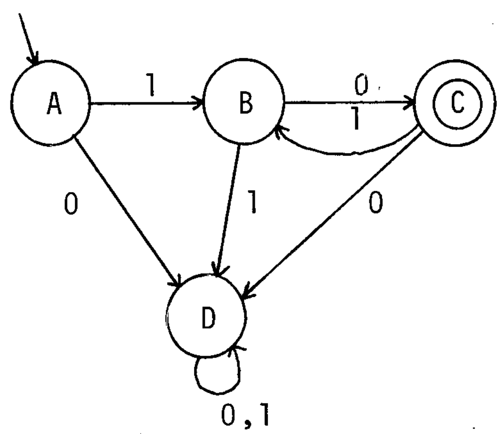
$$g(s, w) = \begin{cases} 1 \text{ se } f(s, w) \in F \\ 0 \text{ se } f(s, w) \notin F \end{cases}$$

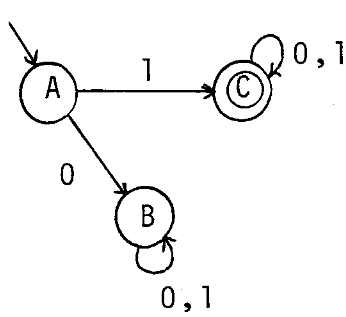
for de memória finita.

IV.2. EXEMPLOS DE AUTÔMATOS DE MEMÓRIA FINITA

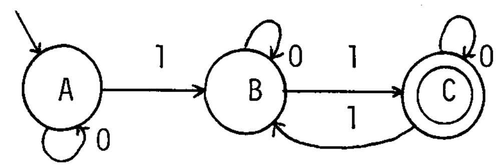
i) 

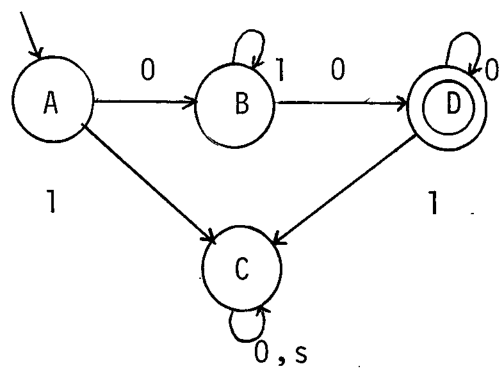
ii) 

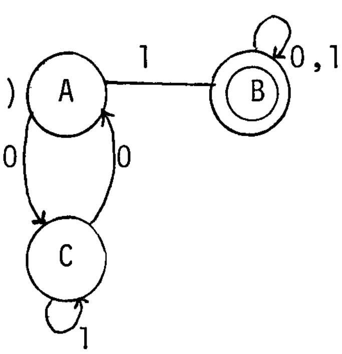
iii) 

iv) 

IV.3. EXEMPLOS DE AUTÔMATOS DE MEMÓRIA NÃO FINITA

i) 

ii) 

iii) 

IV.4. DEFINIÇÃO

Um autômato $A = <\Sigma, S, f, s_0, F>$ é definido (abreviando AD) se a máquina de Mealy

$M = <\Sigma', S', \Delta, f', g>$ tal que

$\Sigma' = \Sigma$

$S' = S$

$f' = f$

$\Delta = \{0,1\}$

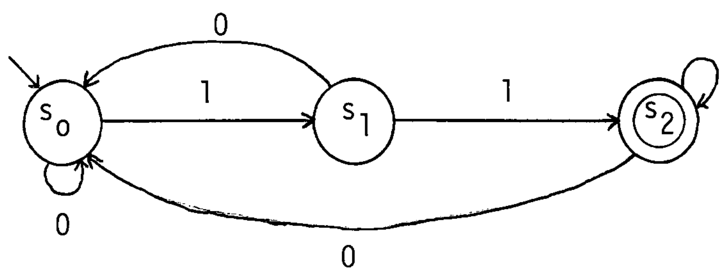
$\forall s \in S'$ e $\forall w \in \Sigma'$

$$g(s, w) = \begin{cases} 1 \text{ se } f(s, w) \in F \\ 0 \text{ se } f(s, w) \notin F \end{cases}$$

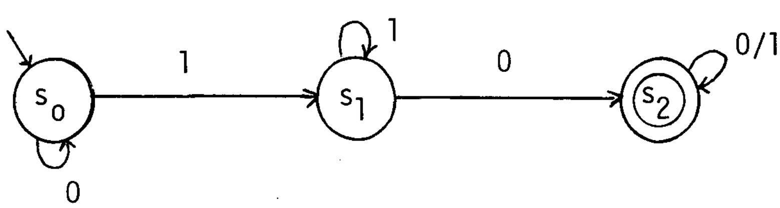
for definida.

<u>Nota</u>: O conjunto dos autômatos de memória finita (AMF) contém os autômatos definidos (AD).

i) Exemplo de AD



ii) Exemplo de um AMF não AD



IV.5. <u>DEFINIÇÃO</u>

Dizemos que dois estados s e t são compatíveis (notado $s \sim t$) se e somente se $w \in \Sigma^+$ tal que

$f(s, w) = s$, $f(t, w) = t$ e

$\forall$ x prefixo de w

$f(s, x) \in F \leftrightarrow f(t, x) \in F$

IV.6. TEOREMA

Um autômato reduzido A é um AMF se e somente se:

$\forall s, t \in S, s \neq t \Longrightarrow s \not\sim t$

Demonstração

i) $(\Longrightarrow)$

Seja s, t $(s \neq t)$ tal que $s \sim t$ então $w \in \Sigma^+$, $f(s, w) = s$, $f(t, w) = t$ e x pref. w$f(s, w) \in F \leftrightarrow f(t, x) \in F$

Na máquina sequencial M obtida através da definição IV.1. temos:

$f(s, w) = s$, $f(t, w) = t$ e

$\forall$ x pref. w $\hat{g}(s, x) = \hat{g}(t, x)$, portanto um

$g(s, w) = g(t, w)$

Logo s e t tem circuitos E/S iguais, portanto M não é uma MMF e A não é uma AMF.

ii) $(\Longleftarrow)$

Se A não é AMF então pela definição IV.1. M não é uma MMF o que implica $\exists$ s, t, $s \neq t$ tal que tem circuitos E/S iguais.

Seja w/y este circuito $f(s, w) = s$ e $f(t, w) = t$ e $g(s, w) = g(t, w)$.

Então, para todo x pref. w

$$f(s, x) \in F \leftrightarrow f(t, x) \in F$$

C A P Í T U L O V

V. LINGUAGEM DE MEMÓRIA FINITA

Neste capítulo definiremos as linguagens de memória finita e estudaremos também as linguagens definidas . Alguns exemplos serão apresentados.

V.1. DEFINIÇÃO

Uma linguagem $L \subseteq \Sigma^*$ é de memória finita (abreviando LMF) se é reconhecida por um autômato de memória finita, isto é, existe um autômato A que é um AMF tal que reconhece L

$$L = R(A)$$

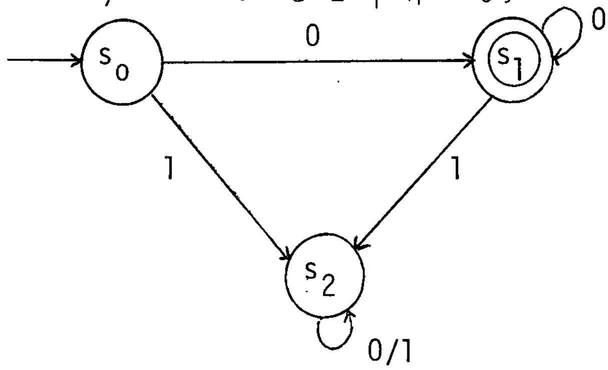
V.1.1. LEMA

L é uma LMF se e somente se o autômato minimal que a reconhece é um AMF.

