

Capítulo 2.

Autómatos Finitos

2.1. Aceitadores determinísticos (DFA)

2.2. Autómatos finitos não-determinísticos (NFA)

2.3. Equivalência entre os DFA e os NFA

2.4 Autómatos finitos transdutores (Mealy e Moore)

A partir de um alfabeto, podem-se definir muitas linguagens

As linguagens podem ser definidas por uma gramática

Dada uma cadeia de caracteres, como saber se pertence a uma dada linguagem ?

A resposta pode ser obtida por um autómato chamado **aceitador**: ele tem um estado “aceitar” ao qual é levado por qualquer cadeia (de uma dada linguagem), que se apresente à sua entrada. Por isso se chama aceitador (*accepter*).

Linguagens



Gramáticas G
Geram as
cadeias de L



Autómatos
Reconhecem as
cadeias de L

2.1. Aceitadores determinísticos (DFA)

Definição 2.1. DFA

Um aceitador determinístico (DFA- Deterministic Finite Acceptor) é definido pelo quinteto

$$M = (Q, \Sigma, \delta, q_0, F)$$

Q : conjunto finito de estados internos

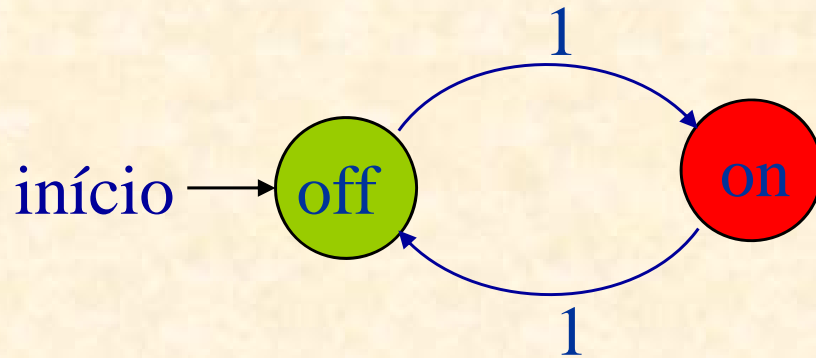
Σ : alfabeto de entrada (conjunto finito de caracteres)

$\delta: Q \times \Sigma \rightarrow Q$ é a função **total** chamada função de transição

$q_0 \in Q$ é o estado inicial

$F \subseteq Q$ é o conjunto de estados finais

Exemplo 1



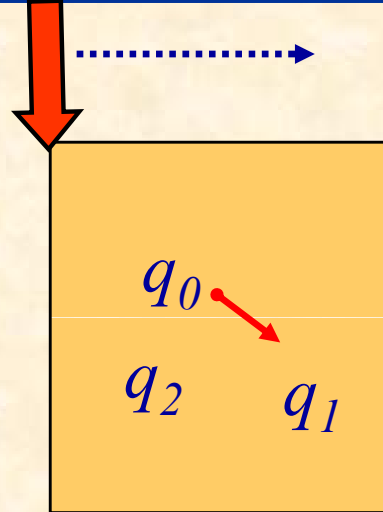
Q , conjunto de estados internos: $\{on, off\}$

Σ , alfabeto de entrada: $\{1\}$

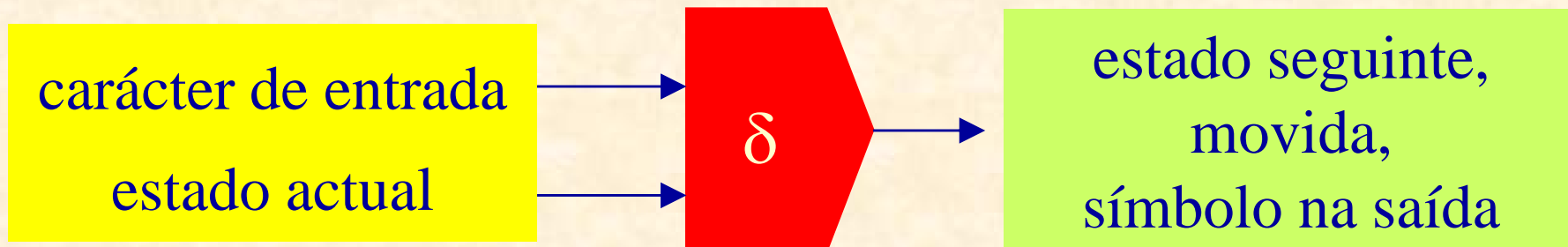
δ , função de transição: $off \times 1 \rightarrow on; on \times 1 \rightarrow off$

q_0 , é o estado inicial: off

F , o estado final: $\{on\}$



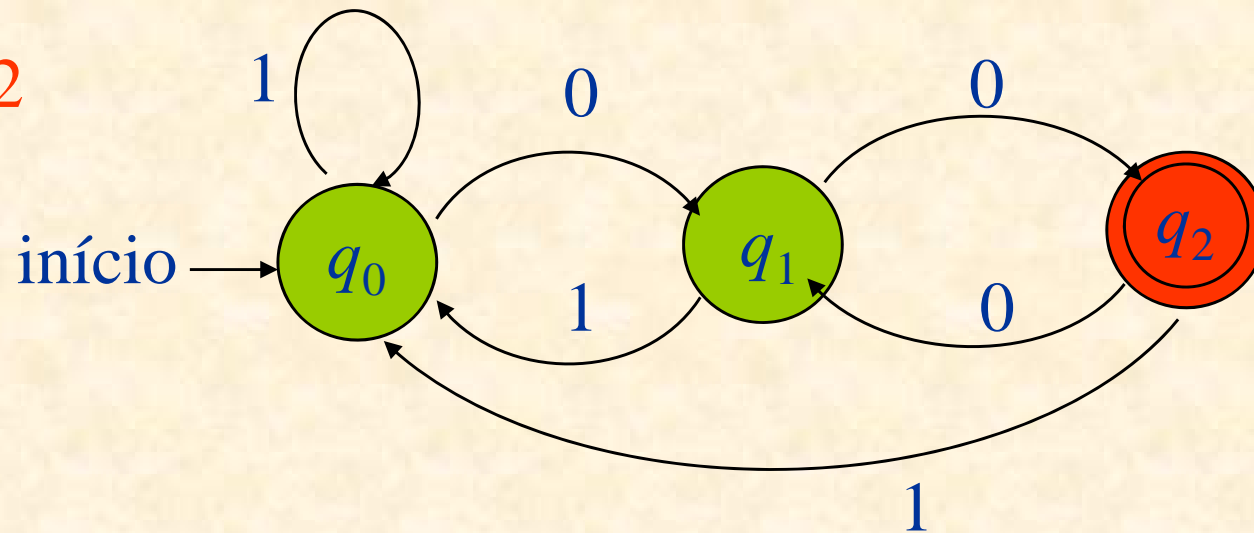
Não tem
dispositivo de
memória nem
cadeia de saída !!!



2.1.1.Representação de um autômato por um grafo



Exemplo 2

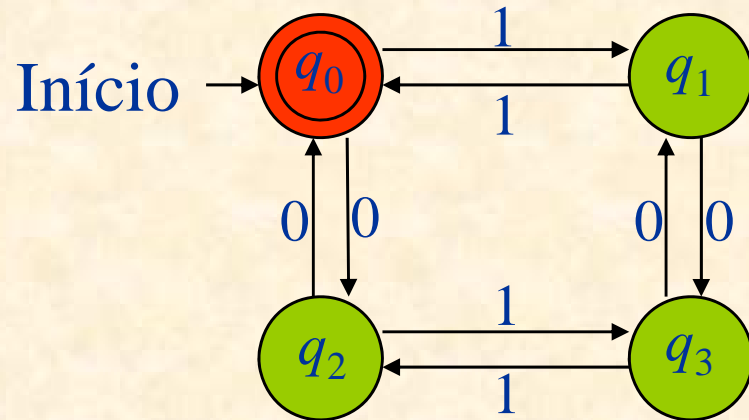


$$M = (Q, \Sigma, \delta, q_0, F)$$

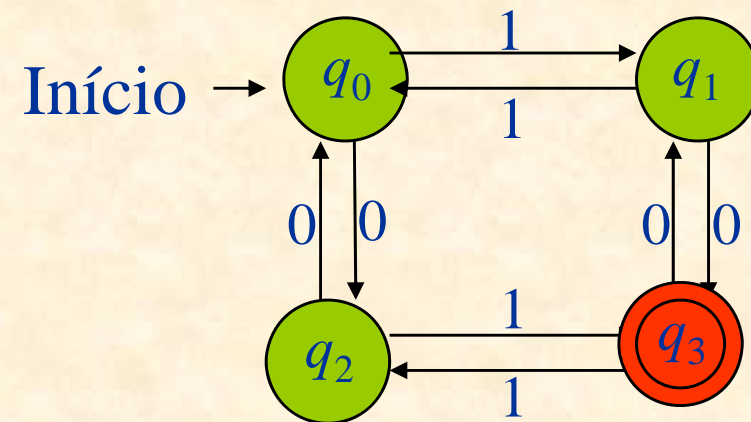
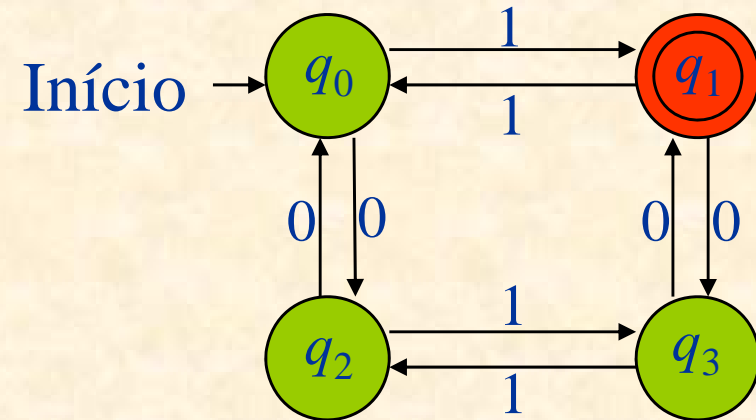
$Q = \{q_0, q_1, q_2\}$, $\Sigma = \{0, 1\}$, $F = \{q_2\}$ e δ é definida por

Estados	Entradas	
	0	1
q_0	q_1	q_0
q_1	q_2	q_0
q_2	q_1	q_0

Exemplo 3



Estados	Entradas	
	0	1
q_0	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2



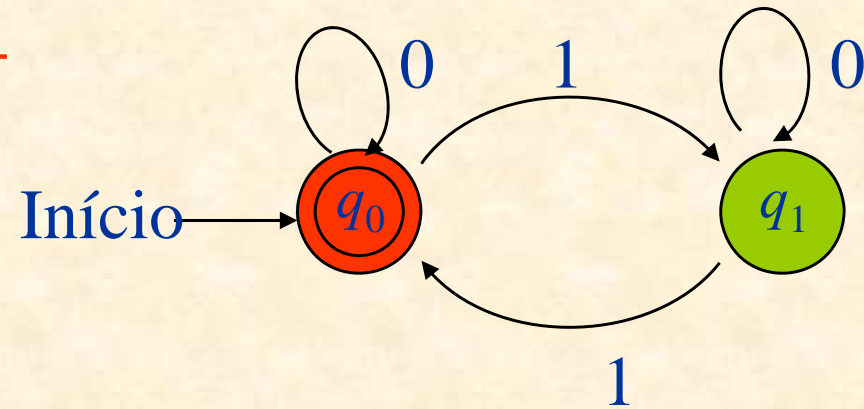
A linguagem de um DFA M

a linguagem aceite (ou reconhecida) por M é o

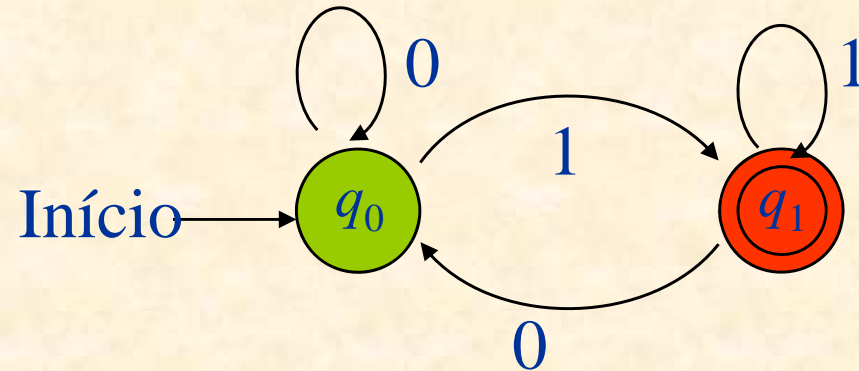
conjunto de todas as cadeias que,
começando no estado inicial, alcançam um
dos estados finais depois de toda a cadeia
ter sido lida.

