LFA – Aula 08

Minimização de AFD
Autômatos Finitos com saídas

Celso Olivete Júnior

olivete@fct.unesp.br

www.fct.unesp.br/docentes/dmec/olivete/lfa

Na aula de hoje

- Minimização de autômatos finitos determinísticos
- Autômatos Finitos com saídas: Máquina de Moore e Máquina de Mealy

Minimização de autômatos finitos determinísticos

Minimização de AFD's

- O objetivo da minimização é gerar um Autômato Finito Determinístico equivalente com o menor número de estados possíveis.

- Em algumas aplicações especiais, a minimização do número de estados não implica necessariamente no menor custo de implementação

Minimização de AFD's

- Basicamente, o algoritmo de minimização unifica os estados equivalentes.

- Dois estados q e p são ditos equivalentes se, e somente se, para qualquer palavra w pertencente ao Σ^*, $\delta(q,w)$ e $\delta(p,w)$ resultam simultaneamente em estados finais, ou não-finais.

- O processamento de uma entrada qualquer a partir de estados equivalentes gera, em qualquer caso, o mesmo resultado aceita/rejeita.

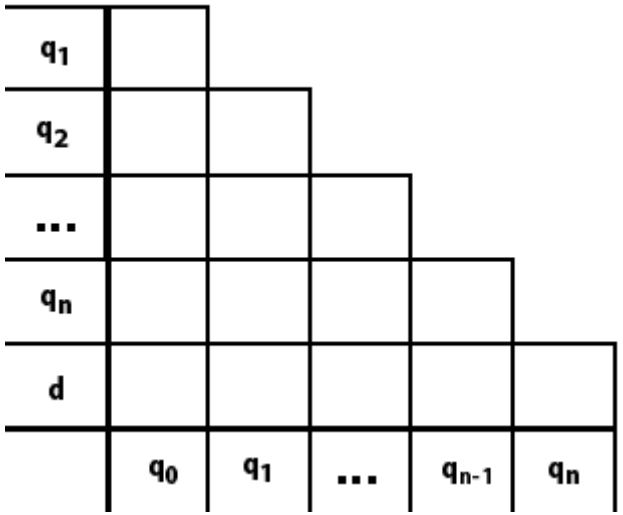
Algoritmo de minimização de AFD's

- Pré-requisitos:
  - Um AF a ser minimizado deve satisfazer:
    - Deve ser determinístico;
    - Não pode ter estados inacessíveis (não-atingíveis a partir do estado inicial);
    - A função programa deve ser total (a partir de qualquer estado são previstas transições para todos os símbolos do alfabeto).
- Obs:
  - Se os pré-requisitos não forem atendidos:
    - Caso o AF não seja AFD, deve-se transformá-lo
    - eliminar os estados inacessíveis
    - tornar a função programa total (é suficiente introduzir um novo estado não final **D** e incluir as transições não previstas, tendo **D** como estado destino. Por fim, incluir um ciclo em **D** para todos os símbolos do alfabeto.)

Algoritmo de minimização de AFD's

- Passos do algoritmo

1. Construir uma tabela relacionando os estados distintos onde cada par de estados ocorre apenas uma vez



Algoritmo de minimização de AFD's

2. Marcação dos estados trivialmente não-equivalentes

•Marcar todos os pares do tipo { estado final, estado não-final }, pois, obviamente, estados finais não são equivalentes a não-finais;

•Obs: verificar os estados de aceitação (finais) e marcar todos os pares que apresentam um destes estados;

Algoritmo de minimização de AFD's

3. Marcação dos estados não-equivalentes.

Para cada par {qu,qv} não-marcado e para cada símbolo a $\in \Sigma$, suponha que $\delta(qu, a) = pu$ e $\delta(qv, a) = pv$ e:

3.1 se pu = pv, então qu é equivalente a qv para o símbolo a e não deve ser marcado;

3.2 se pu $\neq$ pv e o par { pu, pv } não está marcado, então { qu, qv } é incluído em uma lista a partir de { pu, pv } para posterior análise;

3.3 se pu $\neq$ pv e o par { pu, pv } está marcado, então:

- {qu, qv} não é equivalente e deve ser marcado;
- se {qu, qv} encabeça uma lista de pares, então marcar todos os pares da lista (e, recursivamente, se algum par da lista encabeça outra lista);

Algoritmo de minimização de AFD's

4. Unificação dos estados equivalentes.

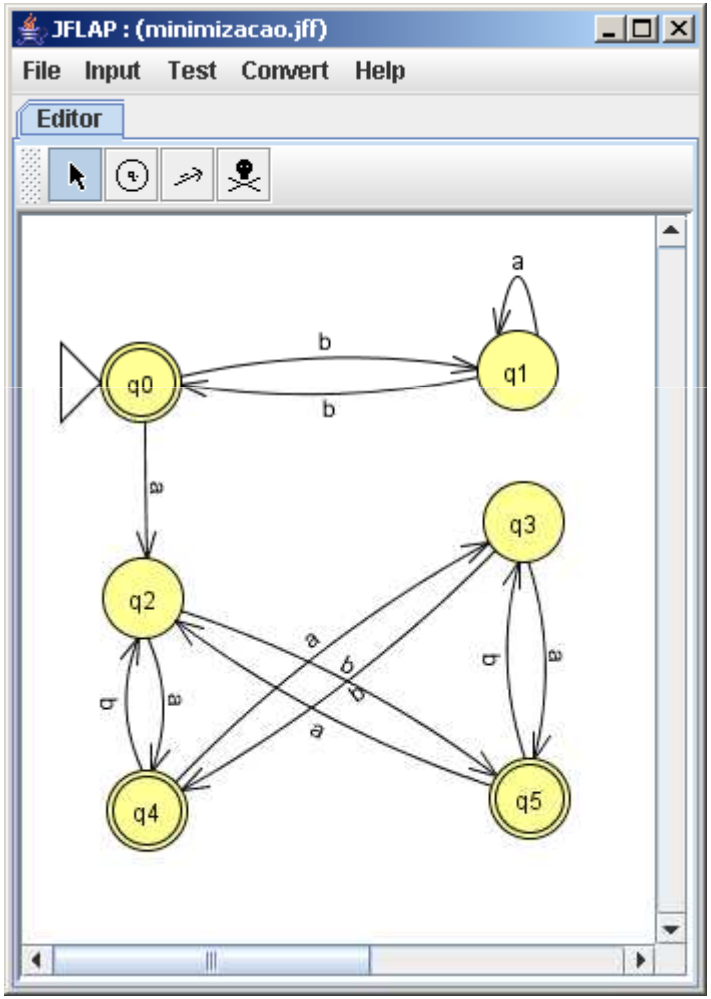
- Os estados dos pares não-marcados são equivalentes e podem ser unificados como segue:
  - pares de estados não-finais equivalentes podem ser unificados como um único estado não-final;
  - pares de estados finais equivalentes podem ser unificados como um único estado final;
  - se algum dos estados equivalentes é inicial, então o correspondente estado unificado é inicial;