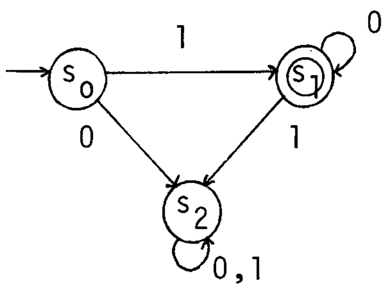
Demonstração : Decorre das definições V.1. e IV.1. e do Lema III.2.

V.1.2. EXEMPLOS DE LINGUAGENS DE MEMÓRIA FINITA

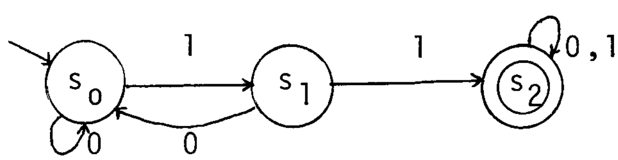
i) $L = <0^n \, \varepsilon \, \Sigma^* | \, \eta > 0\}$



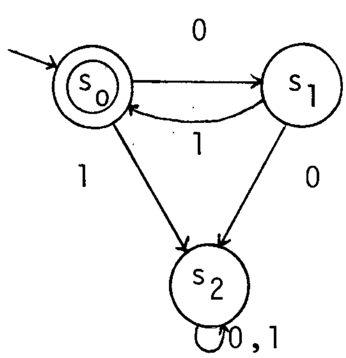
ii) $L = 10^*$



iii) $L = \Sigma^* \, 11 \, \Sigma^*$

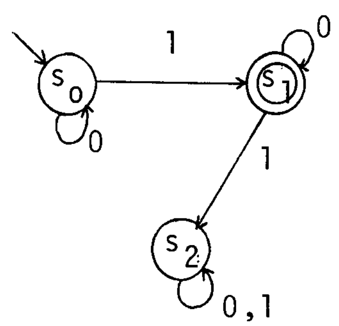


iv) $L = (0 \; 1)^*$

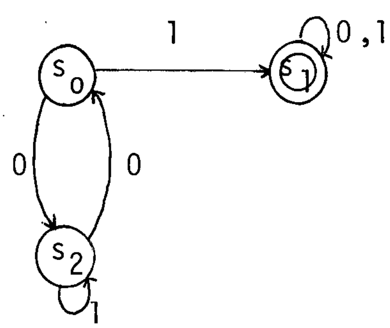


V.1.3. EXEMPLO DE LINGUAGEM REGULAR NÃO LMF

i) $L = 0^* \ 1 \ 0^*$



ii) $L = (0 \ 1^* \ 0)^* \ 1 \ \Sigma^*$



V.2. LINGUAGEM DEFINIDA

V.2.1. DEFINIÇÃO

Uma linguagem $L \subseteq \Sigma^*$ é definida (abreviando LD) se é reconhecida por um AD, isto é, existe A que é um AD tal que A aceita L

$$L = R(A)$$

V.2.2. TEOREMA

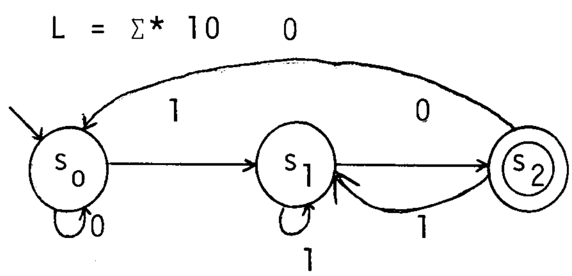
L é LD se e somente se pode ser escrita na forma A U Σ^* B onde A e B são finitos.

Demonstração: Ver |Perles, Rabin, Shamir[2]|.

Nota: Em |Perles, Rabin, Shamir[2]| é seguido um caminho inverso, parte-se da definição de linguagem definida idêntica ao item V.2.2. e demonstra-se um teorema equivalente ao item V.2.1.

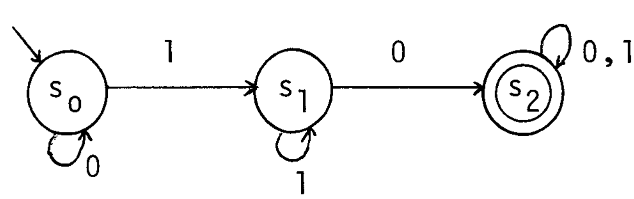
i) Exemplo de uma LD

$L = \Sigma^* 10$



ii) Exemplo de LMF não LD

$L = \Sigma^* 10 \; \Sigma^*$



V.3. <u>TEOREMA</u>

Toda linguagem da forma $L = A_1 B_1{}^* \cup A_2 B_2{}^* \cup \ldots \cup A_n B_n{}^*$ com A_i e B_i palavras e B_i distintos é uma LMF.

<u>Demonstração</u>: Seja A o autômato minimal que reconhece L, como $\bigcup_{i=1}^{n} A_i$ são finitos todas as palavras suficientemente longas que não fizerem parte da linguagem levarão A para o estado poço (sp). Além de sp, só os estados finais têm circuitos.

sp não é compatível com nenhum estado final (sfi) que tenha circuito pois

$$f(sp, B_i) = sp \notin F$$

$$f(sfi, B_i) = sfi \in F$$

e como os B_i são distintos então os sfi não são compatíveis entre si logo A é um AMF → L é LMF.

V.4. <u>TEOREMA</u>

Se L é LMF, $L\Sigma^*$ também é.

<u>Demonstração</u>: Seja $A = \langle\Sigma, S, f, s_0, F\rangle$ um autômato minimal que reconhece L e $A' = \langle\Sigma, S', f', s_0, \{sf\}\rangle$ um autômato tal que:

$S' = (S-F) \cup \{sf\}$, onde $sf \notin S$ e para cada

$a \in \Sigma$ e $s \in (S-F)$

$$f'(s, a) = \begin{cases} f(s, a) & \text{se } f(s, a) \notin F \\ sf & \text{se } f(s, a) \in F \end{cases}$$

$f'(sf, a) = sf$

i) A' só aceita palavras de $L\ \Sigma^*$

Suponha que $f'(s_0, w) = sf$

Seja u o menor prefixo de w tal que $f'(s_0, u) = sf$ e escreva $w = uV$ com $v \in \Sigma^*$.

Então, para todo prefixo próprio x de u, temos:

$f'(s_0, x) \in (S-F)$, logo $f'(s_0, x) = f(s_0, x)$

Portanto, $f(s_0, u) \in F$ e $w = uV \in L\ \Sigma^*$.

ii) Por construção o sf não é compatível com nenhum outro estado, então A' é AMF.

V.5. TEOREMA

Se L é uma LMF, $L \cup \{w\}$, $(w \in \Sigma^*)$ também é uma LMF.

Demonstração: Seja $w = w_1 w_2 \dots w_n$ com $|w| = n$.

Seja $A = \langle \Sigma, S, f, s_0, F \rangle$ o autômato minimal que reconhece L e $A' = \langle \Sigma, S', f', s_0', F \cup F' \rangle$ um autômato construído como se segue.

A cada estado

$s_i = f(s_0, w_1, w_2, \ldots, w_i)$ associamos um

estado novo $s_i' \notin S$, para $i = 0, 1, \ldots, n$.

Tomamos $S' = S \cup \{S_i' \mid 0 \leq i \leq n\}$ e $F' =$
$= \{s_n'\} \cup \{S_i' \mid i < n \quad S_i \in F\}$.