Qual a linguagem do DFA do exemplo 3 ?

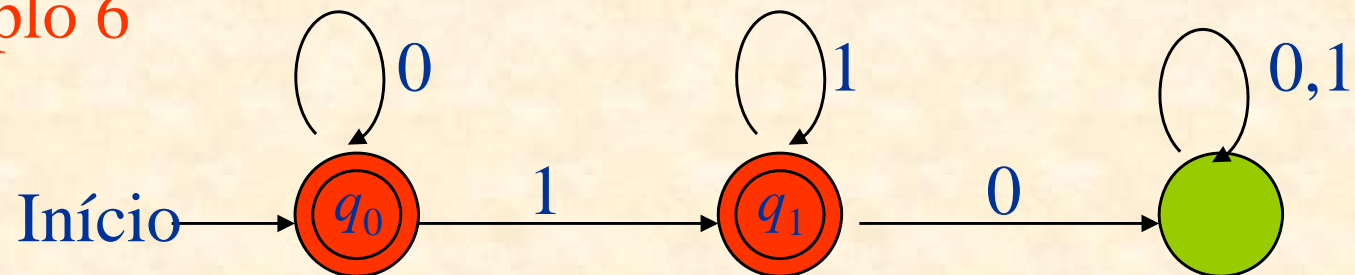
Exemplo 4



Exemplo 5



Exemplo 6



Exemplo 7

$$M = (\{q_0, q_1, q_2\}, \{a, b\}, \delta, q_0, \{q_2\})$$

$$\delta: Q \times \Sigma \rightarrow Q$$

Função de transição:

$$\delta(q_0, a) = q_1$$

$$\delta(q_0, b) = q_2$$

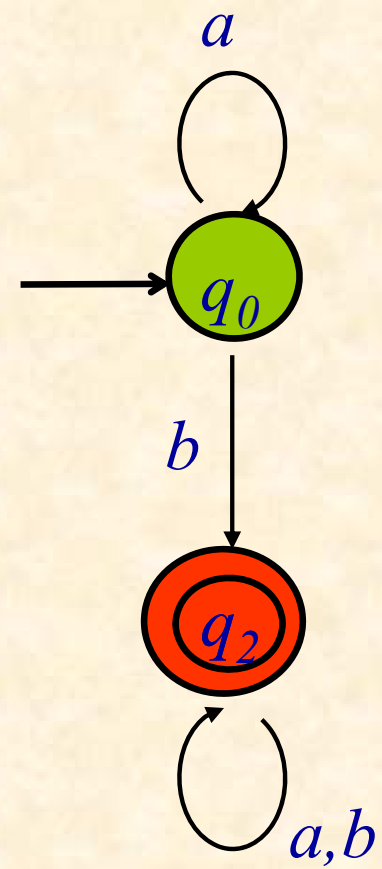
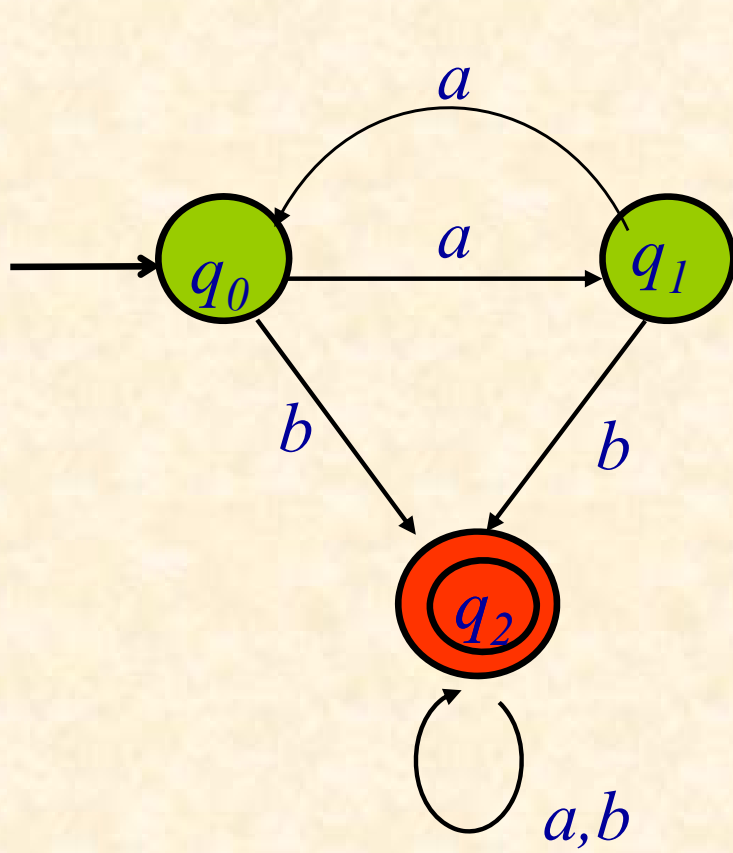
$$\delta(q_1, a) = q_0$$

$$\delta(q_1, b) = q_2$$

$$\delta(q_2, a) = q_2$$

$$\delta(q_2, b) = q_2$$

	a	b
q_0	q_1	q_2
q_1	q_0	q_2
q_2	q_2	q_2



Função de transição estendida δ^* :

$$\delta^* : Q \times \Sigma^* \rightarrow Q$$

- O seu segundo argumento é uma cadeia em vez de um carácter,
- o seu valor de saída dá o estado do autómato depois de ler a cadeia toda,
- é uma forma compacta de descrever um conjunto de transições resultantes da leitura de uma cadeia de entrada.

Por exemplo

$$\delta(q_0, a) = q_1 \text{ e } \delta(q_1, b) = q_2 \text{ então } \delta^*(q_0, ab) = q_2$$

Definição recursiva da função de transição estendida

1. $\delta^*(q, \lambda) = q$
2. $\delta^*(q, wa) = \delta(\delta^*(q, w), a)$

para todo o $q \in Q$, $w \in \Sigma^*$, $a \in \Sigma$.

Para o exemplo anterior teremos :

$$\delta^*(q_0, ab) = \delta(\delta^*(q_0, a), b)$$

$$\delta^*(q_0, a) = \delta^*(q_0, \lambda a) = \delta(\delta^*(q_0, \lambda), a) = \delta(q_0, a) = q_1$$

$$\delta^*(q_0, ab) = \delta(q_1, b) = q_2$$

Linguagens e DFA's

Definição 2.2. Linguagem aceite por um DFA

A linguagem aceite por um DFA $M=(Q, \Sigma, \delta, q_0, F)$ é o conjunto de todas as cadeias em Σ^* aceites por M , i.e.,

$$L(M) = \{ w \in \Sigma^* : \delta^* (q_0, w) \in F \}$$

- As funções de transição δ e δ^* são **funções totais**.
- Em cada passo define-se um e um só movimento, e por isso o autómato chama-se **determinístico**.
- O autómato processa todas as w em Σ^* , aceitando-as ou não.

Complemento da linguagem de um DFA

Quando o DFA não aceita uma cadeia, pára num estado não final, de modo que

$$\text{Compl}(L(M)) = \{ w \in \Sigma^* : \delta^*(q_0, w) \notin F \}$$

Como desenhar o DFA que aceita o $\text{Compl}(L(M))$??

2.1.2. Linguagens regulares

Uma linguagem diz-se **regular** se e só se existir um DFA M que a aceite, i. e.

$$L = L(M)$$

- A família das linguagens regulares é composta por todas as linguagens que são aceites por um qualquer DFA.
- Uma forma de demonstrar que uma linguagem é regular é encontrar um DFA que a aceite.
- O conjunto das linguagens regulares constitui uma **família**.

Construir um DFA que aceite a linguagem L em $\Sigma = \{0, 1\}$ tal que

Exemplo 8

L contém apenas “010” e “1”.

Exemplo 9

L é o conjunto de todas as cadeias que terminam com “00”

Exemplo 10

L é o conjunto de todas as cadeias sem “1”s consecutivos nem “0”s consecutivos.

2.2. Aceitadores finitos não-determinísticos (NFA)

A partir de um estado são possíveis **zero, uma ou mais** transições **com o mesmo símbolo** do alfabeto.

Uma entrada é aceite se houver para ela **um** caminho que leva a um estado final (aceitador).

Definição 2.4. NFA

Um aceitador não-determinístico (**NFA – Nondeterministic Finite Acceptor**) é definido pelo quinteto

$$M = (Q, \Sigma, \delta, q_0, F)$$

em que a função de transição δ é definida por

$$\delta: Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$$

3 diferenças em relação ao DFA:

- 1) o contradomínio de δ é a *potência de conjuntos de Q* , e não Q , precisamente porque o resultado de δ pode ser um subconjunto de Q .
- 2) λ pode ser argumento de δ , ou seja, pode dar-se **uma transição sem consumir um símbolo de entrada** (o mecanismo de leitura pode ficar **parado** em alguns movimentos).
- 3) O conjunto $\delta(q_i, a)$ pode ser vazio, significando que **não há qualquer transição** nesta situação.

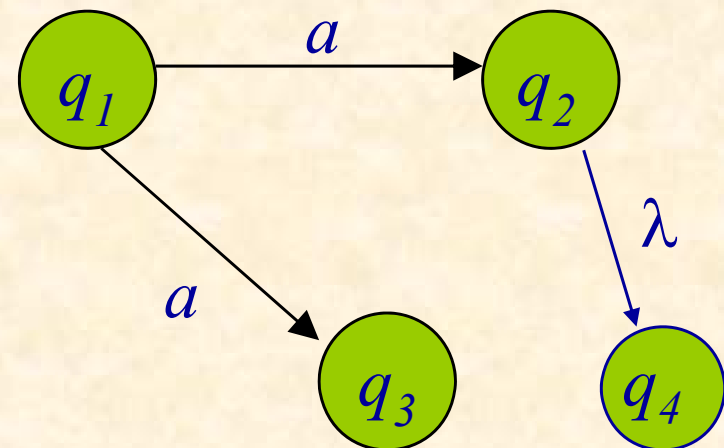
2.2.1.Representação de um NFA por um grafo

OS NFA podem representar-se por grafos tal como o DFA.