Algoritmo de minimização de AFD's

5. Exclusão dos estados inúteis.

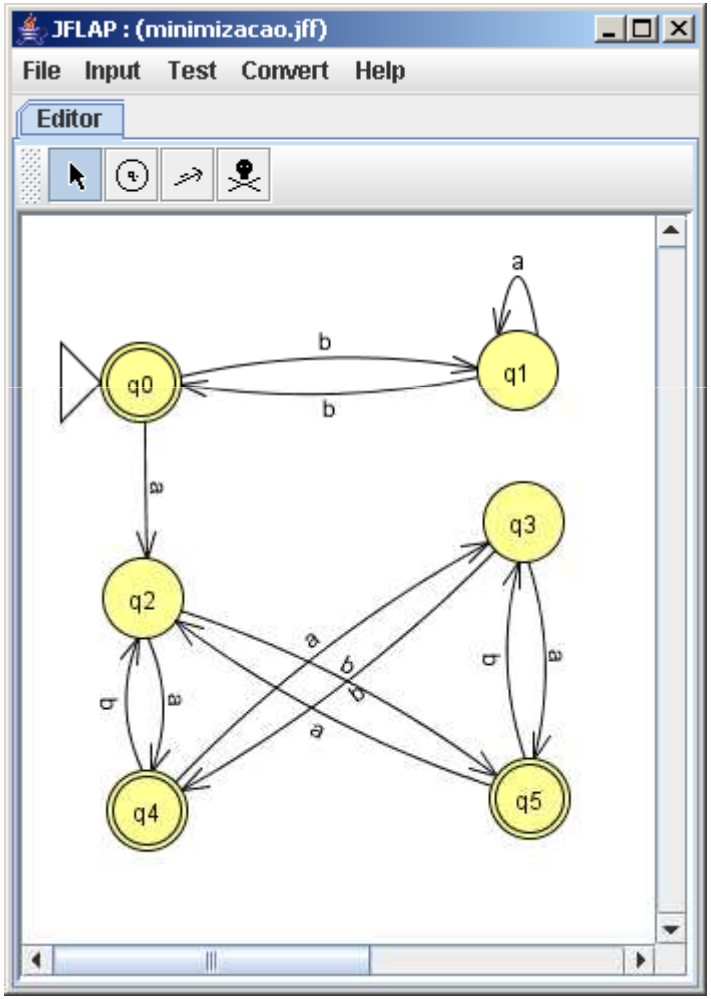
- Por fim, os estados chamados inúteis devem ser excluídos.
  - Um estado q é inútil se é não-final e a partir de q não é possível atingir um estado final.
  - Deve-se reparar que o estado **D** (se incluído) sempre é inútil.

Algoritmo de minimização de AFD's - exemplo



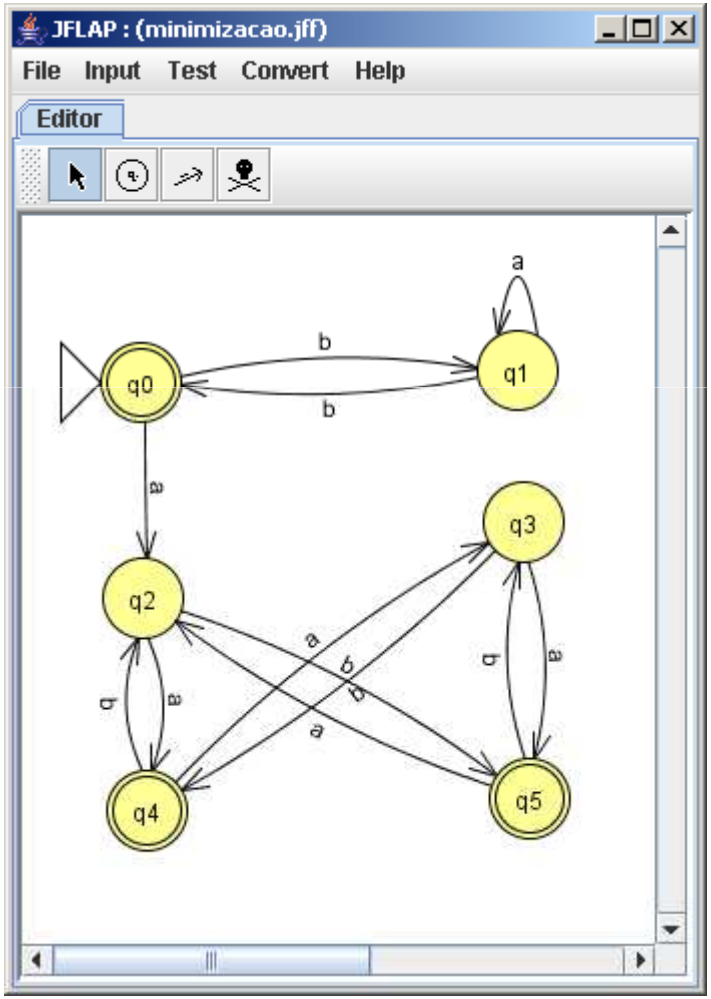
- pré-requisitos:
  - Deve ser determinístico;
  - Não pode ter estados inacessíveis (não-atingíveis a partir do estado inicial);
  - A função programa deve ser total (a partir de qualquer estado são previstas transições para todos os símbolos do alfabeto).

Algoritmo de minimização de AFD's - exemplo



- pré-requisitos:
  - Deve ser determinístico; **OK**
  - Não pode ter estados inacessíveis (não-atingíveis a partir do estado inicial); **OK**
  - A função programa deve ser total (a partir de qualquer estado são previstas transições para todos os símbolos do alfabeto). **OK**

Algoritmo de minimização de AFD's - exemplo



Construção da tabela (Passo 1) e marcação dos estados do tipo {estado final, estado não-final} (Passo 2)

q_1	×				
q_2	×				
q_3	×				
q_4		×	×	×	
q_5		×	×	×	
	q_0	q_1	q_2	q_3	q_4

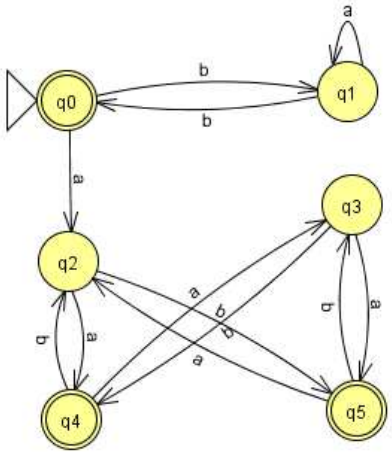
(q0,q1)

final,não-final
(q0,q3)

Obs: como q0, q4 e q5 são finais, inserimos X em cada par que os envolve

Algoritmo de minimização de AFD's

q_1	×				
q_2	×				
q_3	×				
q_4	●	×	×	×	
q_5		×	×	×	
	q_0	q_1	q_2	q_3	q_4



Análise dos pares de estado não-marcados

c-1) Análise do par { q0, q4 }:

(q0, a) = q2

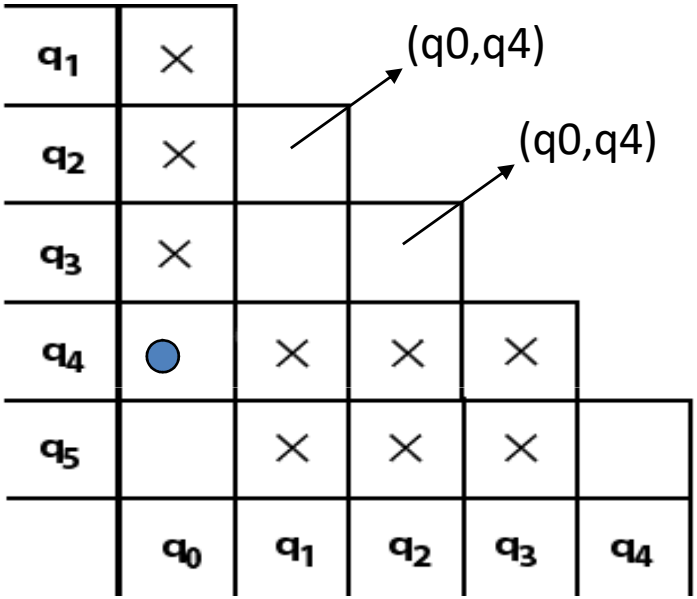
(q0, b) = q1

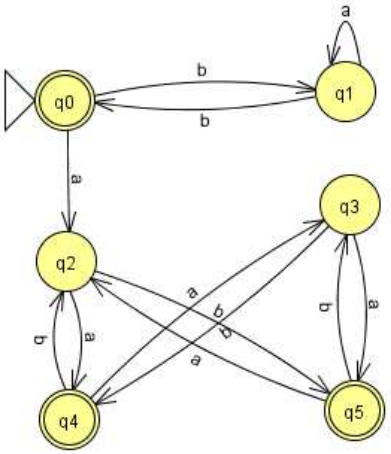
(q4, a) = q3

(q4, b) = q2

- Como { q1, q2 } e { q2, q3 } são não-marcados, então { q0, q4 } é incluído nas listas encabeçadas por { q1, q2 } e { q2, q3 };

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-1) Análise do par { q0, q4 }:

(q0, a) = q2

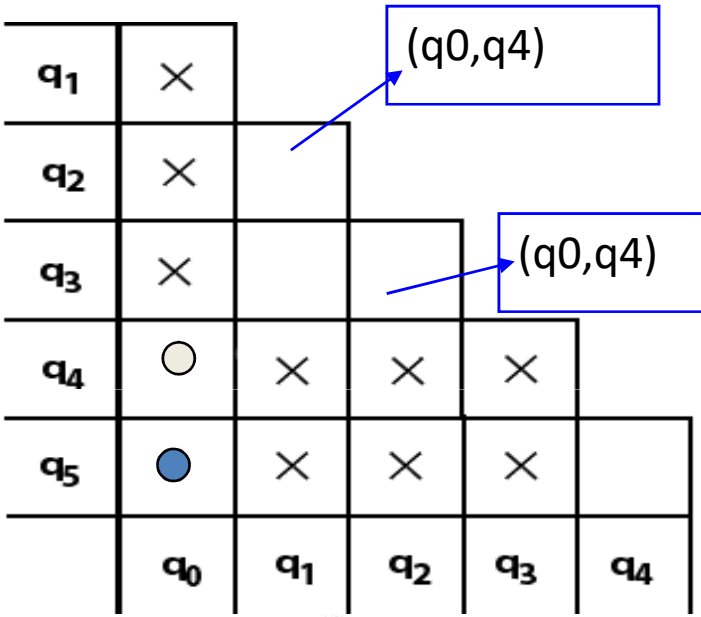
(q0, b) = q1

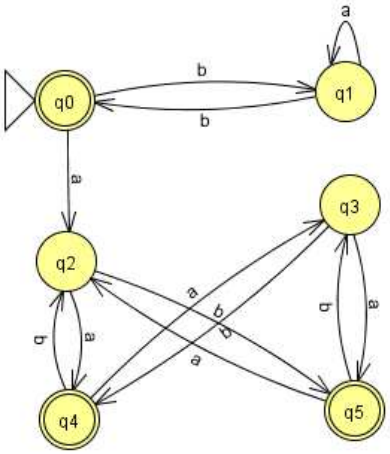
(q4, a) = q3

(q4, b) = q2

- Como { q1, q2 } e { q2, q3 } são não-marcados, então { q0, q4 } é incluído nas listas encabeçadas por { q1, q2 } e { q2, q3 };

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-2) Análise do par { q0, q5 }:

(q0, a) = q2

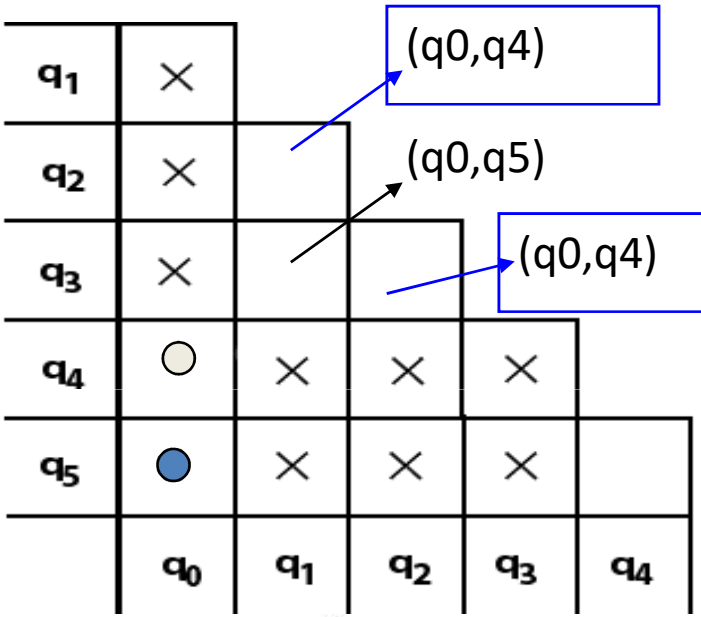
(q0, b) = q1

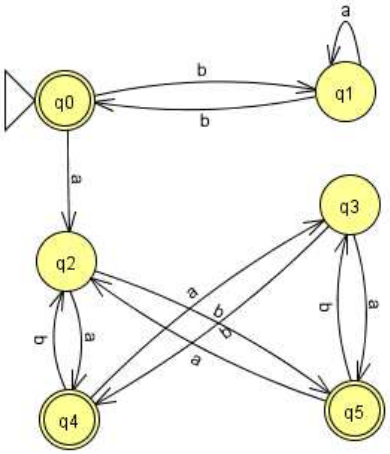
(q5, a) = q2

(q5, b) = q3

- Como { q1, q3 } é não-marcado (e como { q2, q2 } é trivialmente equivalente), então { q0, q5 } é incluído na lista encabeçada por { q1, q3 };

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-2) Análise do par { q0, q5 }:

(q0, a) = q2

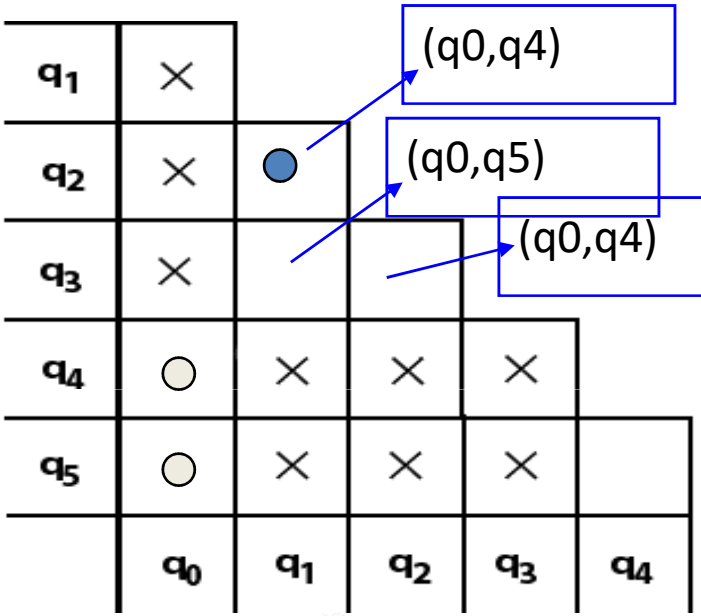
(q0, b) = q1

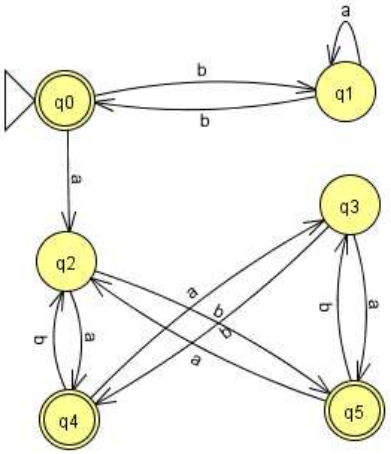
(q5, a) = q2

(q5, b) = q3

- Como { q1, q3 } é não-marcado (e como { q2, q2 } é trivialmente equivalente), então { q0, q5 } é incluído na lista encabeçada por { q1, q3 };

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-3) Análise do par { q1, q2 }:

(q1, a) = q1

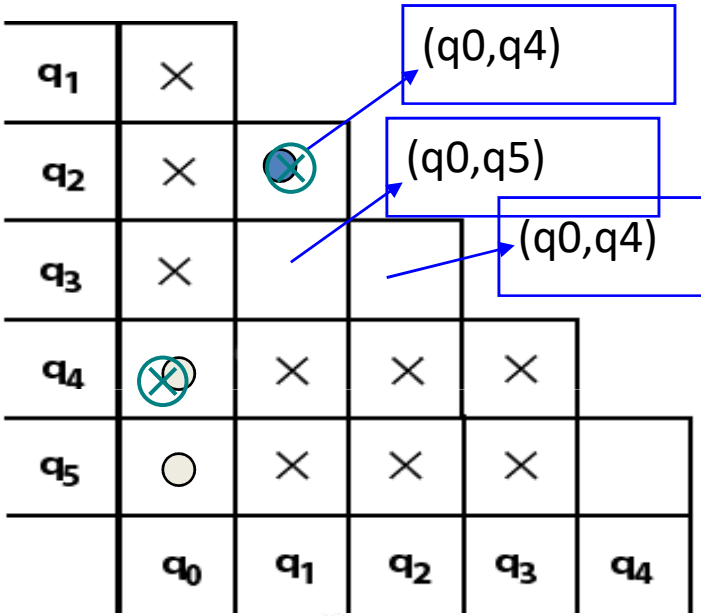
(q1, b) = q0

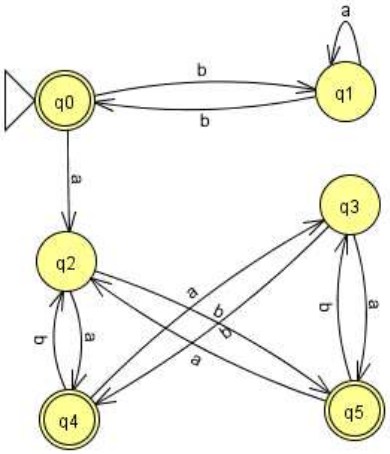
(q2, a) = q4

(q2, b) = q5

- Como { q1, q4 } é marcado, então { q1, q2 } também é marcado.
- Como { q1, q2 } encabeça uma lista, o par { q0, q4 } também é marcado;