Para cada $a \in \Sigma$, definimos $f'(s, a) = f(s, a)$ para $s \in S$.

$f'(s_n', a) = f|f(s_0, w), a|$ e para $i = 0, 1, \ldots, n - 1$.

$$f'(S_i', a) = \begin{cases} S_{i+1}' & \text{se } a = w_{i+1} \\ f(S_i, a) & \text{se } a \neq w_{i+1} \end{cases}$$

i) Por construção A' aceita w

$f'(s_0', w) = s_n' \in F'$

Note que se $w = \lambda$ então

$f'(s_0', \lambda) \in F'$

A' aceita todas as palavras de L, dado $x \in L$, $|x| = p$.

<u>1º Caso</u> - x tem prefixo $\neq \lambda$ em comum com w, seja $i = \max \{i \leq p | x_1 \ldots x_i = w_1 \ldots w_i\}$

$$f'(s'_0, x_1 \ldots x_i) = S'_i$$

$$f'(s'_i, x_{i+1} \ldots x_p) \in F \cup F'$$

<u>2º Caso</u>: Se $x_1 \neq w_1$

$$f'(s'_0, x) \in F$$

ii) A' só aceita palavras de $L \cup \{w\}$.

Suponha que $f'(s'_0, z) = sf \in F \cup F'$ seja u o menor prefixo de z tal que $f'(s'_0, u) = s'_n$ e escreva $z = uv$. Então, para todo prefixo próprio x de u temos

$$f'(s'_0, x) = S' - (F \cup F') \quad \text{logo}$$

$$f'(s'_0, x) = f(s_0, x) \quad \text{portanto}$$

$$f'(s'_0, u) \in F' \text{ e } z = uv \in L \cup \{w\}$$

iii) Por construção os novos estados não têm circuitos logo não são compatíveis, então A' é um AMF.

V.5.1. <u>COROLÁRIO</u>

Se L é uma LMF e P uma linguagem finita , $L \cup P$ é LMF.

V.6. TEOREMA

Se L é uma LMF, $L - \{w\}$, $(w \in \Sigma^*)$ também é uma LMF.

Demonstração: Seja $w = w_1, w_2, \ldots, w_n$ com $(w) = n$. Seja $A = <\Sigma, S, f, s_0, F>$ o autômato minimal que reconhece L e $A' = <\Sigma, S', f', s'_0, F \cup F'>$ um autômato construído como segue.

A cada estado $s_i = f(s_0, w_1, w_2, \ldots, w_i)$ associamos um estado novo $s'_i \notin S$, para $i = 0, 1, \ldots, n$.

Tomamos $S' = S \cup \{S'_i \mid 0 \leq i \leq n\}$ e $F' = \{S_i \mid 0 \leq i \leq n \quad S_i \in F\}$.

Se $w \notin L$ temos $L - \{w\} = L$.

Se $w \in L$ temos: para cada $a \in \Sigma$, definimos $f'(s, a) = f(s, a)$ para $s \in S$.

$f'(s'_n, a) = f|f(s_0, w), a|$ e para

$i = 0, 1, \ldots, n - 1$

$$f'(s'_i, a) = \begin{cases} s'_{i+1} & \text{se } a = w_{i+1} \\ f(s_i, a) & \text{se } a = w_{i+1} \end{cases}$$

i) A' aceita $L - \{w\}$ e A' é AMF. A prova é análoga ao Teorema V.5.

V.6.1. <u>COROLÁRIO</u>

Se L é uma LMF e P uma linguagem finita, L-P é LMF.

V.6.2. <u>COROLÁRIO</u>

Se P é uma LMF e (P-Q) U (Q-P) é finita então Q também é uma LMF.

V.7. <u>PROPOSIÇÃO</u>

As linguagens escritas nas seguintes formas: Σ^* B Σ^*, A Σ^* (A e B finitos) são LMF's.

<u>Demonstração</u>: Toda LD, L = (A U Σ^* B) é LMF e pelo (teorema V.4.) L Σ^* = (A U Σ^* B)Σ^* é LMF.

Fazendo A = $\phi \Longrightarrow \Sigma^* B \Sigma^* \in$ LMF

e B = $\phi \Longrightarrow A \Sigma^* \in$ LMF.

V.8. <u>PROPOSIÇÃO</u>

As linguagens definidas reversas (A U B Σ^*) são LMF's.

<u>Demonstração</u>:

$L_1 = B\ \Sigma^* \in LMF$ (por V.7)

$L = A \cup L_1 \in LMF.$

CAPÍTULO VI

VI. PROPRIEDADES DAS LINGUAGENS DE MEMÓRIA FINITA

Neste capítulo estudaremos algumas operações sobre as LMF's. Mostraremos que as LMF's só são fechadas sobre complemento e derivada.

VI.1. COMPLEMENTO

$$L = \{x \varepsilon \Sigma^* \mid x \varepsilon L\}$$

VI.1.1. TEOREMA

As LMF's são fechadas sob complemento.

Demonstração: Seja A um AMF que reconhece L $L = R(A)$.

Para construir um autômato A' que reconheça $\overline{L}$, basta trocar os estados finais por não finais e vice-versa.

$$A = \langle \Sigma, S, f, s_0, F \rangle$$

$$A' = \langle \Sigma, S, f, s_0, S-F \rangle$$

$$R(A') = \overline{R(A)}$$

O novo autômato A' é um AMF pois a transformação não torna compatíveis os estados.

VI.2. UNIÃO E INTERSEÇÃO

$$L_1 \cup L_2 = \langle x \in \Sigma^* \mid x \in L_1 \vee x \in L_2\}$$

$$L_1 \cap L_2 = \langle x \in \Sigma^* \mid x \in L_1 \wedge x \in L_2\}$$

VI.2.1. LEMA

As LMF's não são fechadas sob união ou interseção com as LD's.

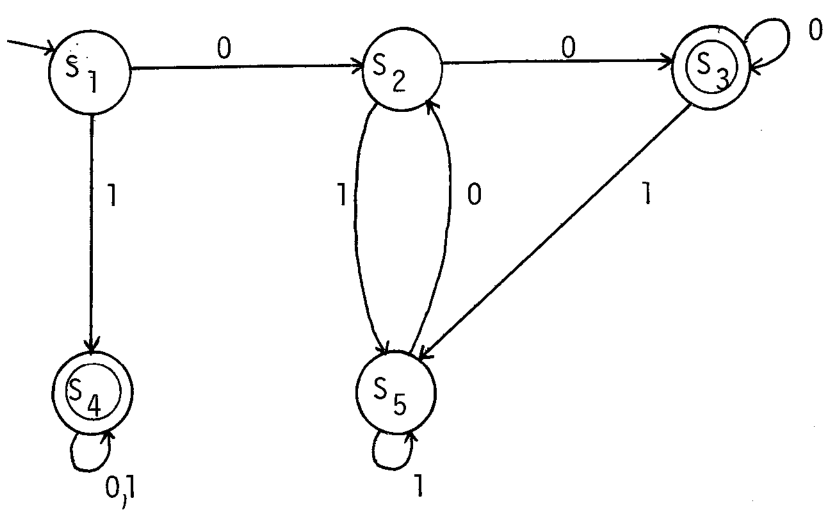
Demonstração:

i) União de $L_1 = \Sigma^* \, 0 \, 0$ (LD) e

$L_2 = 1 \, \Sigma^*$ (LMF)

$L = L_1 \cup L_2$.