Diferenças:

- i) Do mesmo vértice podem partir diversas arestas com a mesma etiqueta

$$\delta(q_1, a) = \{ q_2, q_3 \}$$



- ii) Algumas arestas podem ser etiquetadas por λ
- iii) Nem todas as transições de um vértice são definidas

Uma cadeia é aceite pelo NFA

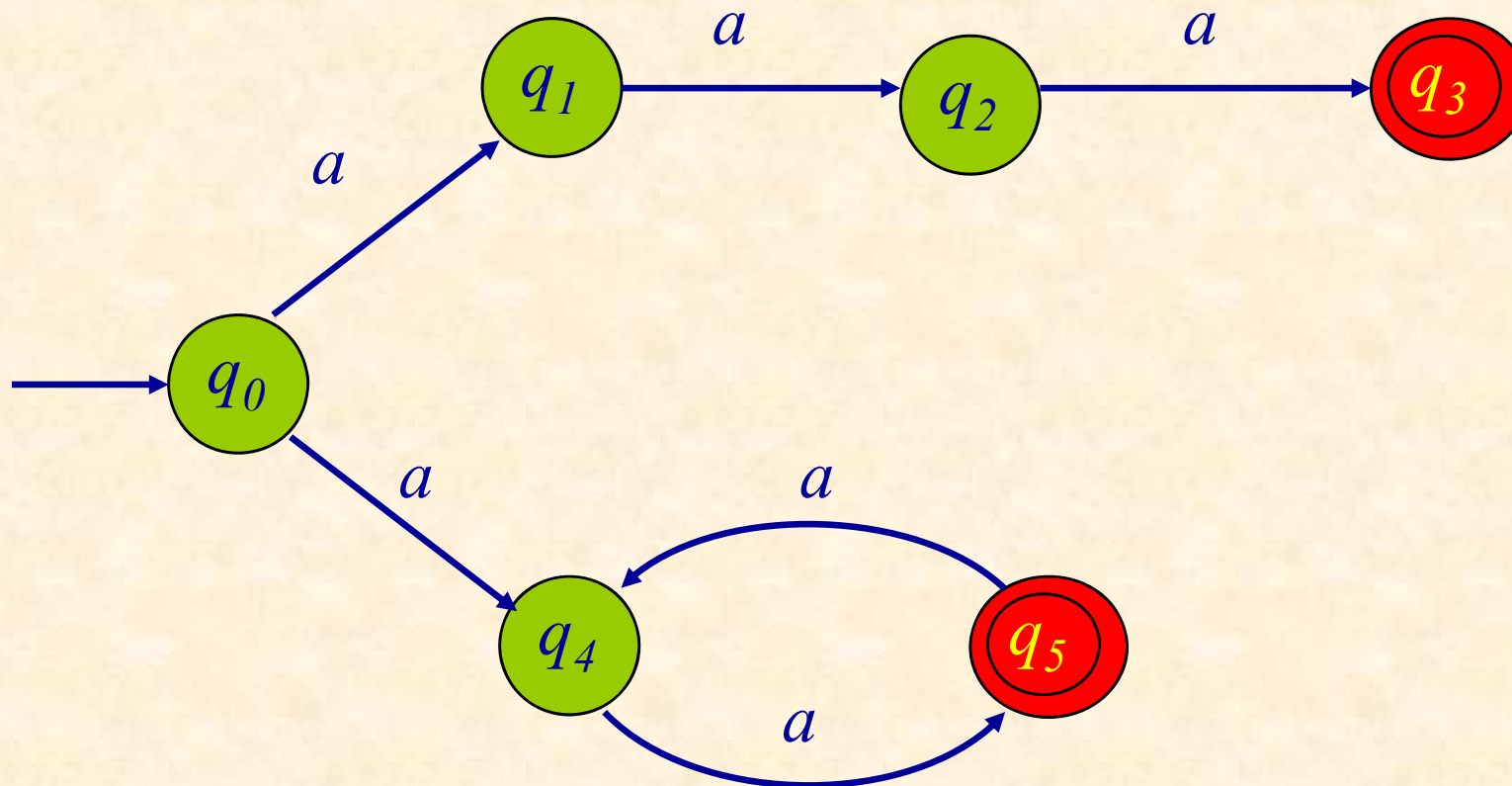
se houver alguma sequência de movimentos possíveis que coloquem o autómato num estado final no fim da cadeia.

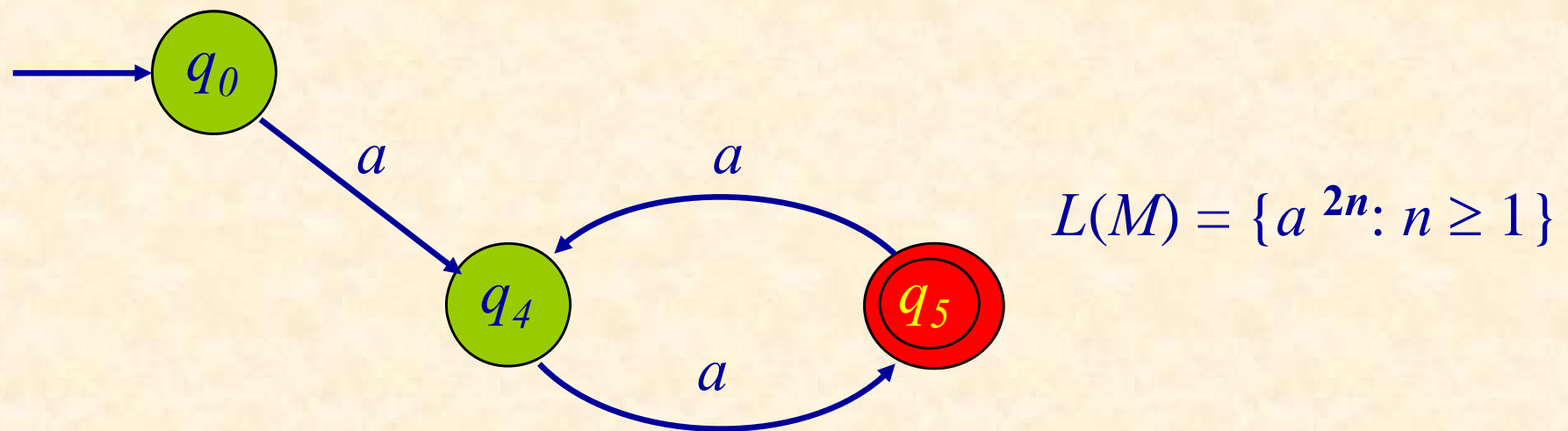
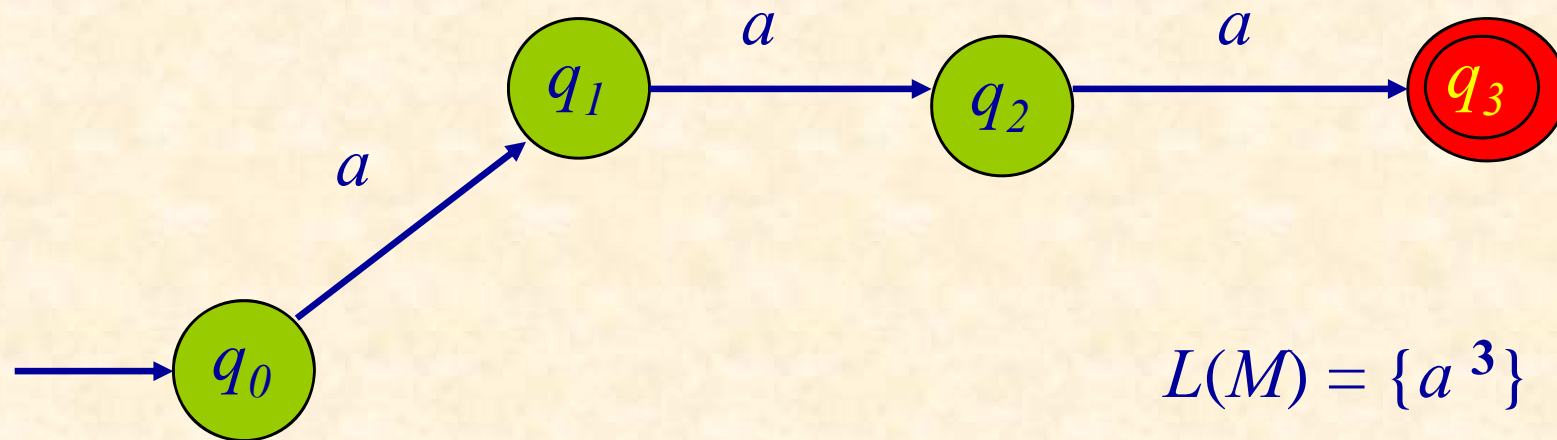
Caso contrário é rejeitada.

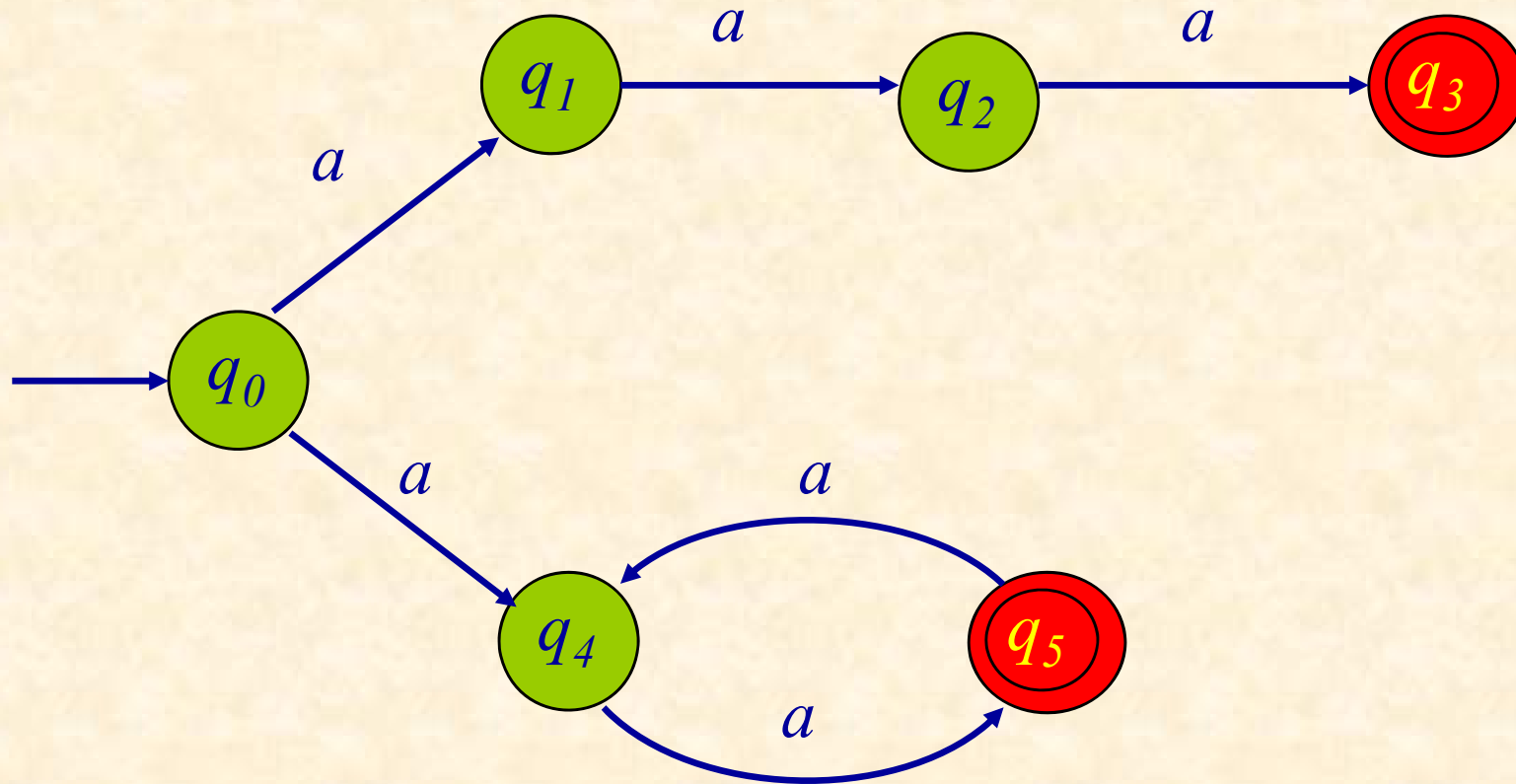
O não-determinismo permite escolher entre diversos caminhos possíveis para uma cadeia de entrada.

Exemplo 11

Que linguagem é aceite pelo NFA ?