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-3) Análise do par { q1, q2 }:

(q1, a) = q1

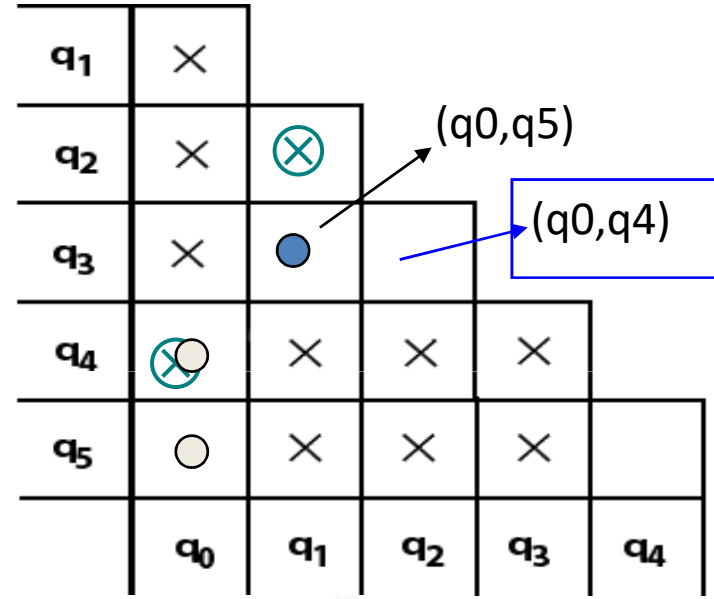
(q1, b) = q0

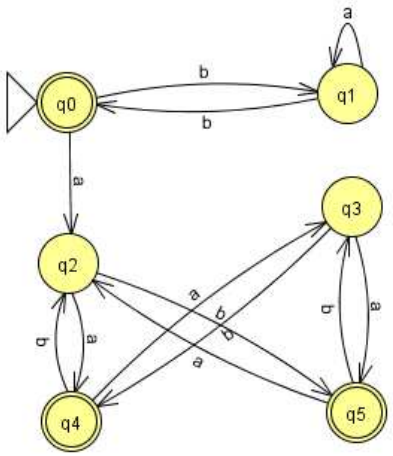
(q2, a) = q4

(q2, b) = q5

- Como { q1, q4 } é marcado, então { q1, q2 } também é marcado.
- Como { q1, q2 } encabeça uma lista, o par { q0, q4 } também é marcado;

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-4) Análise do par { q1, q3 }:

(q1, a) = q1

(q1, b) = q0

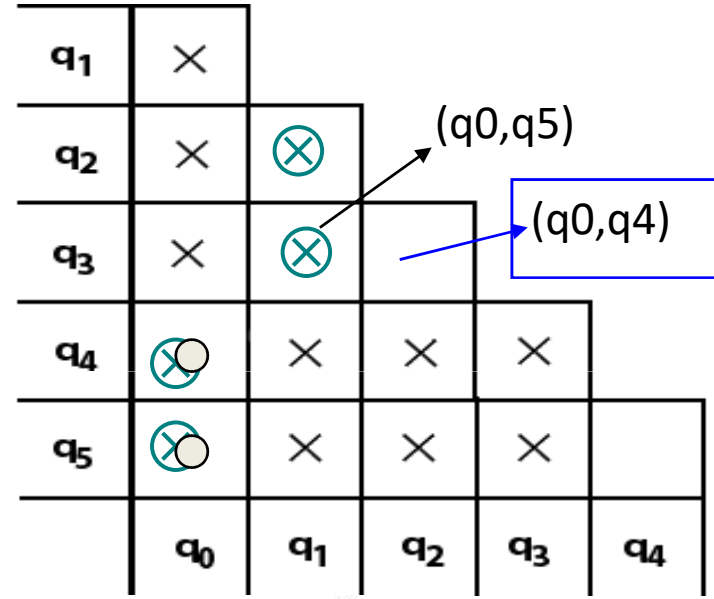
(q3, a) = q5

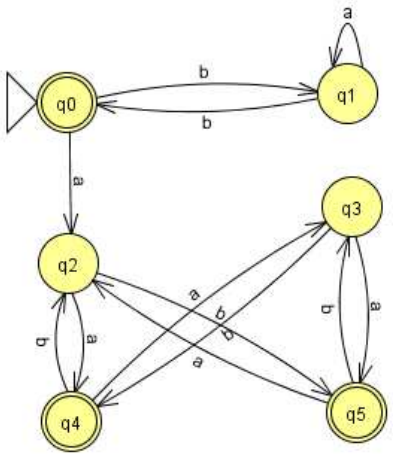
(q3, b) = q4

Como { q1, q5 } bem como { q0, q4 } são marcados, então { q1, q3 } também é marcado.

Como { q1, q3 } encabeça uma lista, o par { q0, q5 } também é marcado;

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-4) Análise do par { q1, q3 }:

(q1, a) = q1

(q1, b) = q0

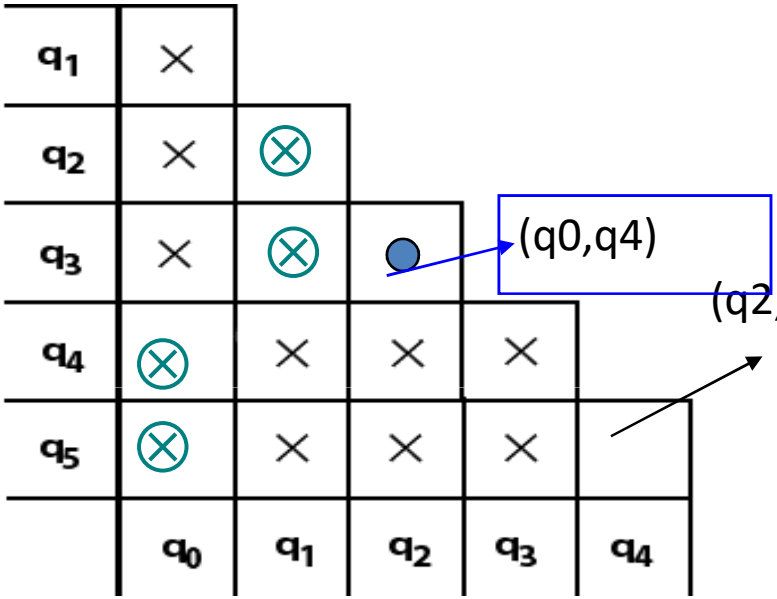
(q3, a) = q5

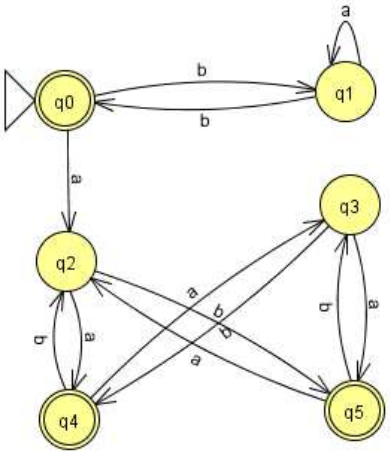
(q3, b) = q4

Como { q1, q5 } bem como { q0, q4 } são marcados, então { q1, q3 } também é marcado.