O autômato finito minimal que reconhece L é



Os estados S_3 e S_4 são compatíveis

$f(S_3, 0) = S_3$, $f(S_4, 0) = S_4$

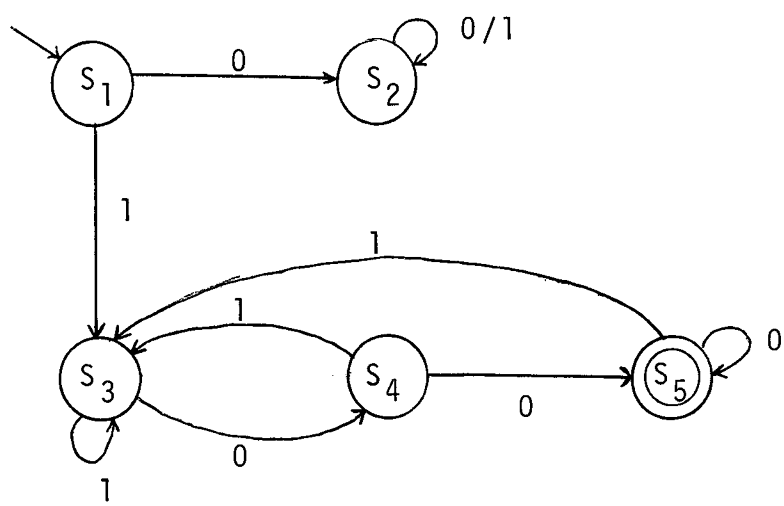
$f(S_3, 0) \in F$ $\leftrightarrow$ $f(S_4, 0) \in F$

logo A não é um AMF $\Longrightarrow$ L não é LMF.

ii) Interseção de L_1 e L_2 do exemplo anterior.

$L = L_1 \cap L_2 = 1 \ \Sigma^* \ 0 \ 0$

Seja A $\Longrightarrow$ L = R(A)



Os estados S_2, S_3 e S_4 são compatíveis

$f(S_2, 1) = S_2$, $f(S_3, 1) = S_3$

$f(S_2, 1) \notin F$ $\leftrightarrow$ $f(S_3, 1) \notin F$

o mesmo ocorre para

$$f(S_2,\ 01) = S_2 \quad , \quad f(S_3,\ 01) = S_3 \quad e$$

$$f(S_2,\ 10) = S_2 \quad , \quad f(S_4,\ 10) = S_4$$

então A não é AMF $\Longrightarrow$ L não é LMF.

VI.2.2. TEOREMA

As LMF's não são fechadas sob união ou interseção.

Demonstração: Conforme demonstração VI.2.1.

VI.2.3. CASOS PARTICULARES DE FECHAMENTO

i) Se L é LMF e L' é LF então L U L' ε LMF (teorema V.5.).

ii) Seja L ε LMF, L' ε LF chama-se co-finita $\overline{L'}$.

$L \cap \overline{L'}\ \varepsilon$ LMF.

iii) Seja L uma linguagem regular, L' ε LF e $\overline{L'}$ (co-finita).

L U $\overline{L'}$ (pois seu complemento é finito).

VI.3. CONCATENAÇÃO

A concatenação de L_1 com L_2 é

$$L = L_1 . L_2 = \{xy \mid x \varepsilon L_1, \ y \varepsilon L_2\}$$

VI.3.1. LEMA

As LMF's não são fechadas sob pós-concatenação por linguagem finita. A concatenação de duas LD's não precisa ser LMF.

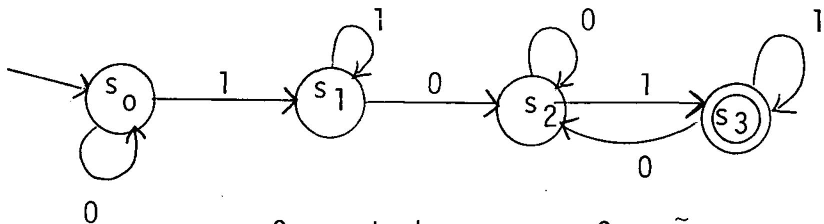
Demonstração: Seja

$L_1 = \Sigma^* \, 10 \, \Sigma^*$ (LMF)

$L_2 = 1$ (LF)

$L = \Sigma^* \, 10 \, \Sigma^* \, 1$

Construindo o autômato A que reconhece L temos:



Os estados s_0 e S_2 são compatíveis

$f(s_0, 0) = s_0$, $f(s_2, 0) = s_2$

$f(s_0, 0) \notin F \leftrightarrow f(s_2, 0) \notin F$

Logo A não é AMF $\Longrightarrow$ L não é LMF. Note que $L = (\Sigma^* 10).(\Sigma^* 1)$.

VI.3.2. <u>TEOREMA</u>

As LMF's não são fechadas para concatenação.

<u>Demonstração</u>: conforme demonstração VI.3.1.

VI.3.3. <u>CASOS PARTICULARES DE FECHAMENTO</u>

i) Seja $D = LL'$ com $L = \Sigma^* B$, $B \in LF$ e $L' \in LF$ (concatenação à direita do L), D é LMF.

ii) Seja $E = L'L$ com $L = B\Sigma^*$, $B \in LF$ e $L' \in LF$, E é LMF (concatenação à esquerda do L).

VI.4. <u>TRANSPORTA</u>

$$L^T = \{x^T \in \Sigma^* | x \in L\}$$

VI.4.1. <u>LEMA</u>

As LMF's não são fechadas sob pós-concatenação sob LD reversas.

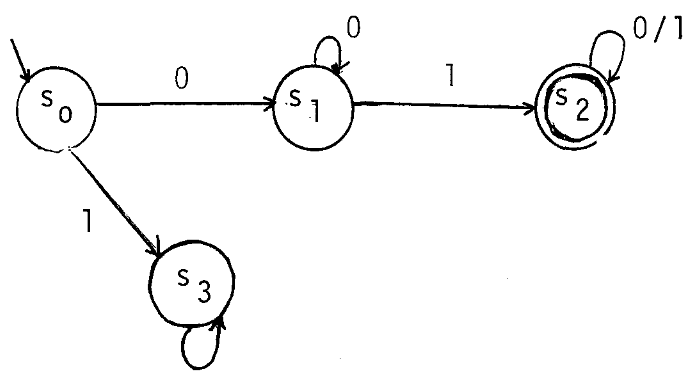
<u>Demonstração</u>:

$L_1 = 0^*$ (LMF por V.3).

$L_2 = 0\ 1\ \Sigma^*$ (LD reversa)

$L\ = 0^*\ 0\ 1\ \Sigma^*$

Construindo o autômato minimal A que reconhece L temos:



Os estados S_1 e S_3 são compatíveis

$f(S_1, 0) = S_1$, $f(S_3, 0) = S_3$

$f(S_1, 0) \notin F$ $\leftrightarrow$ $f(S_3, 0) \notin F$

logo A não é AMF $\Longrightarrow$ L não é LMF.

VI.4.2. <u>TEOREMA</u>

As LMF's não são fechadas para transposição.

<u>Demonstração</u>: Seja

$L = \Sigma^*\ 10\ 0^*$

$L^T = 0^*\ 01\ \Sigma^*$

$L^T \notin$ LMF (conforme demonstração de VI.4.1.).

VI.4.3. CASOS PARTICULARES DE FECHAMENTO

i) Se L é LMF das seguintes formas:

$\Sigma^* A \Sigma^*$, $A \Sigma^*$, A, $\Sigma^* A$ (A linguagem finita). L^T é LMF.

VI.5. DERIVADAS

Seja L uma linguagem sobre Σ e s uma palavra sobre Σ. A derivada de L em relação a s é

$$D_s L = \{t \mid st \in L\}$$

VI.5.1. TEOREMA

As derivadas de uma LMF são LMF.

Demonstração: Seja L uma LMF e o autômato minimal $A = \langle\Sigma, S, f, s_0, F\rangle$ tal que $L = R(A)$.