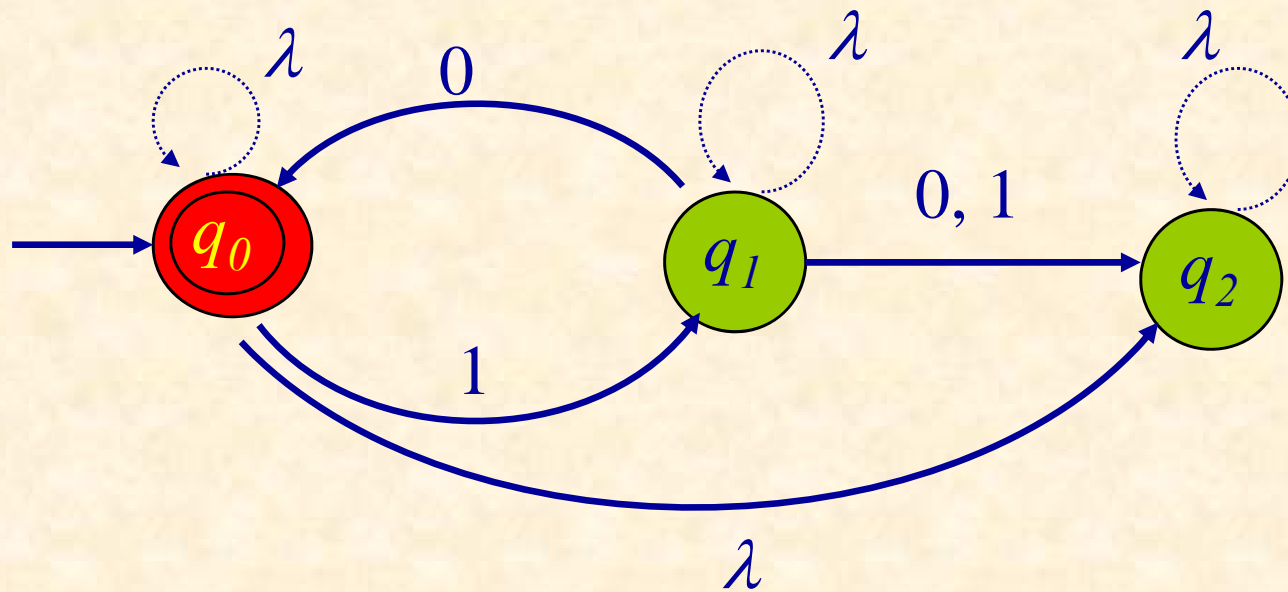






$$L(M) = \{a^3\} \cup \{a^{2n} : n \geq 1\}$$

Exemplo 12



$$\begin{aligned}\delta(q_0, 1) &= \{q_1\} \\ \delta(q_0, 0) &= \emptyset \\ \delta(q_1, 0) &= \{q_0, q_2\} \\ \delta(q_1, 1) &= \{q_2\} \\ \delta(q_0, \lambda) &= \{q_0, q_2\} \\ \delta(q_2, 0) &= \emptyset \\ \delta(q_2, 1) &= \emptyset\end{aligned}$$

As 3 categorias de não determinismo:

- de q_1 partem duas arestas etiquetadas por 0
- de q_0 parte uma aresta etiquetada λ
- de q_2 não é definida qualquer transição, $\delta(q_2, 0) = \emptyset$
- existe sempre a transição $\delta(q_i, \lambda) = \{q_i\}$, para todo o i

Função de transição estendida δ^* em NFA

Define-se de modo análogo ao caso do DFA

$$\delta^*(q_i, w) = Q_j$$

sendo Q_j o conjunto de todos os estados em que o autômato se pode encontrar, partindo de q_i e tendo lido a cadeia w .

Definição 2.5.

A função de transição estendida (NFA) δ^* define-se de modo que $\delta^*(q_i, w)$ contém q_j se e só se existir um caminho no grafo de transição desde q_i até q_j com etiqueta w , para **todos** os $q_i, q_j \in Q$ e todas as $w \in \Sigma^*$.

Linguagens e NFA's

Definição 2.2. Linguagem aceite por um NFA

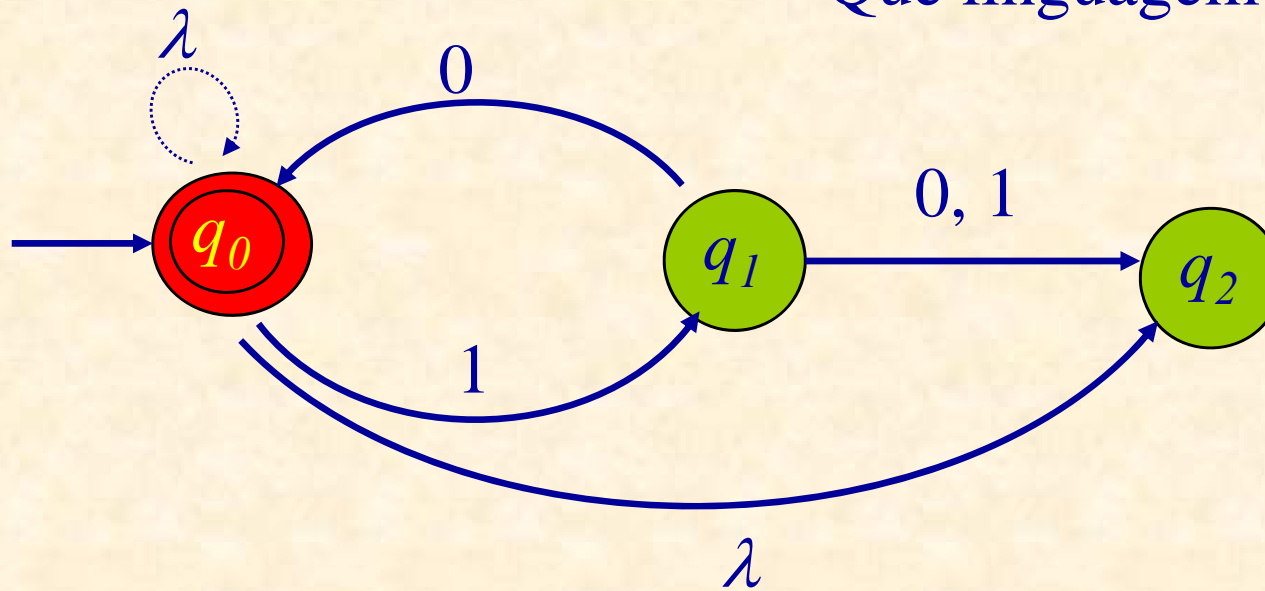
A linguagem L aceite por um NFA $M = (Q, \Sigma, \delta, q_0, F)$ é o conjunto de todas as cadeias em Σ^* aceites por M , i.e.,

$$L(M) = \{ w \in \Sigma^* : \delta^*(q_0, w) \cap F \neq \emptyset \}$$

Uma cadeia w pode levar a vários estados, depois de lida; se **pelo menos um** deles é final, a cadeia é aceite.

Exemplo 13

Que linguagem aceita ?



Aceita :

λ
10
1010
101010

Não aceita :

1
11
1011
1101

$$L(M) = \{ (10)^n : n \geq 0 \}$$

Porquê o não determinismo?

- servem para modelizar algoritmos em que tem que se fazer escolhas (search-and-backtracking, por exemplo)
- serve para definir linguagens compostas por conjuntos bastante diferentes (como no exemplo que aceita a linguagem $\{a^3\} \cup \{a^{2n} : n \geq 0\}$).
- o não-determinismo é um mecanismo apropriado para descrever de forma simples e concisa linguagens complicadas. Na definição de uma gramática existe um elemento de não-determinismo, como em
$$S \rightarrow aSb \mid \lambda$$
em qualquer ponto se pode escolher a primeira ou a segunda produção; podemos assim especificar um grande número de cadeias usando apenas duas regras.
- há uma razão de ordem técnica: alguns resultados são mais facilmente estabelecidos para NFA do que para DFA.
- Como veremos não há nenhuma diferença essencial entre NFA's e DFA's. Por conseguinte o facto de se permitir o não determinismo simplifica por vezes os argumentos formais sem afectar a generalidade da conclusão.

2.3. Equivalência entre DFA's e NFA's

Definição 2.7. Autómatos (aceitadores) equivalentes

Dois aceitadores M_1 e M_2 são equivalentes se

$$L(M_1) = L(M_2)$$

isto é, se ambos aceitam a mesma linguagem.

Existem geralmente muitos aceitadores para uma dada linguagem, e por isso **qualquer DFA ou NFA tem muitos aceitadores equivalentes.**

- os DFA podem-se considerar casos especiais (restritos) de NFA.
- se uma linguagem é aceite por um DFA então existe um NFA que também a aceita.
- se uma linguagem é aceite por um NFA, existirá um DFA que a aceite ?

Será que o não-determinismo eliminou esta possibilidade ?

De facto não eliminou !

SIM !!!



Determinação do DFA equivalente a um NFA

Depois de um NFA ler uma cadeia w , não se sabe precisamente em que estado está, mas sabe-se apenas que está num estado entre um conjunto Q_w de estados possíveis

$$Q_w = \{q_i, q_j, \dots, q_k\}.$$

Depois de um DFA ler a mesma cadeia tem que estar num estado bem definido, q_w .

Como encontrar uma correspondência entre estas duas situações ?

Truque:

- cria-se um (super) estado no DFA equivalente a Q_w , isto é, etiqueta-se, no DFA, um estado por Q_w .

Mas resultará isso num autómato finito ?

Qual o número máximo de estados do DFA que se podem obter por este processo ?

Se no NFA existem no total Q estados, o número máximo de estados que se podem obter no DFA é igual à potência de conjuntos de Q , isto é $2^{|Q|}$, e portanto finito.