Como { q1, q3 } encabeça uma lista, o par { q0, q5 } também é marcado;

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-5) Análise do par { q2, q3 }:

(q2, a) = q4

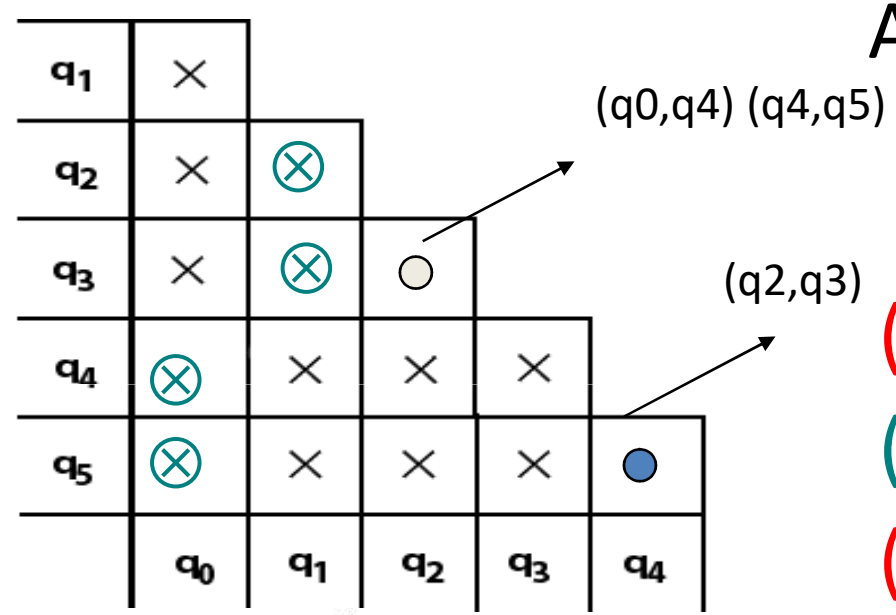
(q2, b) = q5

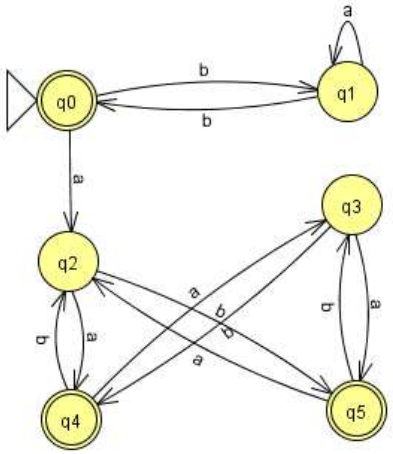
(q3, a) = q5

(q3, b) = q4

Como { q4, q5 } é não-marcado, então { q2, q3 } é incluído na lista encabeçada por { q4, q5 };

Algoritmo de minimização de AFD's





Análise dos pares de estado não-marcados

c-6) Análise do par { q4, q5 }:

(q4, a) = q3

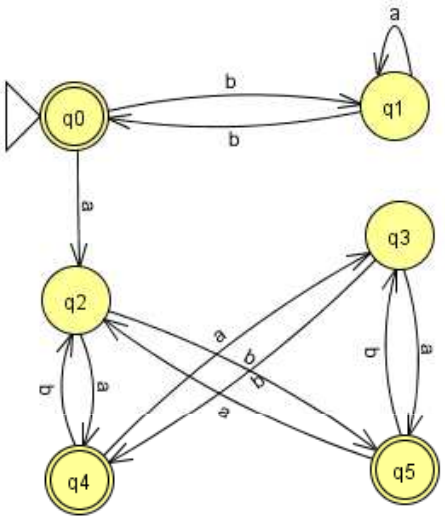
(q4, b) = q2

(q5, a) = q2

(q5, b) = q3

Como { q2, q3 } é não-marcado, então { q4, q5 } é incluído na lista encabeçada por { q2, q3 };

Algoritmo de minimização de AFD's



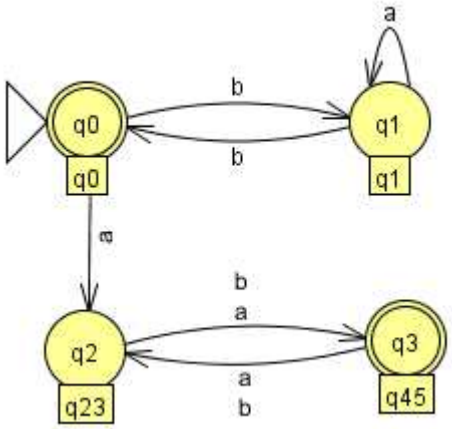
Unificação

d) Como os pares { q2, q3 } e { q4, q5 } são não-marcados, as seguintes unificações podem ser feitas:

q23 representa a unificação dos estados não-finais q2 e q3;

q45 representa a unificação dos estados finais q4 e q5.

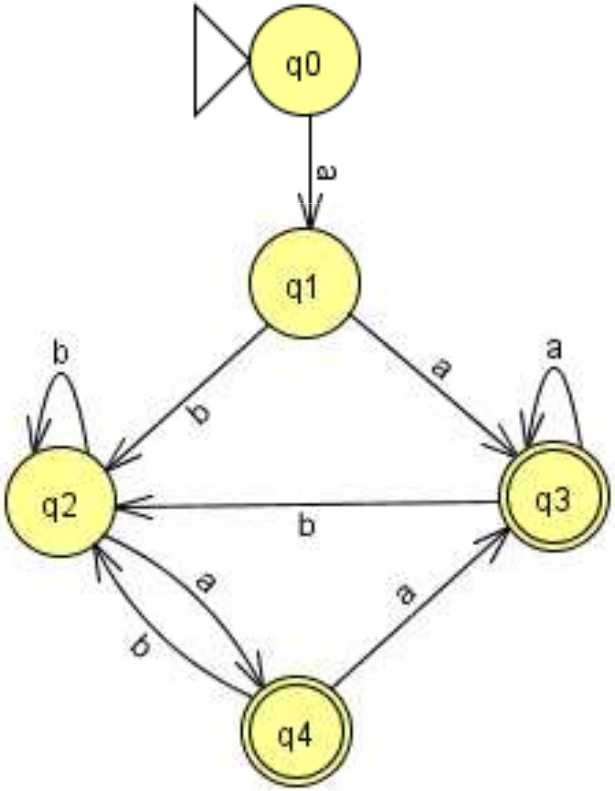
AF mínimo

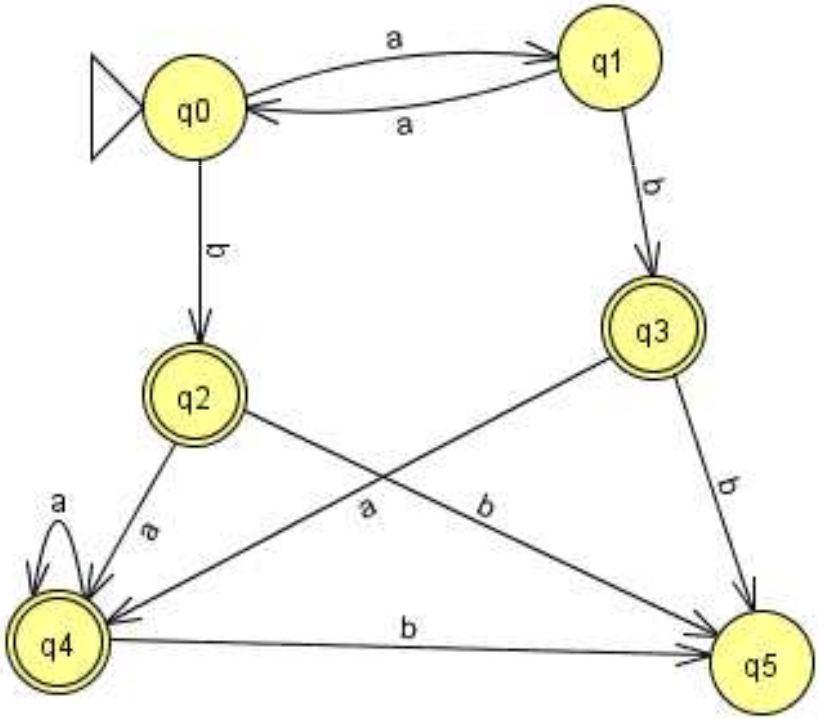


q_1	×				
q_2	×	⊗			
q_3	×	⊗			
q_4	⊗	×	×	×	
q_5	⊗	×	×	×	
	q_0	q_1	q_2	q_3	q_4

Exercícios

1. Minimize os seguintes AFD's utilizando o algoritmo apresentado anteriormente





Exercícios

2. Considere o AF não determinista M = ({1,2,3,4,5}, {x,y}, *d*, 1, {5}), com a função de transição dada na tabela

d	x	y
→ 1	{1}	{1,2}
2	Ø	{3}
3	{3}	{3,4}
4	Ø	{5}
* 5	{5}	{5}

Construa o AFD correspondente

Construa o AFD *mínimo* correspondente, utilizando o algoritmo apresentado.