Por |Brzozowski[6]| as derivadas de L são reconhecidas pelos $A_i = \langle\Sigma, S, f, s_i, F\rangle$, $\forall s_i \in S$.

Então as $L_i = R(A_i)$.

Os pares de estados compatíveis em A_i são exatamente os mesmos de A.

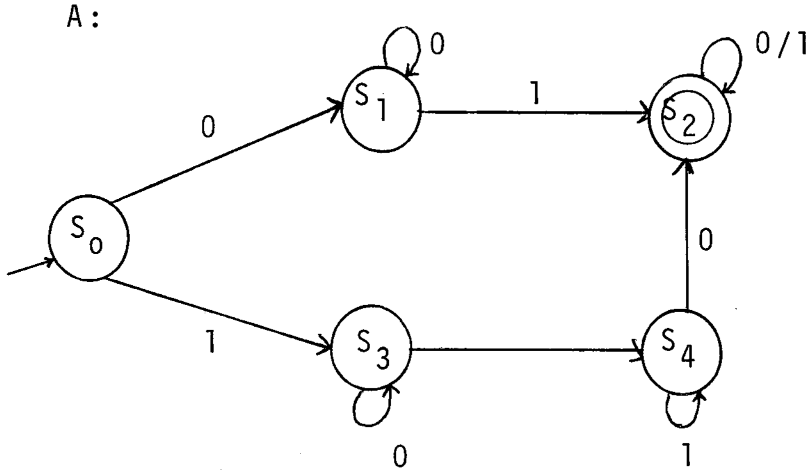
VI.5.2. NOTA

Se para cada $\nabla \in \Sigma$, $D_\sigma L$ é LMF, então L pode não ser LMF.

Exemplo:

$L = R(A)$

A:



$D_0L = 0\ 0^*\ 1\ \Sigma^*$ e $D_1L = 1\ 0^*\ 1\ 1^*\ 0\ \Sigma^*$ são LMF's mas A não é um AMF $\Longrightarrow L = (0 + 1 + 10^*\ 1^+)\ \Sigma^*$ não é LMF.

VI.6. <u>ESTRELA</u>

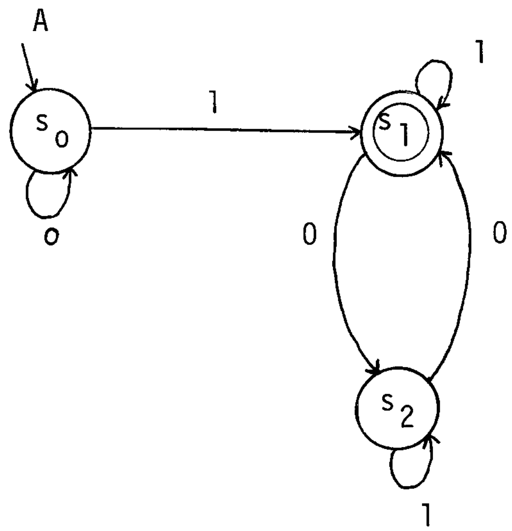
$$L^* = L^0 \cup L^1 \cup L^2 \cup \ldots \cup L^n = \bigcup_{n=0}^{\infty} L^n .$$

VI.6.1. <u>TEOREMA</u>

As LMF's não são fechadas sob estrela.

<u>Demonstração</u>: Seja

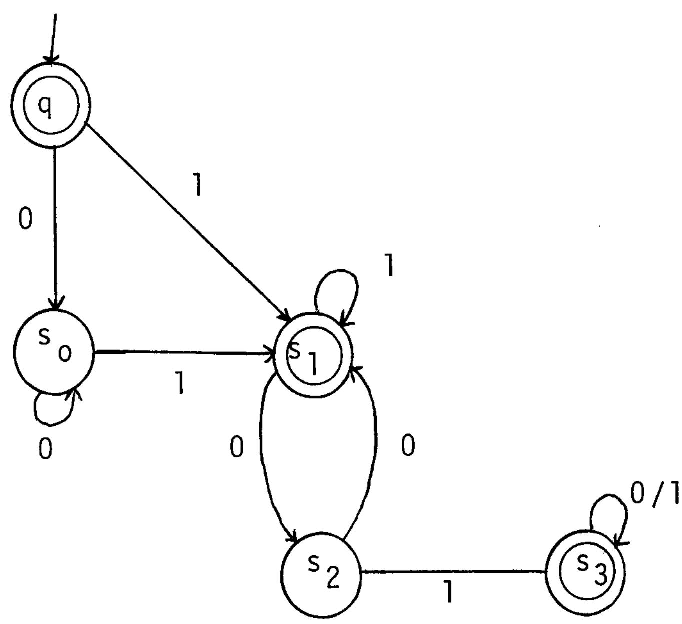
$L = 0^*\ 1\ (1 \times 01^*\ 0)^*$ e A o autômato minimal que reconhece L.



A é LMF $\Longrightarrow$ L é LMF.

Então $L^* = (0^*\ 1\ (1 + 01^*\ 0)^*)^*$.

Construindo o autômato minimal A' que reconhece L^* temos:



Os estados s_1 e s_3 são compatíveis

$f(s_1, 1) = s_1$

$f(s_3, 1) = s_3$

$f(s_1, 1)\ \varepsilon\ F \leftrightarrow f(s_3, 1)\ \varepsilon$

logo A' não é um AMF $\Longrightarrow$ L^* não é LMF.

VI.6.2. CASOS PARTICULARES DE FECHAMENTO

i) Se $L\ \Sigma^*$ é LMF então $(L\ \Sigma^*)^*$ é LMF.

ii) Estrela de $L\ \Sigma^*$ para $L\ \Sigma^* = (A \cup \Sigma^*\ B)\Sigma^*$ é LMF.

iii) Estrela de $L = \Sigma^*\ B$ é LMF.

iv) Estrela de $L = \{w\}$ $(w \in \Sigma^*)$ é LMF.

Pois (iv) é caso particular do teorema V.5. e nos primeiros casos a estrela se reduz a incluir λ.

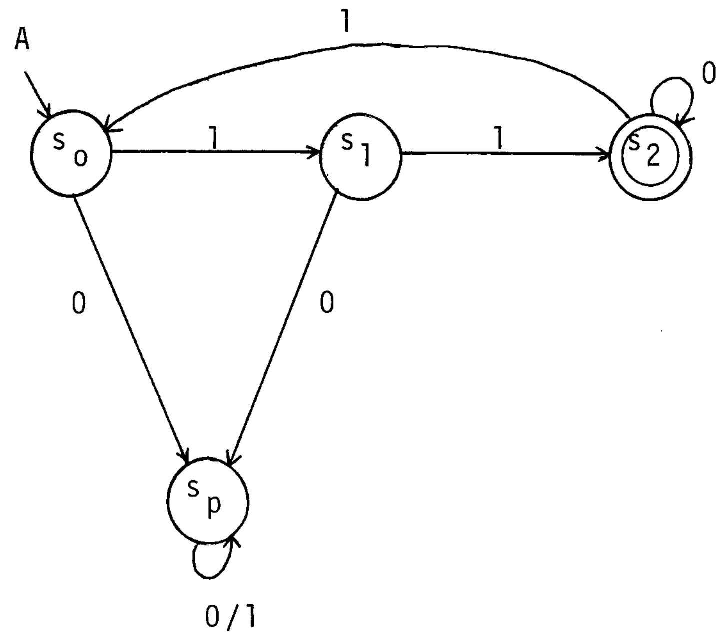
VI.7. QUOCIENTES

VI.7.1. TEOREMA

As LMF's não são fechadas sob quocientes à direita P/Q onde:

$$P/Q = \{u \mid \exists\ v \in Q \quad uv \in P\}$$

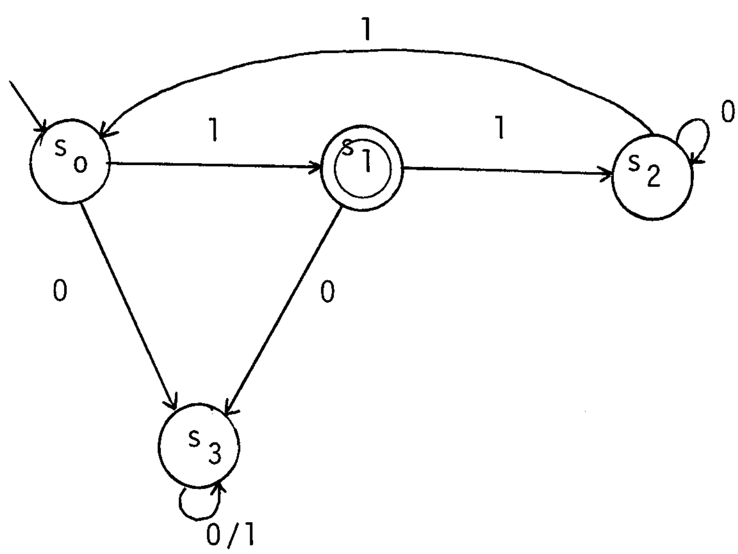
Demonstração: Sejam $P = 11\ (0 + 111)^*$ e A o autômato minimal que reconhece P.



A é AMF ⟹ P é LMF.

Q = 10 ε LMF. Então P/Q = 1 (10* 11)*.

Construindo o autômato minimal A' que reconhece P/Q temos:



Os estados s_2 e s_3 são compatíveis

$f(s_2, 0) = s_2$

$f(s_3, 0) = s_3$

$$f(s_2, 0) \notin F \iff f(s_3, 0) \notin F$$

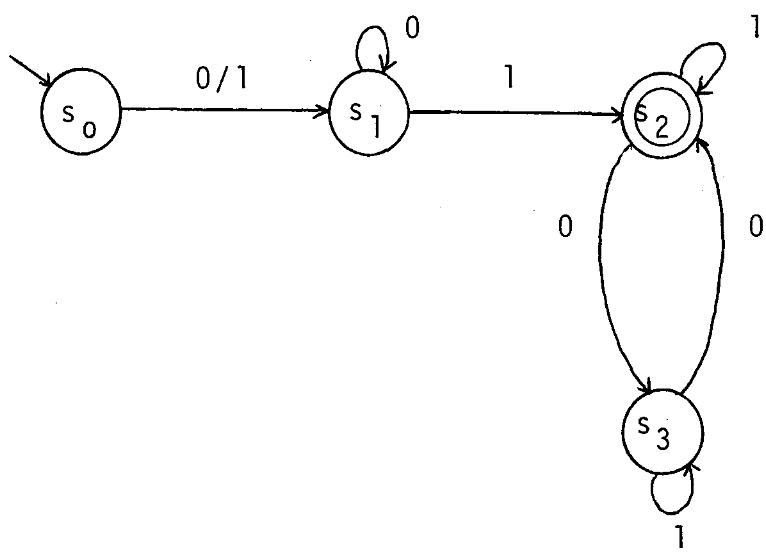
logo A' não é AMF $\Longrightarrow$ P/Q não é LMF.

VI.7.2. TEOREMA

As LMF's não são fechadas sob quociente à esquerda $Q \backslash P = \{v \mid \exists\ u \in Q \quad uv \in P\}$.

Demonstração: Temos $Q \backslash P = \bigcup_{u \in Q} D_u P$.

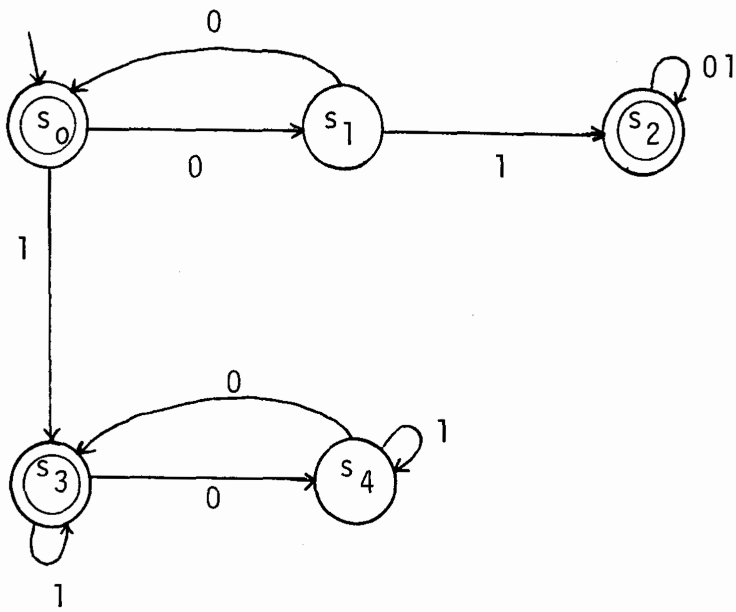
Sejam $P = (0 + 1)\ 0^*\ 1(1 + 01^*\ 0)$ e A o autômato minimal que a reconhece



A é AMF $\Longrightarrow$ P é LMF.

$Q = (0 + 01) \in$ LMF. Então $D_0P = 0^*1(1 + 01^*0)^*$ e $D_{01}P = (1 + 01^*0)^*$.

Construindo o autômato minimal A' que reconhece $D_0P \cup D_{01}P = (\lambda + 0^*1)(1 + 01^*0)^*$ temos:



Os estados s_0 e s_3 são compatíveis

$f(s_0, 0\ 0) = s_0$

$f(s_3, 0\ 0) = s_3$

$f(s_0, 0\ 0) \in F \leftrightarrow f(s_3, 0\ 0) \in F$

$f(s_0, 0) \notin F \leftrightarrow f(s_3, 0) \notin F$

Os estados s_3 com s_2 e s_1 com s_4 são também compatíveis, logo A' não é um AMF $\rightarrow$ Q\P não é LMF.

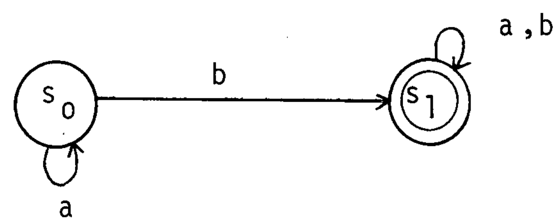
VI.8. IMAGEM HOMOMORFA

$\phi(L) = \{\phi(w) \mid w \in L\}$, com $\phi: \Sigma \rightarrow \Delta^*$ extendido a Σ^*.

VI.8.1. TEOREMA

As LMF's não são fechadas para imagens homomorfa.