Exemplo 14

NFA

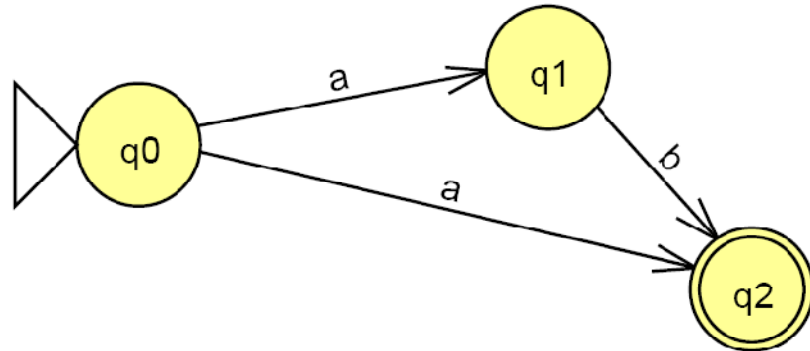


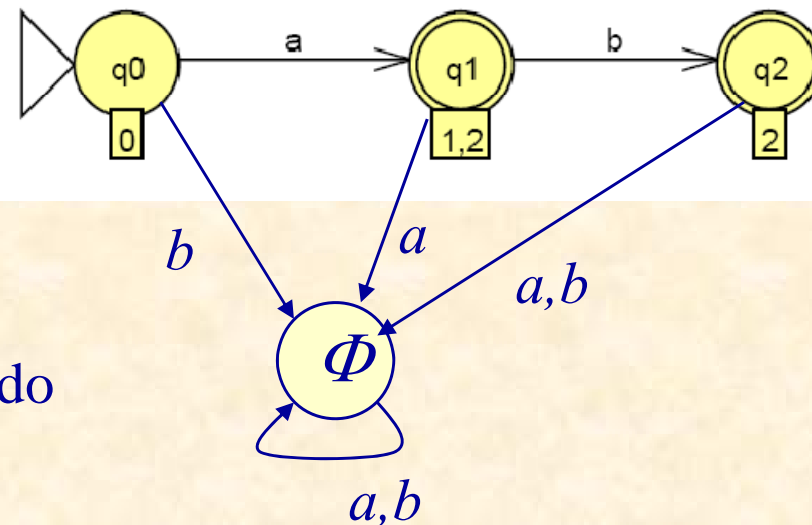
Tabela de transições

	a	b
q_0	q_1q_2	Φ
q_1q_2	Φ	q_2
Φ	Φ	Φ
q_2	Φ	Φ

Vários estados (q_1 , q_2) no NFA resultam um super-estado (q_{12}) no DFA

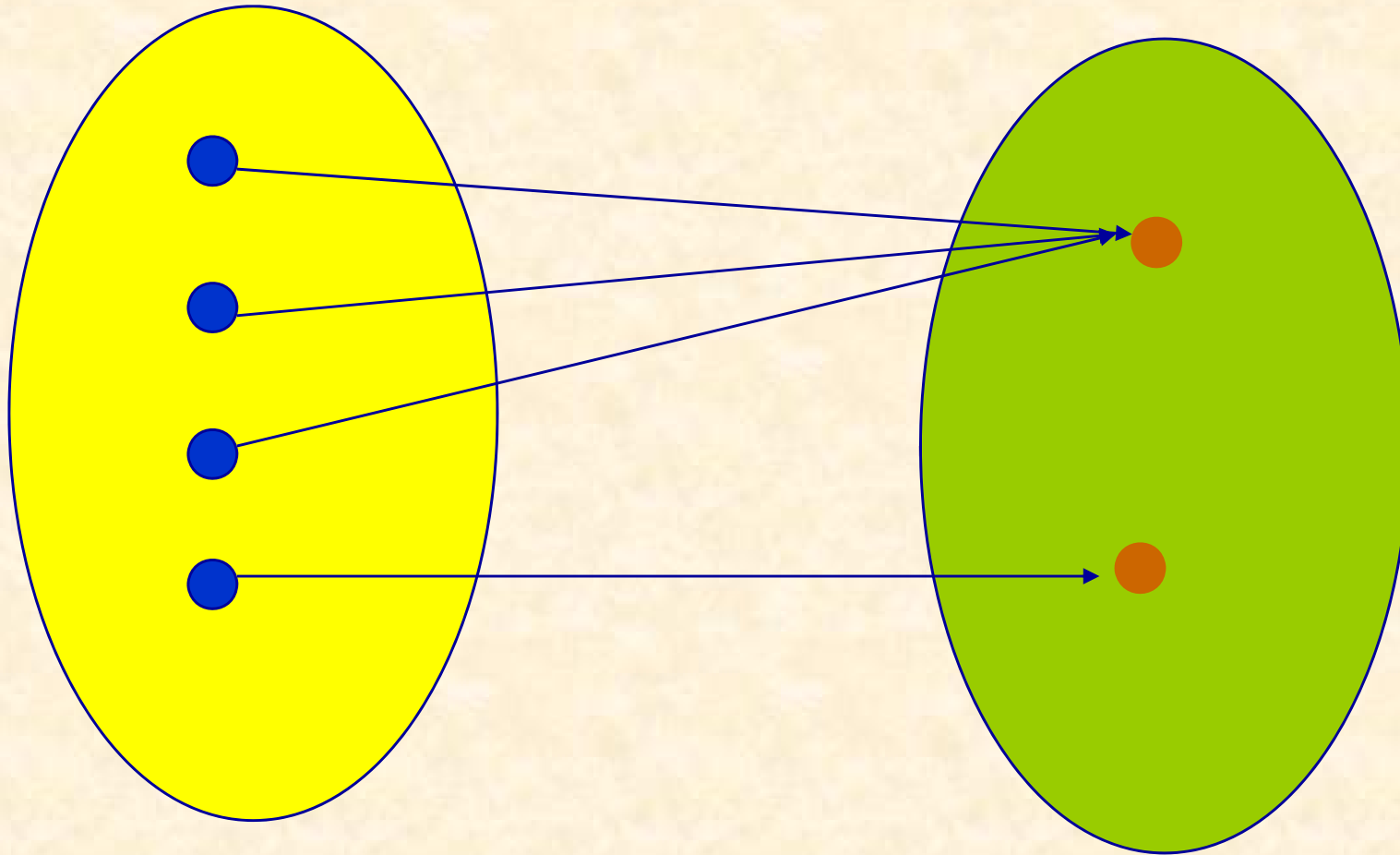
Nota: o JFLAP não introduz o estado vazio, Φ

DFA



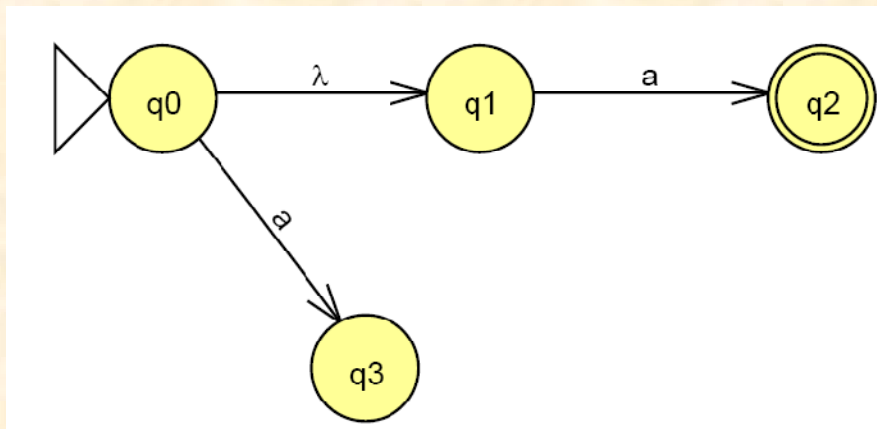
Q
NFA

Q
DFA



Exemplo 15

NFA

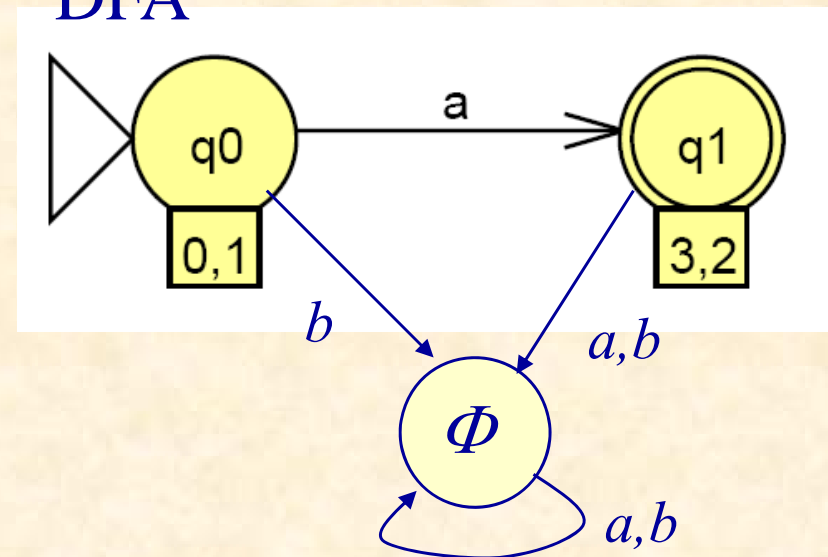


O estado inicial do DFA é composto pelo estado inicial do NFA e por todos os alcançáveis a partir deste com a transição λ

Tabela de transições

	a	b
q_0	q_2q_3	Φ
q_2q_3	Φ	Φ
Φ	Φ	Φ

DFA



Exemplo 16

Construir um NFA que aceite a linguagem L composta pelas cadeias em $\Sigma = \{0,1\}$ que contenham