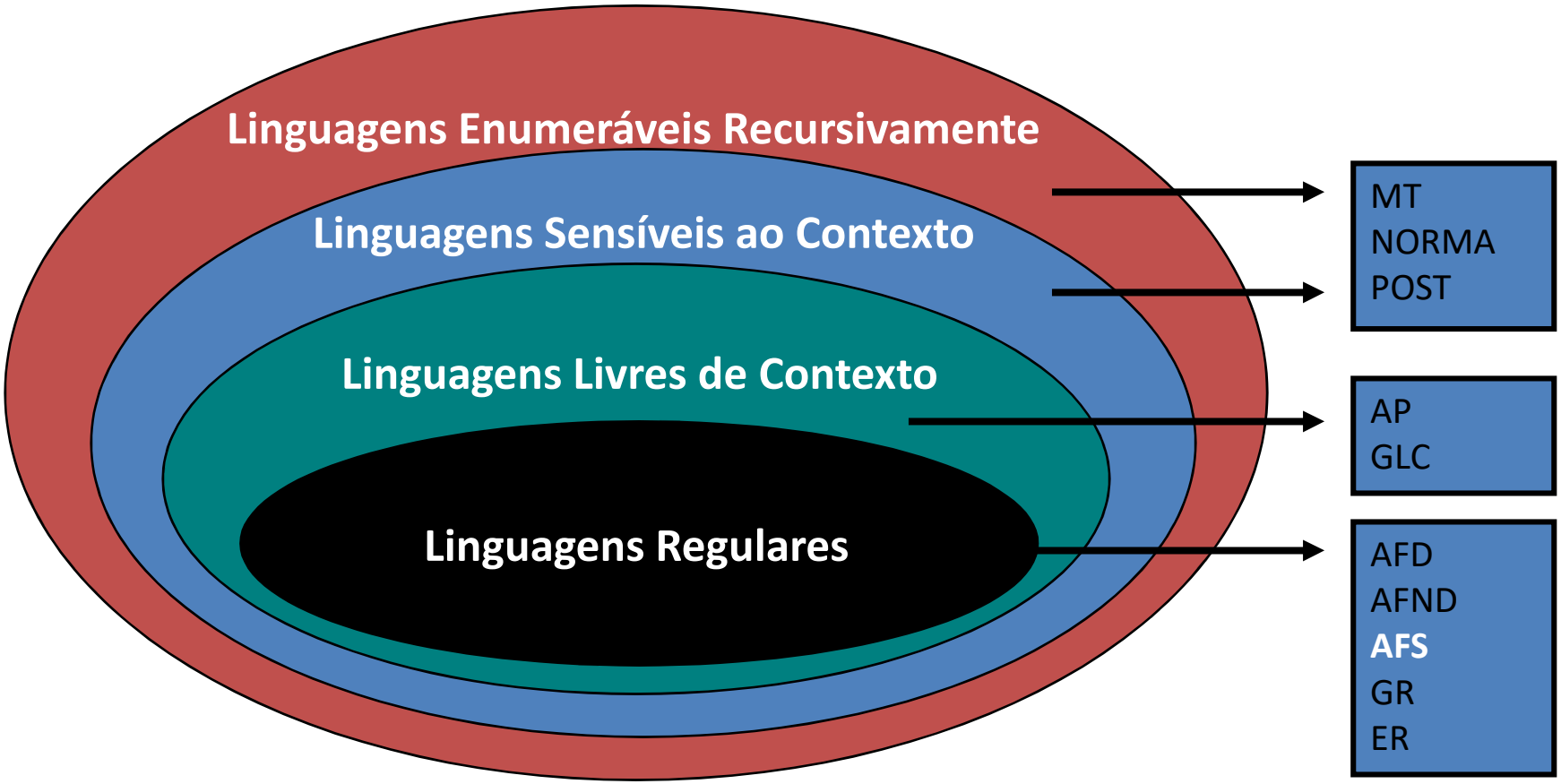
Exercícios

3. Minimize o AFD:

(→q0,a)=(q1)
(q0,b)=(q0)
(q1,a)=(q2)
(q1,b)=(q0)
(q2,a)=(q3)
(q2,b)=(q2)
(*q3,a)=(q3)

4. Construa um AFD que reconheça **(a + b)+ b*(a+b)*** e construa um AFD-mínimo, se possível.

Hierarquia de Chomsky

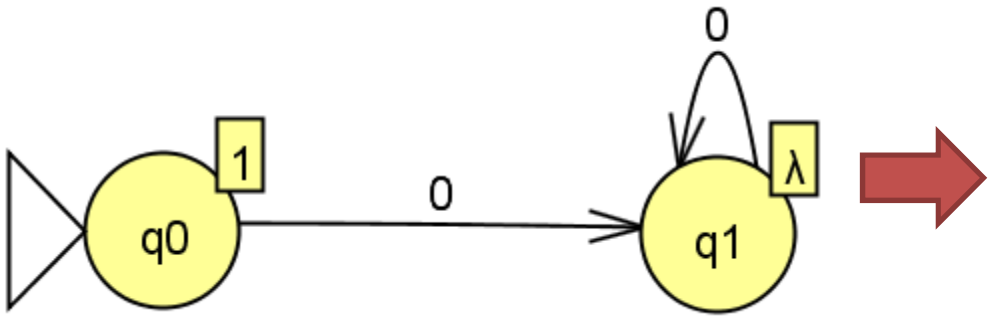


Autômatos Finitos com Saídas (AFS):

Máquina de Moore e Máquina de Mealy

AFS - Máquina de Moore

- Extensões para um AF que associam, para cada estado uma saída sobre o alfabeto, eventualmente distinto do alfabeto de entrada;
  - Máquina de Moore: para cada estado existe uma saída



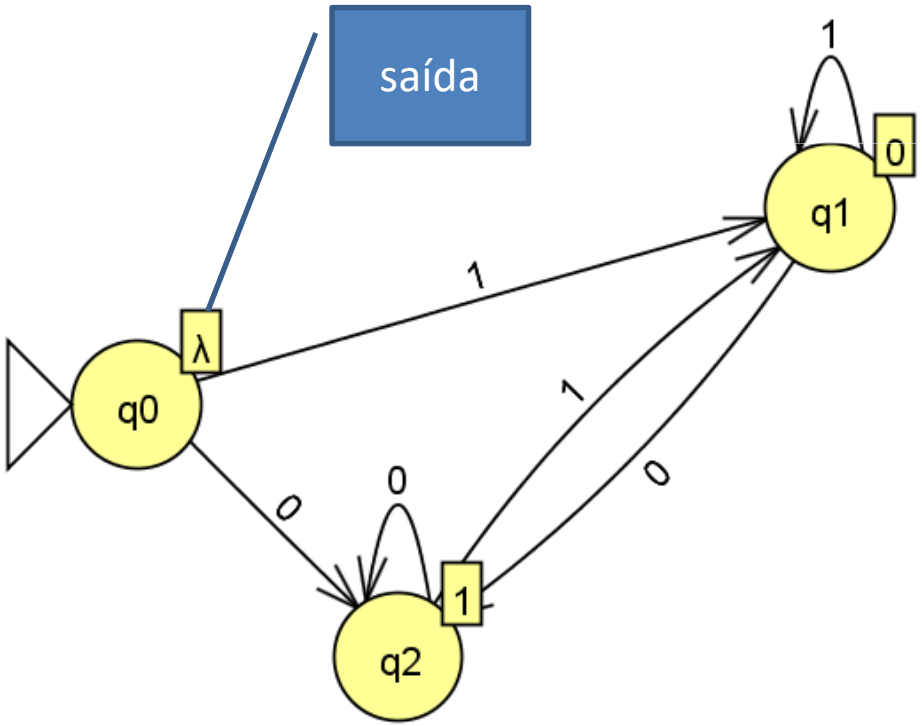
Ao ler o símbolo **0** (q0) gera 1 na saída (q0) e avança para o estado q1. Em q1, a cada 0 lido, gera λ na saída(q1).