Demonstração: Seja $L = a^* b (a + b)^*$ e A o autômato minimal que a reconhece



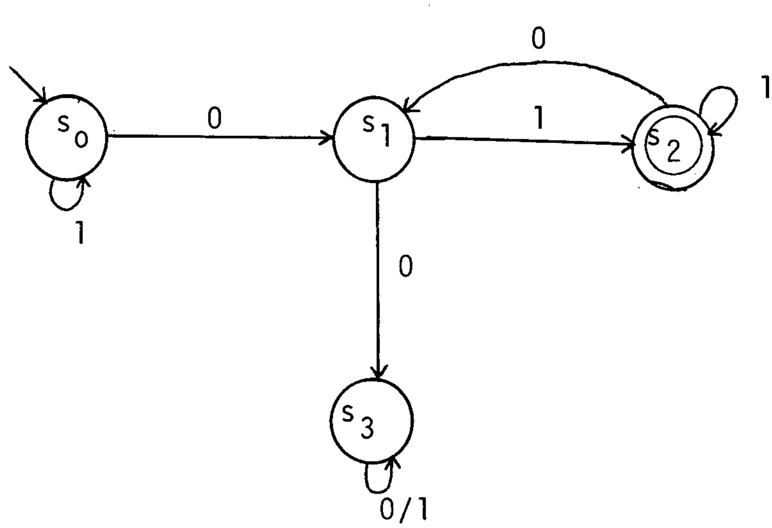
A é AMF $\Longrightarrow$ L é LMF.

Seja ϕ: $a \rightarrow 1$

$b \rightarrow 01$

$$\phi(L) = 1^* \, 0 \, 1 \, (1 + 01)^*$$

Construindo o autômato minimal A' que reconhece $\phi(L)$ temos:



Os estados s_0 e s_3 são compatíveis

$f(s_0, 1) = S_0$

$f(s_3, 1) = S_3$

$f(s_0, 1) \varepsilon F \leftrightarrow f(s_3, 1) \varepsilon F$

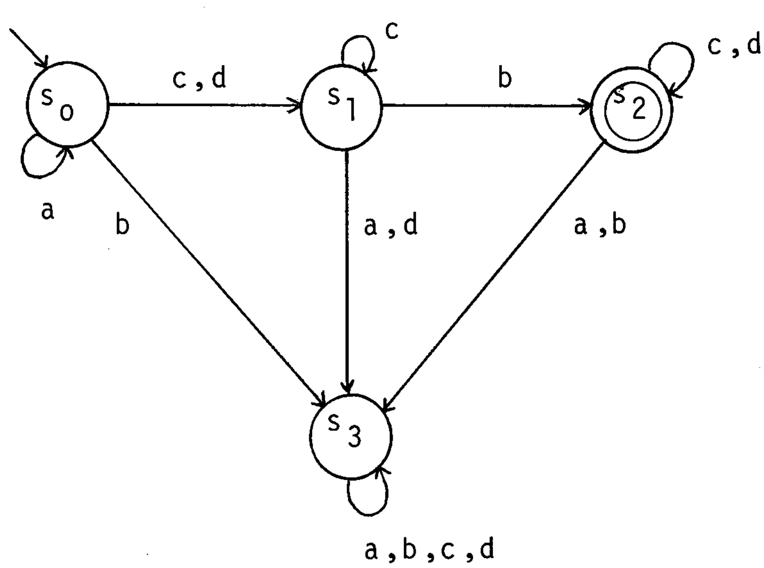
logo A' não é um AMF $\rightarrow \phi(L)$ não é LMF.

VI.8.2. <u>TEOREMA</u>

$\overline{LMF}$ não é fechada sob imagem homomorfa, isto é,

$\phi(L)$ para $L \notin$ LMF pode ser uma LMF.

<u>Demonstração</u>: Seja $L = a^* (c + d) c^* b (c + d)^*$, construíndo o autômato minimal A que reconhece L temos:



os estados s_o, s_1 e s_3 são compatíveis, logo A não é AMF $\Longrightarrow$ L não é LMF.

Seja ϕ: a $\rightarrow$ 01

b $\rightarrow$ 0

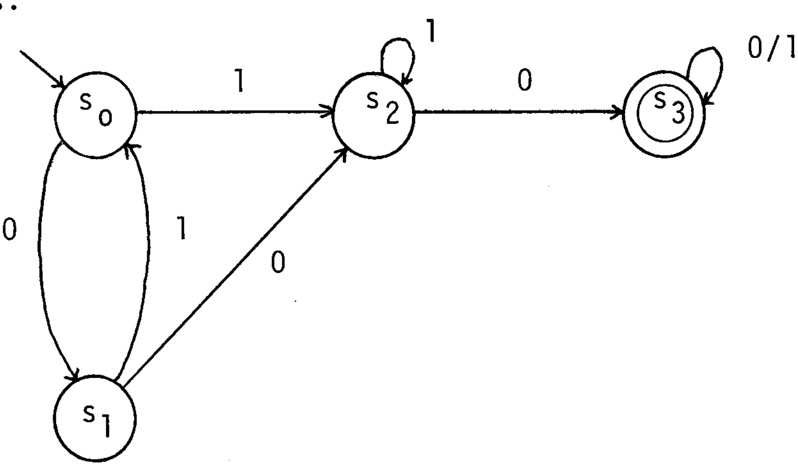
c $\rightarrow$ 1

d $\rightarrow$ 00

temos:

$$\phi(L) = (01)^* \ (1 + 00)1^* \ 0(1 + 0)^*$$

Construindo um autômato minimal A' que reconhece $\phi(L)$ temos:



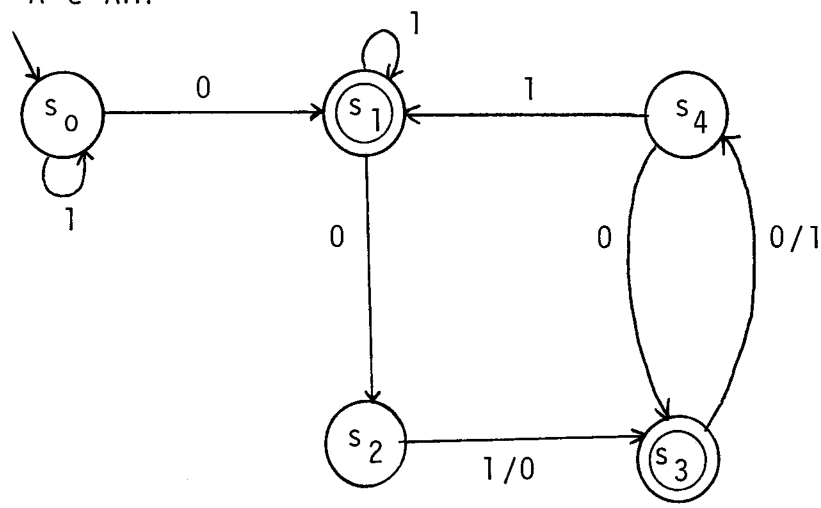
A' não tem estados compatíveis logo é um AMF $\Longrightarrow$ $\phi(L)$ é LMF.

VI.8.3. TEOREMA

As LMF's não são fechadas sob imagem homomorfa inversa, isto é, L ε LMF não implica em $\phi^{-1}(L) = \{w|\phi(w) \varepsilon L\}$ ε LMF.