três 0's seguidos

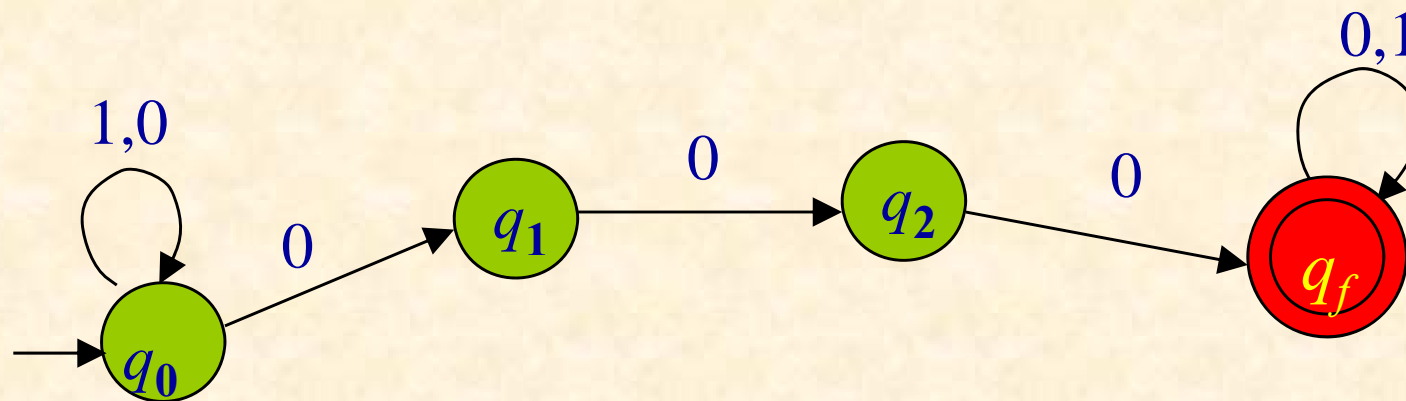
ou

três 1's seguidos ,

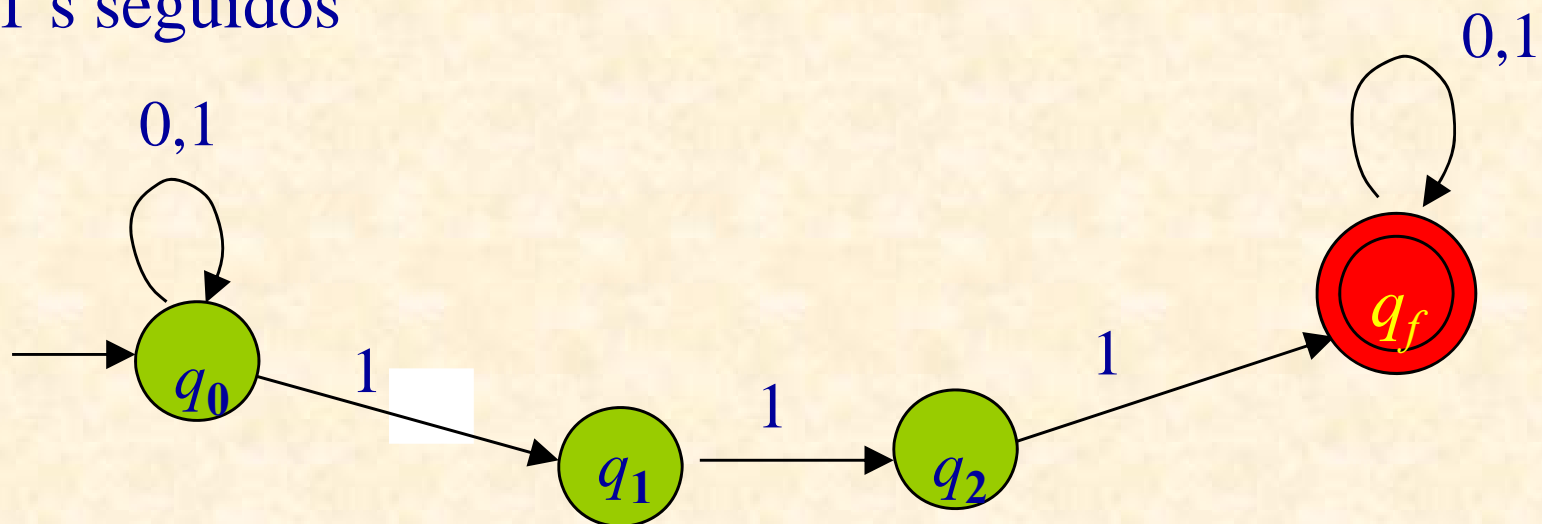
como por exemplo

000, 111, 10100010, 00111101000

três 0's seguidos

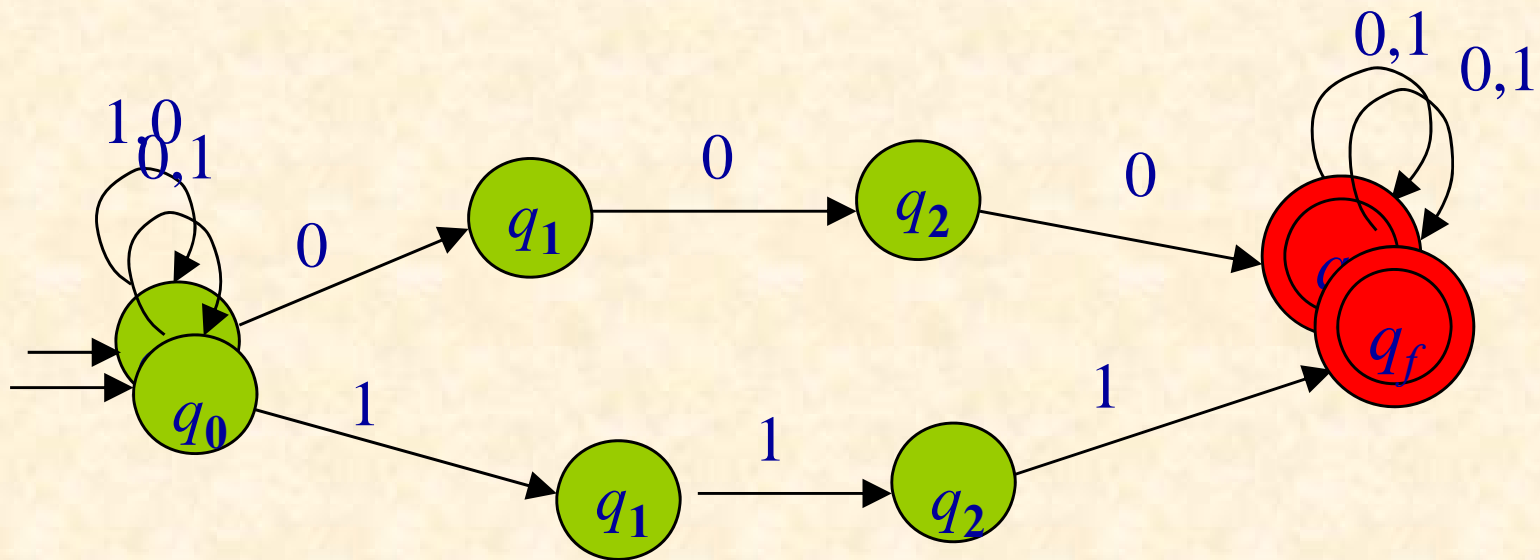


três 1's seguidos

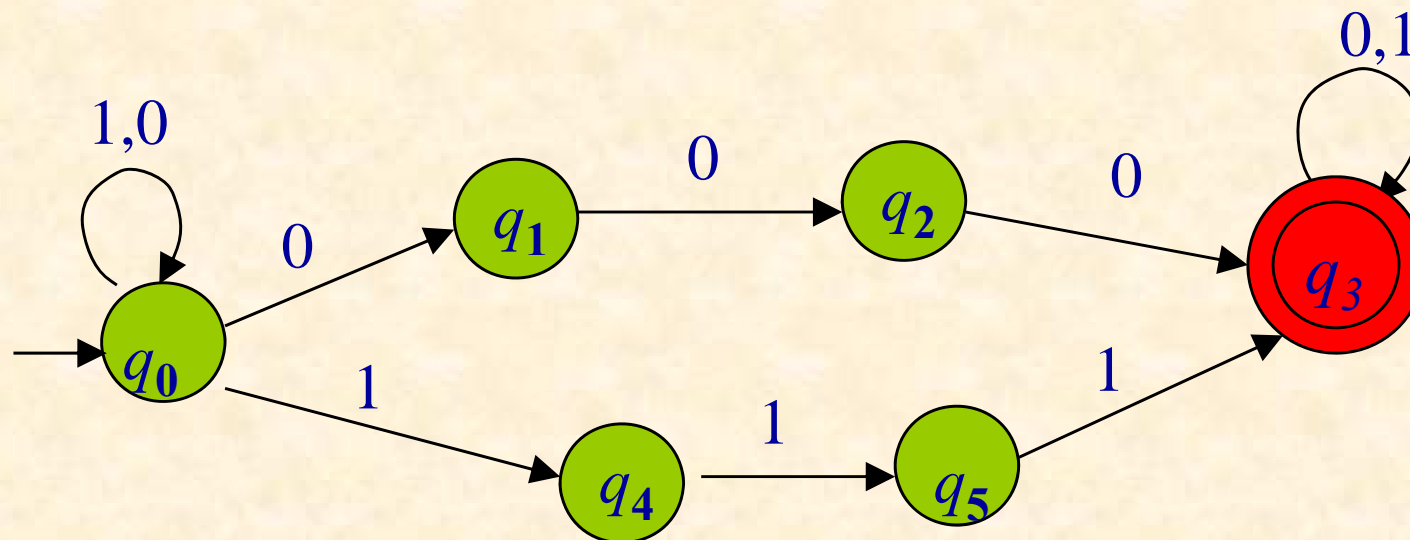


três 0's seguidos **ou** três 1's seguidos

Juntam-se os dois autómatos




Finalmente ...



Experimentar construir um DFA para o mesmo problema ...


Vamos agora procurar um DFA equivalente.

Para isso construa-se a tabela de transições:



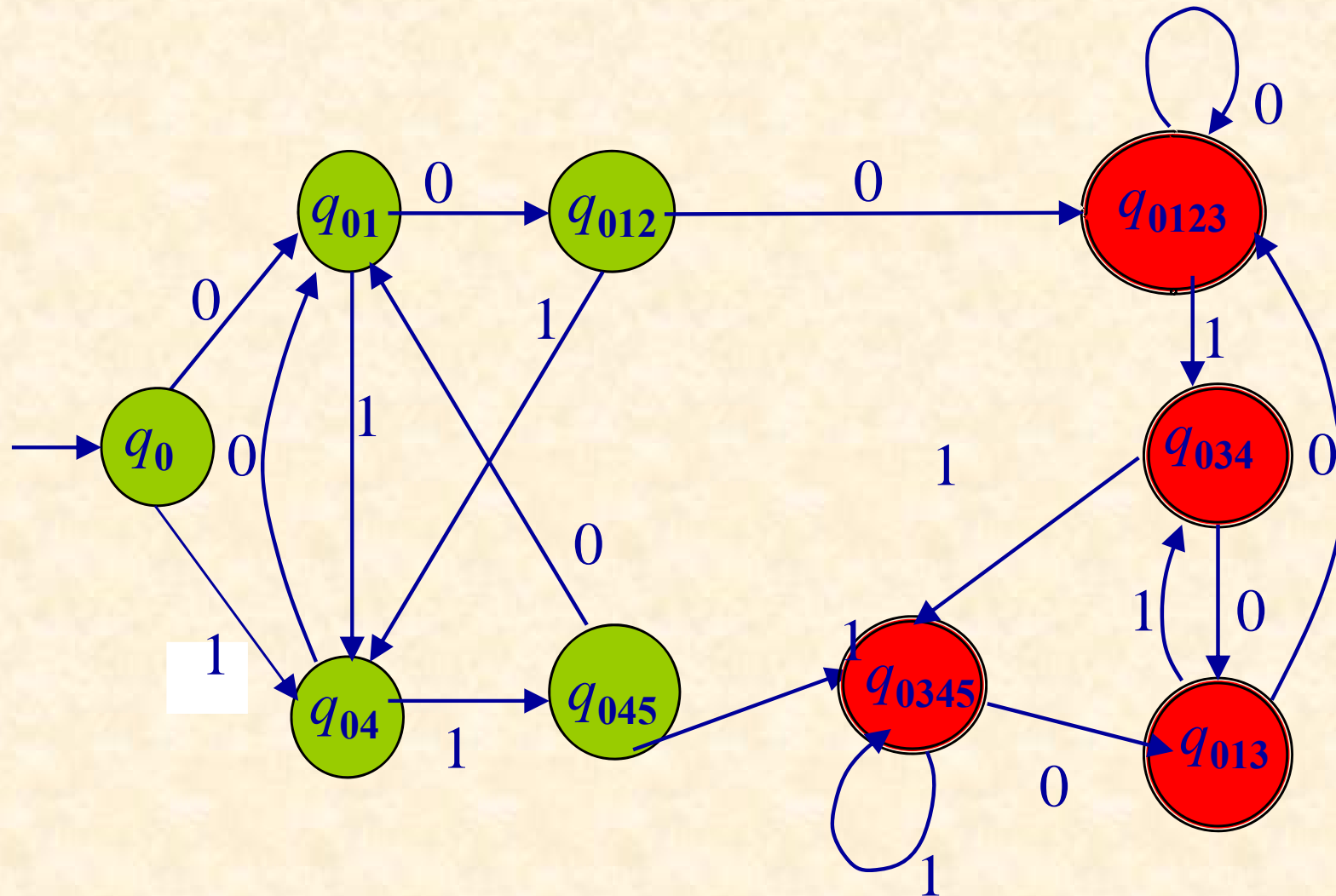
δ_N	0	1
q_0	q_0, q_1	q_0, q_4
q_1	q_2	-
q_2	q_3	-
q_3	q_3	q_3
q_4	-	q_5
q_5	-	q_3

Para as transições no DFA, etiquetam-se os seus estados com índices agrupando os índices dos estados do NFA alcançáveis pelas mesmas transições :



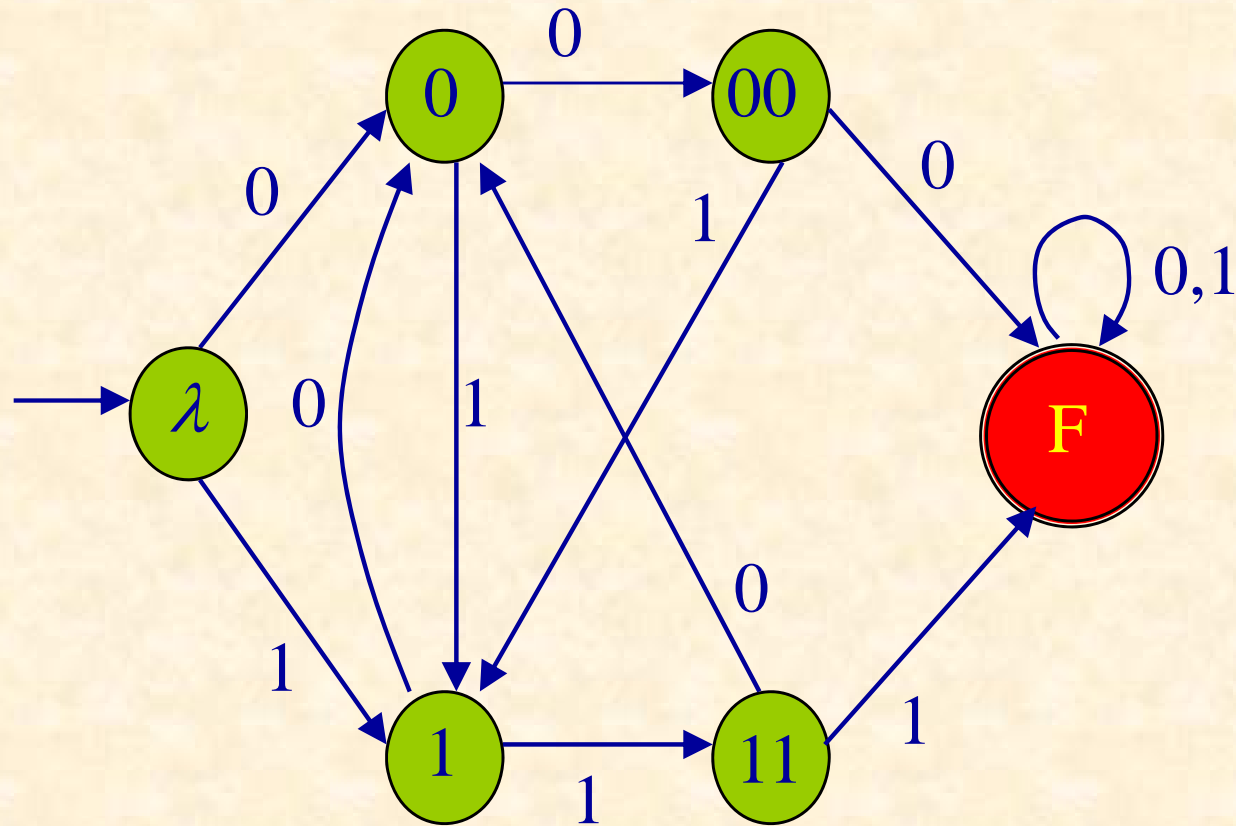
δ_D	0	1
q_0	q_{01}	q_{04}
q_{01}	q_{012}	q_{04}
q_{04}	q_{01}	q_{045}
q_{012}	q_{0123}	q_{04}
q_{045}	q_{01}	q_{0345}
q_{0123}	q_{0123}	q_{034}
q_{0345}	q_{013}	q_{0345}
q_{013}	q_{0123}	q_{034}
q_{034}	q_{013}	q_{0345}

O DFA que aceita as cadeias contendo 000 e/ou 111 é:



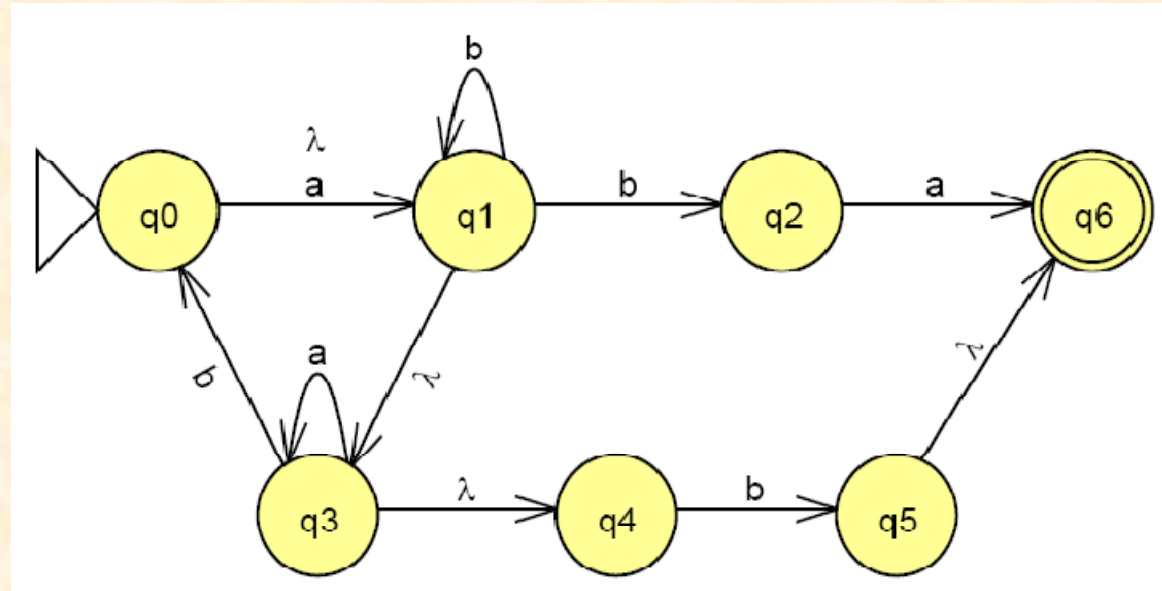
Todos os que contenham q_3 são aceitadores

adoptando etiquetas mais intuitivas e simplificando



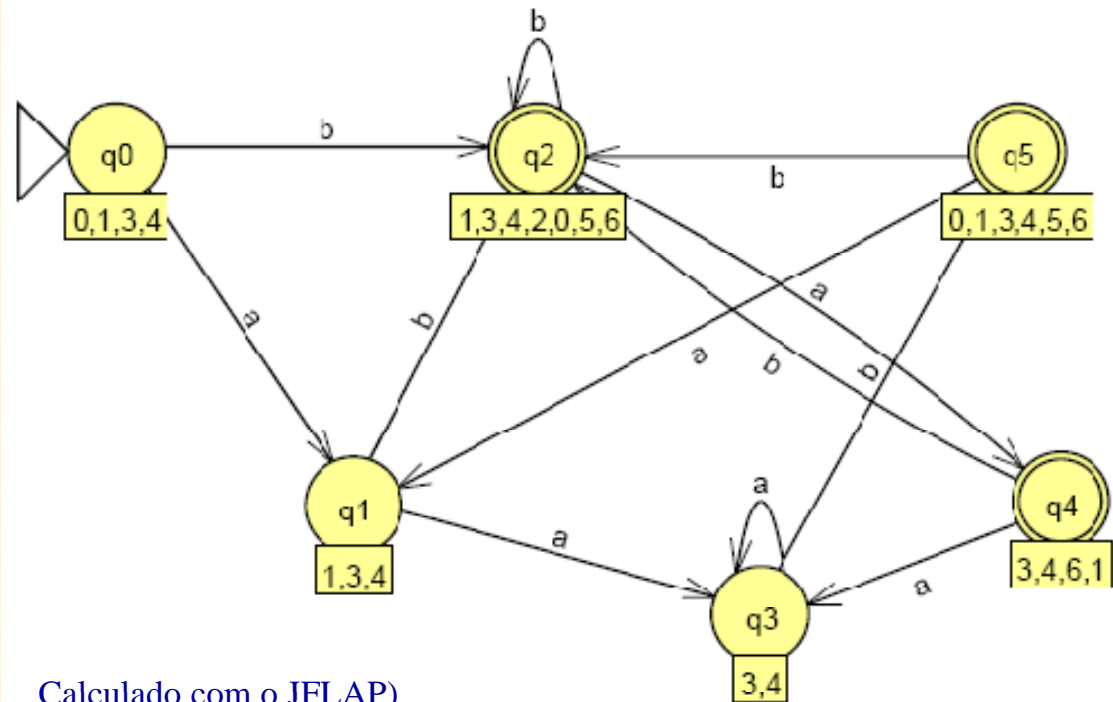
Exemplo 17

NFA



Construa a
tabela de
transições

DFA



Calculado com o JFLAP)

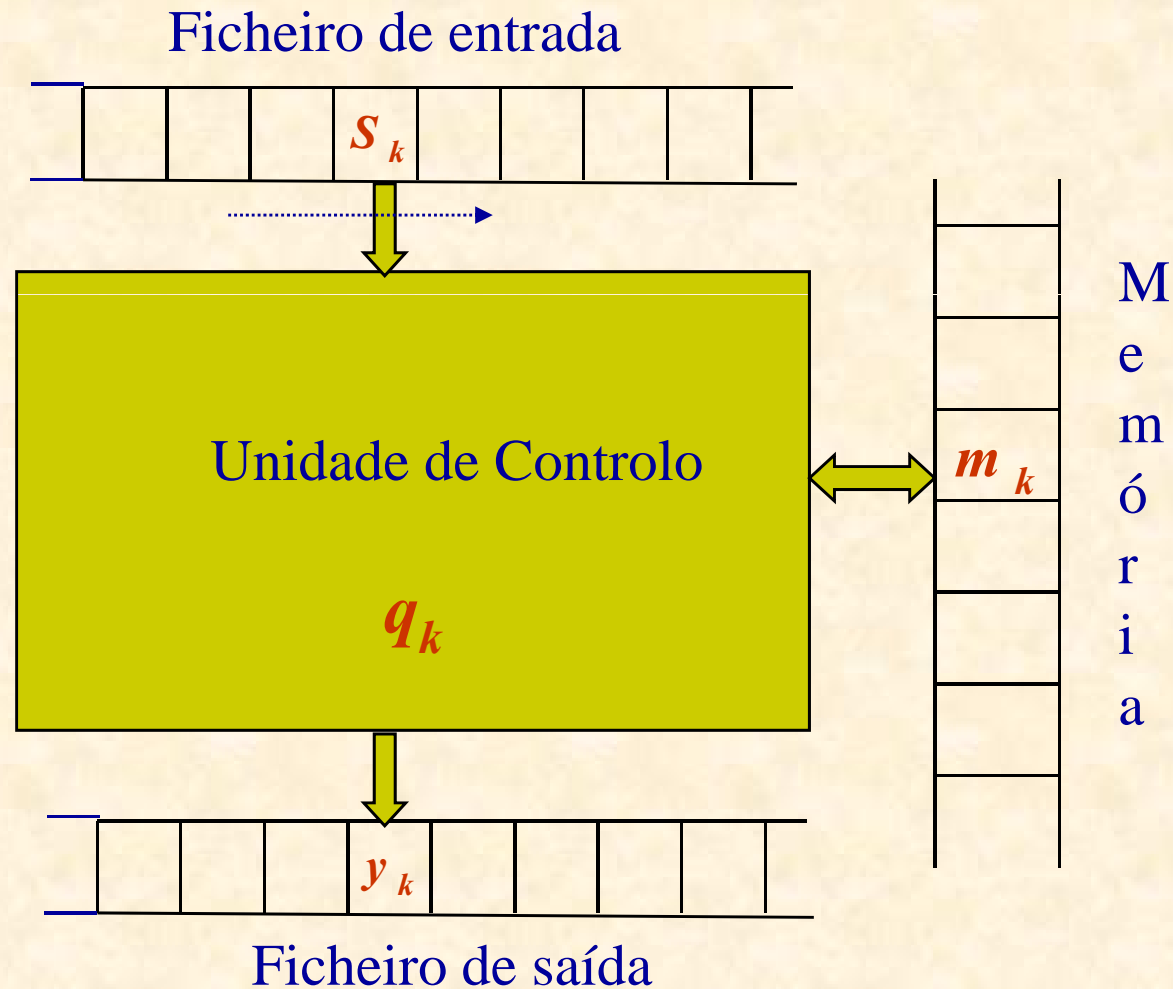
2.4. Redução do número de estados em DFA's

Ver Linz (62-65) ou Hopcroft et al.(159-164)

(Implementado no JFLAP)

2.5. Autómatos finitos transdutores

- Têm entrada, estado e saída



2.5.1. Máquinas de Mealy ¹

A saída depende do estado actual e da entrada actual

$$y_k = f(s_k, q_k)$$

Definição: Uma Máquina de Mealy é definida pelo sexteto

$$M = (Q, \Sigma, \Lambda, \delta, \gamma, q_0)$$

Q : conjunto finito de estados internos

Σ : alfabeto de entrada (conjunto finito de caracteres)

Λ : alfabeto de saída (conjunto finito de caracteres)

$\delta: Q \times \Sigma \rightarrow Q$ é a função de transição de estado

$\gamma: Q \times \Sigma \rightarrow \Lambda$ é a função de saída

$q_0 \in Q$ é o estado inicial

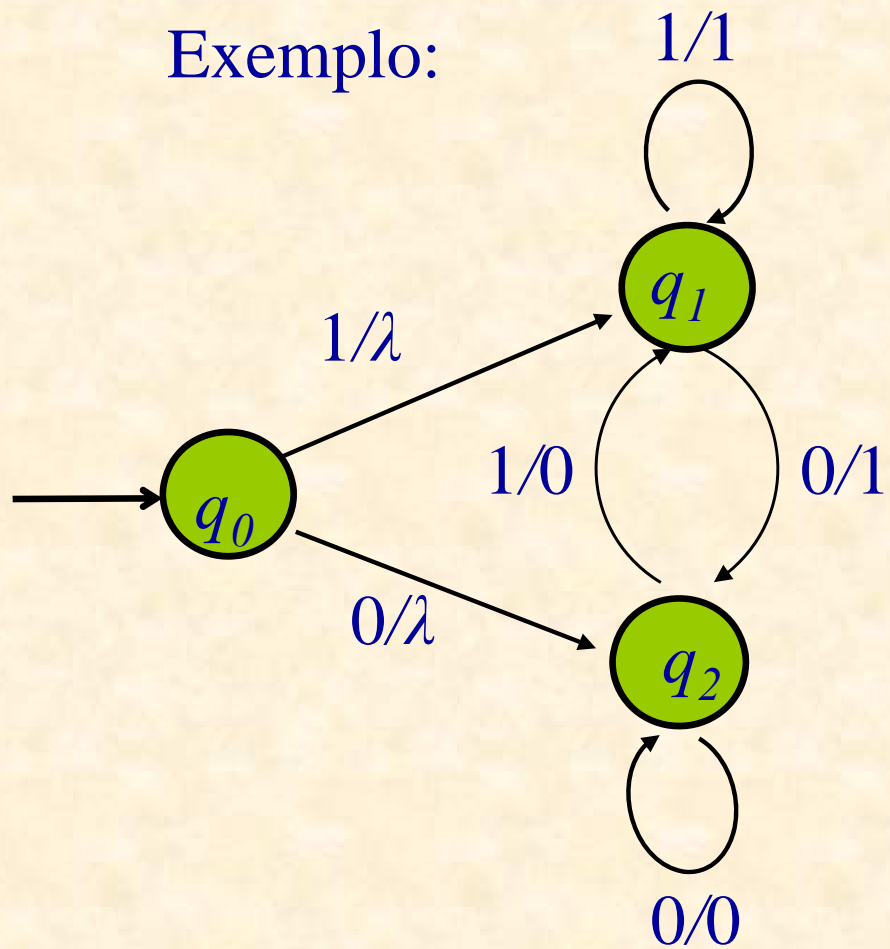
Se está no estado q_i e aparece a entrada 1, envia 0 para a saída e passa ao estado q_j

arestas



(¹) G. H. Mealy, *A Method for Synthesizing Sequential Circuits*, Bell System Tech. J. vol 34, pp. 1045–1079, Sept 1955.
© ADC/TCI/Cap.2/2000-10/LEI/DEIFCTUC

Exemplo:



Que faz ?

10110001...