Diferenças entre Máquina de Moore e AF:

- Máquina de Moore não tem estado final;
- Cada estado produz uma saída;
- Não aceita ou rejeita a entrada, em vez disso, ele gera uma saída para a entrada;
- Máquina de Moore deve ser determinística.

Máquina de Moore

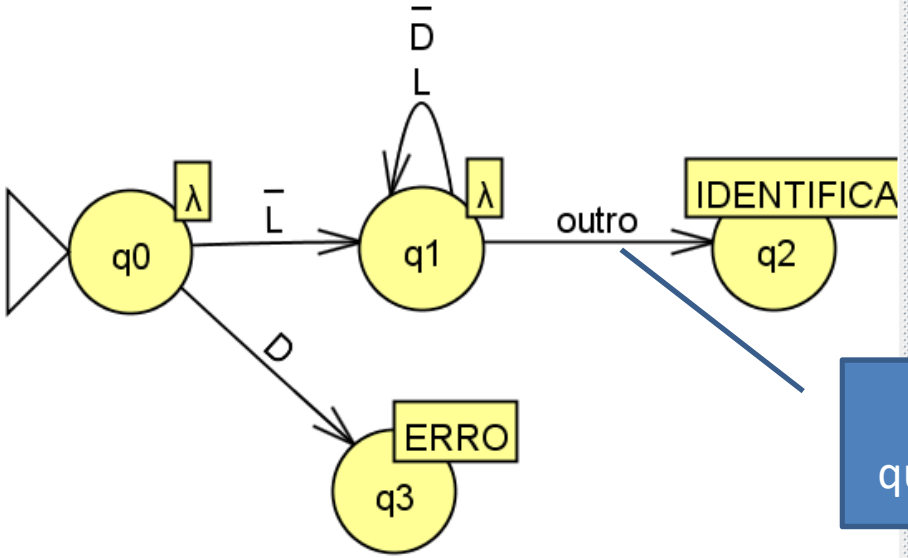
- Exemplo 1: *NOT*(b): recebe 0 ou 1 e devolve a operação NOT



Input	Result
11	00
101010	010101
00	11
10	01

Máquina de Moore

- Exemplo 2: reconhece *IDENTIFICADOR*

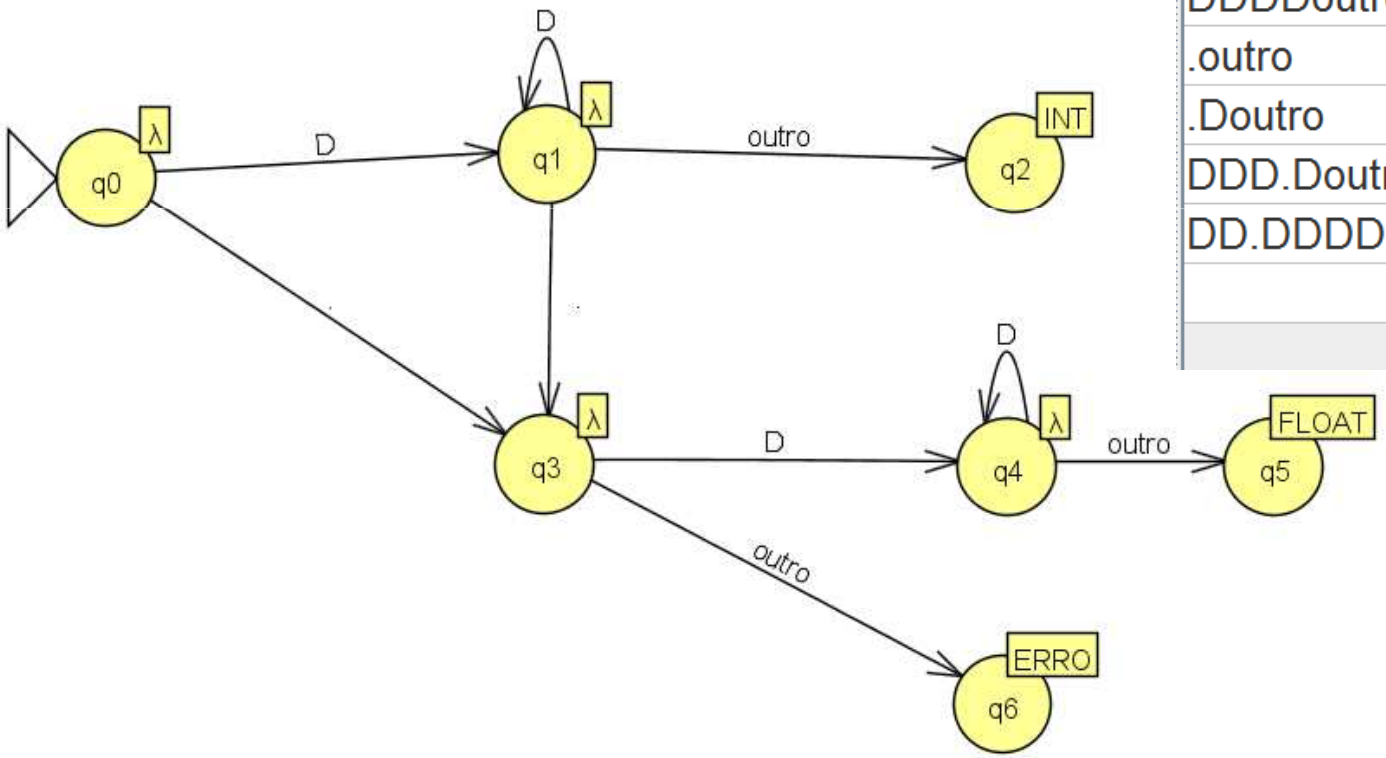


Input	Result
LD_outro	IDENTIFICADOR
Loutro	IDENTIFICADOR
D	ERRO

Um delimitador qualquer. Ex: espaço

Máquina de Moore

- Exemplo 3: reconhece *INT* ou *FLOAT*



Input	Result
DDDDoutro	INT
.outro	ERRO
.Doutro	FLOAT
DDD.Doutro	FLOAT
DD.DDDDDoutro	FLOAT

Máquina de Moore

- Denotado por $AF_{Moore} = (\{Q\}, \Sigma, \alpha, \delta, \gamma, q0)$
  - Um conjunto finito de estados, frequentemente denotado por Q
  - Um alfabeto de entrada, denotado pelo Σ
  - Alfabeto de saída para cada estado $\in \{Q\}$
  - Uma função transição δ que toma como argumentos um estado e um símbolo de entrada e retorna um novo estado.
  - Função transição γ: $Q \rightarrow \alpha$
  - q0 - Um estado inicial (pertencente a Q)

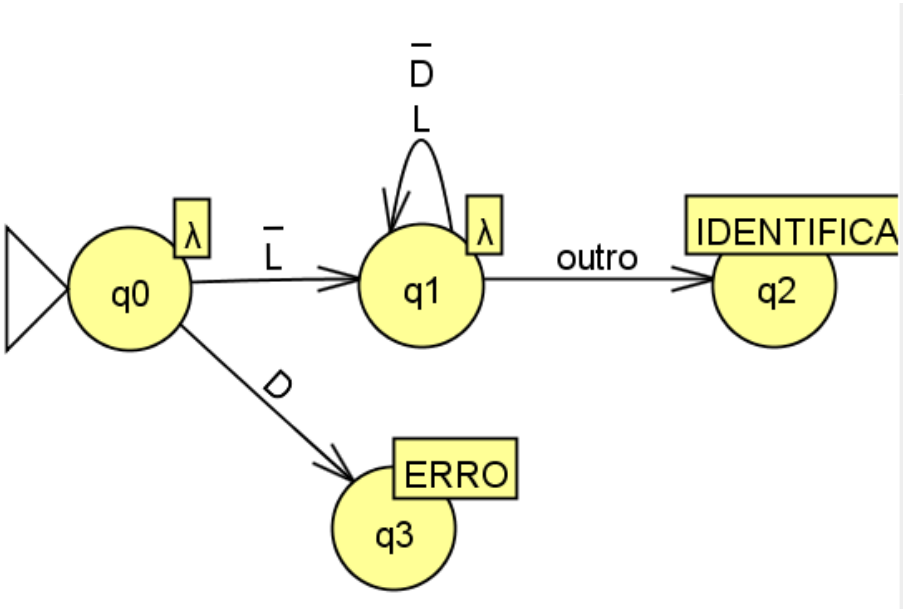
Máquina de Moore

$AF_{Moore} = (\{Q\}, \Sigma, \alpha, \delta, \gamma, q0, \{F\})$

- $AF_{MooreID}$ = ({q0,q1,q2,q3}, {L,_,D,outro}, {λ, ERRO, IDENTIFICADOR}, δ, γ, q0)