Demonstração: Seja L a linguagem reconhecida pelo autômato A abaixo

A é AMF

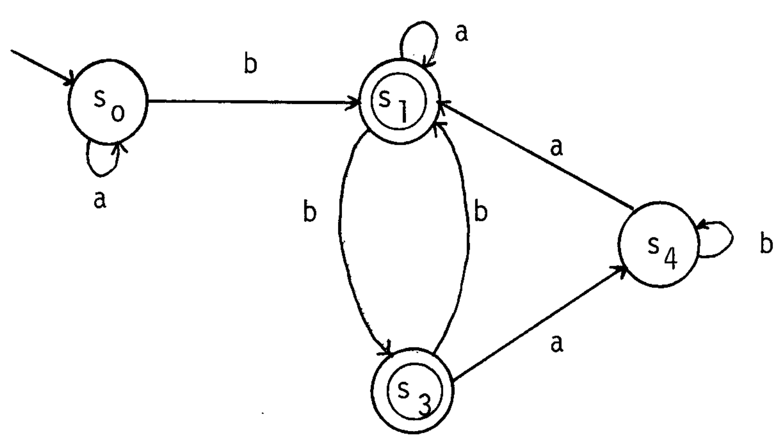


Seja $\phi: \Sigma \to \Delta^*$

$a \to 1$

$b \to 01$

A partir de A uma construção natural nos dá um autômato $\phi^{-1}(A)$ para $\phi^{-1}(L)$ temos $\phi^{-1}(A)$:



ϕ^{-1} (A) não é um AMF, os estados s_1 e s_3 são compatíveis

$f(s_1, b\ b) = S_1$ e $f(s_3, b\ b) = S_3$

$f(s_1, b\ b) \in F \leftrightarrow f(s_3, b\ b) \in F$

$f(s_1, b) \in F \leftrightarrow f(s_3, b) \in F$

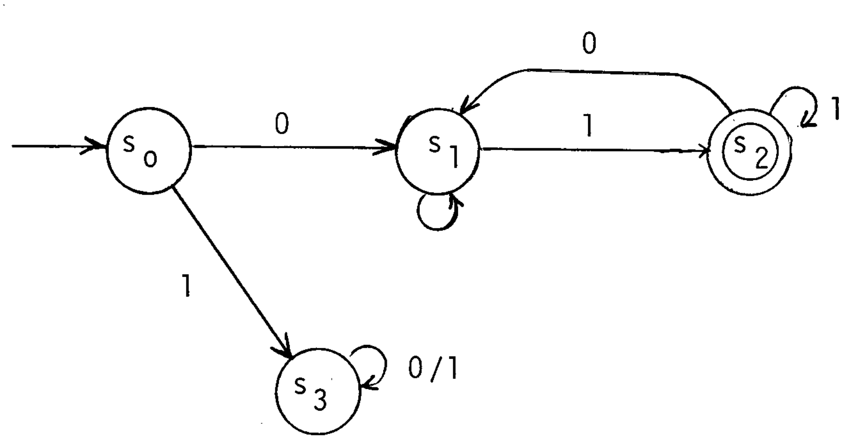
logo ϕ^{-1} (L) não é uma LMF.

VI.8.4. TEOREMA

$\overline{LMF}$ não é fechada sob imagem homomorfa inversa, isto é, $\phi^{-1}(L)$ para $L \notin LMF$ pode ser uma LMF.

Demonstração: Seja $L = 0\ \Sigma^*\ 1$ ($L \notin LMF$) e A um autômato minimal que reconhece L:

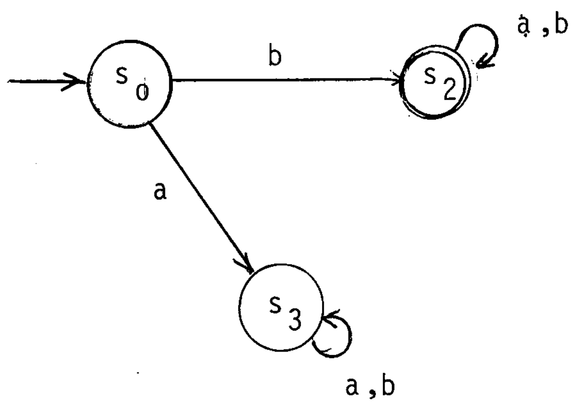
A:



ϕ: a $\rightarrow$ 1

$b \rightarrow 01$

temos: $\phi^{-1}(A)$, como antes



A não tem estados compatíveis, logo A é um AMF $\Longrightarrow$ $\phi^{-1}(L)$ é uma LMF.

CAPÍTULO VII

VII. CARACTERIZAÇÃO DAS LINGUAGENS DE MEMÓRIA FINITA

Uma caracterização das linguagens de memória finita, formulada baseada apenas em conceitos de linguagem, isto é, sem recorrer ao autômato e ao conceito de estados compatíveis pode ser feito utilizando-se a congruência de Nerode.

A relação congruência de Nerode da linguagem L, simbolizando $\perp_L$, é definida por:

$$x \perp_L y \quad \text{sse } \forall z \ \varepsilon \ \Sigma^*, \ xz \ \varepsilon \ L \leftrightarrow yz \ \varepsilon \ L$$

As classes de equivalência desta relação correspondem aos estados do autômato minimal que reconhece L.

L não é LMF se e somente se $\exists$ u, v, w tal que

a) $u \perp_L v$

b) $uw \perp_L u$ e $vw \perp_L v$

c) $\forall$ x pref (w), $u\,x \ \varepsilon \ L \leftrightarrow v\,x \ \varepsilon \ L$.

ou onde de outra forma L é LMF se e somente se $\forall$u, v com $u \perp_L v$ e $\forall$w se $u\,w \perp_L u$ e $vw \perp_L v$ então $\exists$ x pref (w) tal que $u\,x \ \varepsilon$ L e $vx \notin L$, ou $ux \notin L$ e $vx \ \varepsilon$ L.

Uma caracterização intrínseca mais simples seria desejável.

CAPÍTULO VIII

VIII. CONCLUSÃO

O conceito de memória finita é definido primeiramente com respeito a máquina sequencial.

Uma máquina sequencial reduzida é MMF se e somente se dois estados distintos não tenham circuitos E/S iguais. Com o conceito de circuitos definimos também as MD'S .

Baseado no conceito de estados compatíveis , isto é, $s \sim t \leftrightarrow \exists w \in \Sigma^+$ tal que $f(s, w) = s$, $f(t, w) = t$ e $\forall x$ pref (w) $f(s, x) \in F \leftrightarrow f(t, x) \in F$, definimos os AMF's.

A linguagem reconhecida por um autômato AMF é denominada Linguagem de Memória Finita. O mesmo ocorre para as LD's que são reconhecidas pelos AD's.

Examinamos os seguintes tipos de LMF's:

i) $\bigcup_{i=1}^{n} A_i B_i^*$ para A_i e B_i finitos e B_i distintos entre si.

ii) $L \Sigma^*$ $L \in$ LMF.

iii) $L \cup P$, $L-P$. $L \in$ LMF e P finito.

iv) $\Sigma^* A \Sigma^*$ A finito.

v) $A \Sigma^*$ A finito.

vi) $(A \cup B \Sigma^*)$ A e B finitos (definidas reversas).

Verificamos também que a classe das LMF's só é fechada para as operações de complemento e derivadas (VI.1 e VI.5), para as outras operações apresentamos os casos particulares do fechamento.

Como sugestões para futuros desenvolvimentos, parecem interessantes:

- característica intrínseca mais simples do que a do Capítulo VII.
- Sob que operações "corrigidas" (por exemplo concatenação marcada) as LMF's são fechadas.
- Caracterização à la Kleene. Análoga a expressões regulares.

B I B L I O G R A F I A

|1| Z. Kohavi - Switching and Finite Automata Theory - Mc Graw-Hill, New York (1970).

|2| M. Perles, M.O. Rabin, E. Shamir - The Theory of Definite Automata - IEEE Trans. Electron Computers, vol. EC-12, pp. 233-243 (1963).

|3| T. Booth - Sequential Machines and Automata Theory - John Wiley & Sons, New York (1967).

|4| J. Hopcroft, J. Ullman - Formal Languages and Their Relation to Automata - Addison-Wesley Publishing Company (1969).

|5| E.F. Moore - Sequential Machines - Selected Papers, Addison Wesley Publishing Company (1964).

|6| J.A. Brzozowski - Derivation of Regular Expressions, JACM, vol. 11, nº 4, pp. 481-494 (1964).

|7| P.A.S. Veloso - On Interconnections of Definite and Finite-Recall Machines - COPPE/UFRJ (1975).

|8| P.A.S. Veloso - Anotações de Aulas.