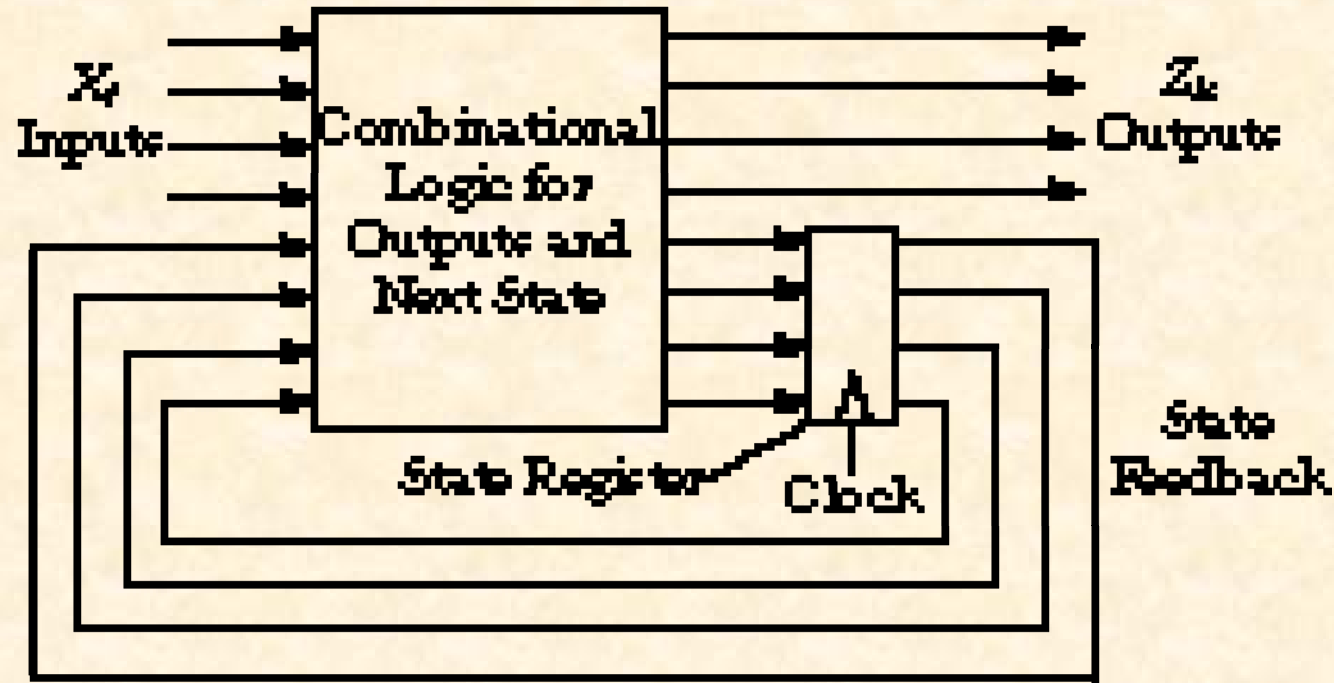


Máquina



????????

Máquina de Mealy



(<http://www2.ele.ufes.br/~ailson/digital2/cld/chapter8/chapter08.doc4.html>)

Logo que a entrada varia pode variar a saída:
funcionamento **assíncrono**

(Nota: também existem em versão síncrona, diferentes destas)

© ADC/TCI/Cap.2/2000-10/LEI/DEIFCTUC

2.5.2. Máquinas de Moore²

A saída depende apenas do estado (e não da entrada)

$$y_k = f(q_k)$$

Definição: Uma Máquina de Moore é definida pelo sexteto

$$M = (Q, \Sigma, \Lambda, \delta, \gamma, q_0)$$

Q : conjunto finito de estados internos

Σ : alfabeto de entrada (conjunto finito de caracteres)

Λ : alfabeto de saída (conjunto finito de caracteres)

$\delta: Q \times \Sigma \rightarrow Q$ é a função de transição de estado

$\gamma: Q \rightarrow \Lambda$ é a função de saída

$q_0 \in Q$ é o estado inicial

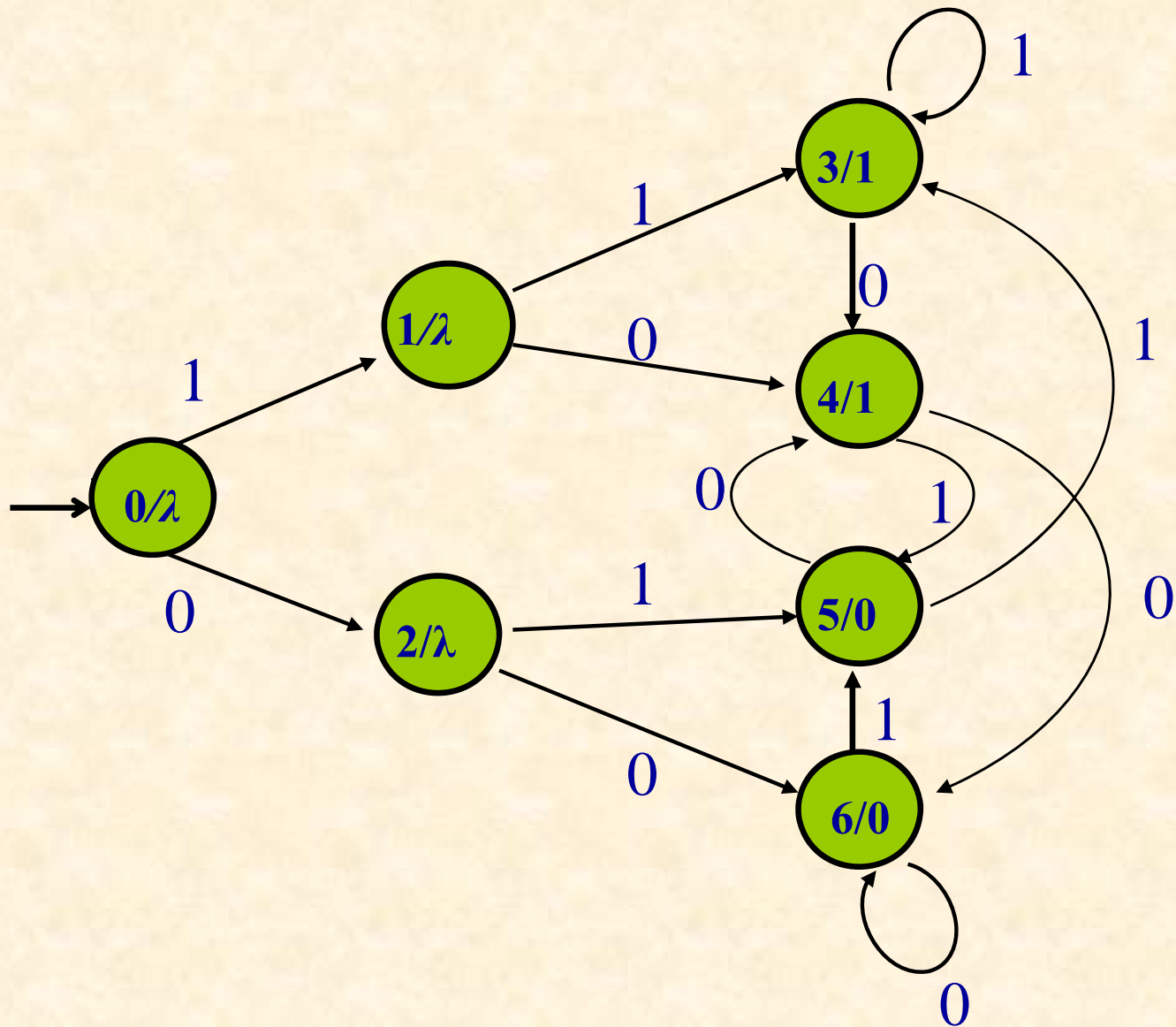
Se está no estado q_i (e saída 1) e aparece a entrada 1, transita para o estado q_j e a saída passa a 0

Estado/Saída



(²) E. F. Moore, *Gedanken-experiments on Sequential Machines*, pp 129 – 153, Automata Studies, Annals of Mathematical Studies, no. 34, Princeton University Press, Princeton, N. J., 1956.

Exemplo



Que faz ?

10110001...

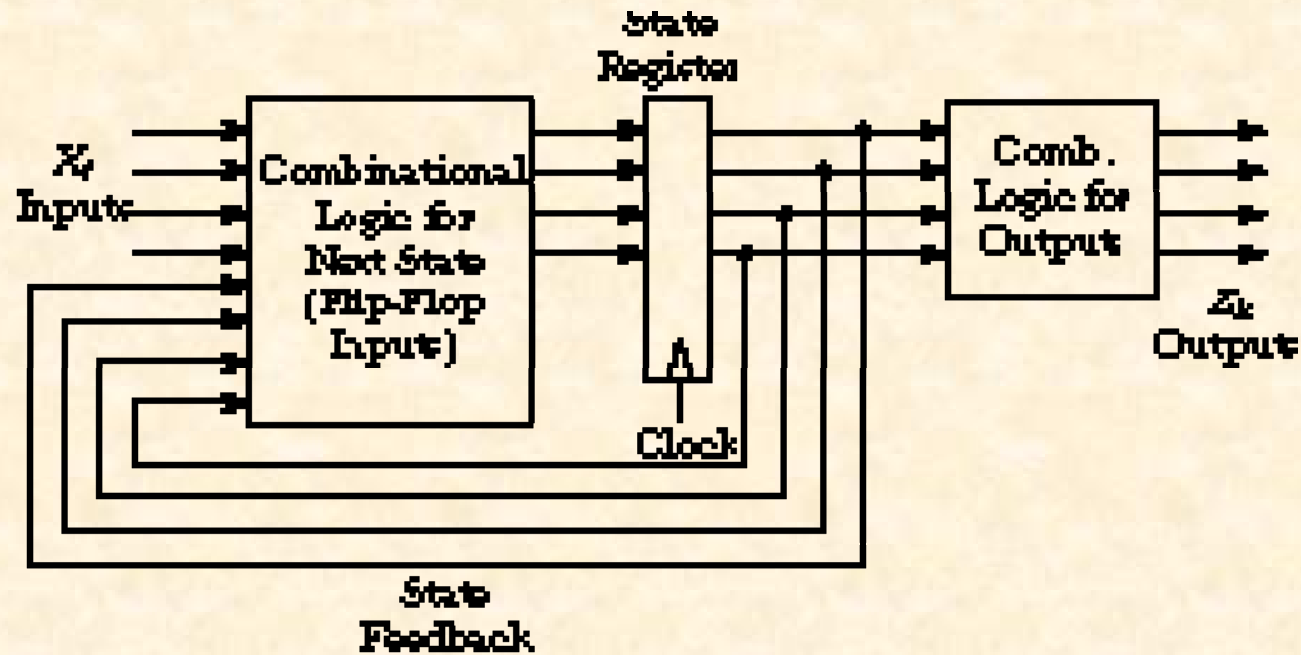


Máquina



????????

Máquina de Moore



(<http://www2.ele.ufes.br/~ailson/digital2/cld/chapter8/chapter08.doc4.html>)

A saída só pode variar depois de uma mudança de estado:
funcionamento **síncrono**

Equivalência entre as máquinas de Mealy e de Moore

Dada uma máquina de Mealy é possível encontrar uma máquina de Moore que, para as mesmas entradas, dá as mesmas saídas. E vice-versa. (Ignorando a saída do estado inicial na máquina de Moore).

A de Moore equivalente tem um maior número de estados.

Utilidade das máquinas de Mealy e de Moore

- Projectar circuitos lógicos
- Existe software que simula estas máquinas e gera o circuito lógico a partir do autómato. (ver por ex. http://www.seas.upenn.edu/~ee201/foundation/foundation_impl4.html ou <http://www.xilinx.com/univ/xse42.html>)

Bibliografia

An Introduction to Formal Languages and Automata, Peter Linz, 3rd Ed., Jones and Bartlett Computer Science, 2001

Introduction to Automata Theory, Languages and Computation, 2nd Ed., John Hopcroft, Rajeev Motwani, Jeffrey Ullman, Addison Wesley, 2001.

Elements for the Theory of Computation, Harry Lewis and Christos Papadimitriou, 2nd Ed., Prentice Hall, 1998.

Introduction to the Theory of Computation, Michael Sipser, PWS Publishing Co, 1997.

Wikipédia (enciclopédia livre) <http://en.wikipedia.org> ou <http://pt.wikipedia.org>)