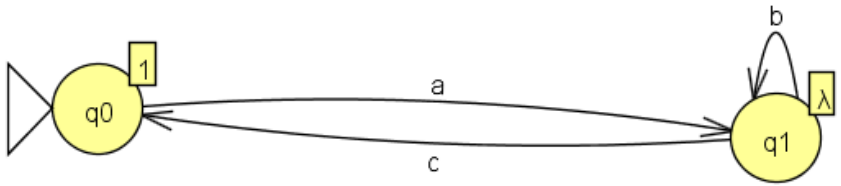
γ

q0 → λ
q1 → λ
q2 → IDENTIFICADOR
q3 → ERRO



Máquina de Moore

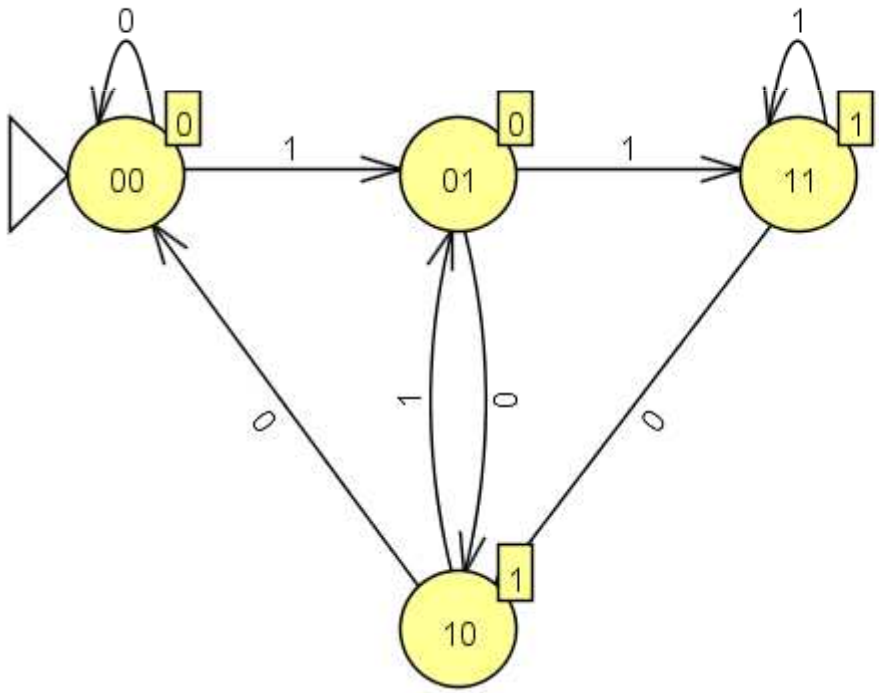
• Exemplo 4: Uma máquina de Moore que aceita a linguagem **ab*(cab*)***, ou seja, uma sequência de uma ou mais cadeias **ab** separadas pelo símbolo **c**. A função de transição γ, neste caso, faz com que a máquina emita o símbolo "1" toda vez que estiver iniciando o reconhecimento de uma nova cadeia com o formato ab*. Assim, a máquina funciona como um contador do número de subcadeias ab presentes na cadeia de entrada.



Input	Result
ab	1
abcab	11

Máquina de Moore

- Exemplo 5: Número binário dividido por 2

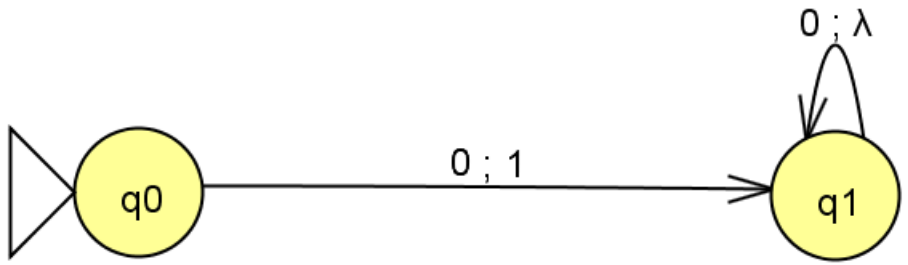


Input	Result
1010	00101
100	0010

Autômatos Finitos com saídas: Máquina de Moore e Máquina de Mealy

AFS - Máquina de Mealy

- Extensões para um AF que associam, a cada transição uma correspondente cadeia de saída sobre um segundo alfabeto, eventualmente distinto do alfabeto de entrada;
  - para cada transição existe uma saída



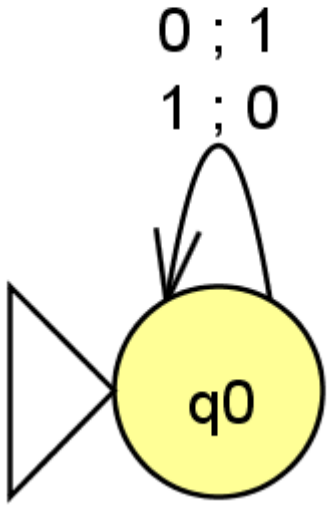
Ao ler o símbolo **0** (q0) gera 1 na transição (q0-q1) e avança para o estado q1. Em q1, a cada 0 lido, gera λ na transição(q1-q1).

Diferenças entre Máquina de Mealy e AF:

- Não tem estado final;
- Cada transição produz uma saída;
- Não aceita ou rejeita a entrada, em vez disso, ela gera uma saída para a entrada;
- Máquina de Mealy deve ser determinística.

Máquina de Mealy

- Exemplo 1: *NOT*(b): recebe 0 ou 1 e devolve NOT



Input	Result
101	010
00	11
11	00

Máquina de Mealy

- Denotado por $AF_{Moore} = (\{Q\}, \Sigma, \alpha, \delta, \gamma, q0)$
  - Um conjunto finito de estados, frequentemente denotado por Q
  - Um alfabeto de entrada, denotado pelo Σ
  - Alfabeto de saída para cada estado $\in \{Q\}$
  - Uma função transição δ que toma como argumentos um estado e um símbolo de entrada e retorna um novo estado.
  - Função transição γ: $Q \times \Sigma \rightarrow \alpha$
  - q0 - Um estado inicial (pertencente a Q)

Máquina de Moore e Mealy - Equivalência

- Apesar de se tratar de dois modelos distintos de AF, pode-se demonstrar a plena equivalência de ambos: toda e qualquer Máquina de Moore pode ser simulada por uma Máquina de Mealy e vice-versa. Dessa maneira, portanto, a opção por um ou outro tipo de máquina pode ser feita levando-se em conta exclusivamente a conveniência de manipulação e a facilidade de representação obtidas conforme o caso em questão.

Exercícios

1. Construa uma máquina (Mealy ou Moore) para o problema abaixo:

A empresa de refrigerantes X deseja projetar um circuito que realize o controle de venda de 1 lata de refrigerante na sua máquina de refrigerantes. Para isto o empresa o contratou, você deve especificar o diagrama de uma máquina estados finitos que realize o controle da entrada de moedas na máquina. Se entrar o valor correto a latinha deve sair da máquina, caso contrário, deve voltar para o estado inicial e devolver as moedas. Sabe-se que o preço do refrigerante é um real, e também que a máquina somente aceita moedas de 1 real, 50 centavos e 25 centavos. Porém, a máquina pode aceitar qualquer sequência de moedas.

2. Construa uma Maquina de Mealy que reconheça uma palavra w, $|w| \geq 1$, tal que $w = x^+$ e produza uma sequencia de saída $v = (ab)^+$, em que $|v| = 2|w|$.

Exercícios

3. Construa o diagrama de estados de uma máquina de estados finitos que realize o controle de um elevador. O elevador deverá respeitar a seguinte especificação:

• Se o elevador está parado e o andar requisitado é igual ao andar corrente , então o elevador continua parado.

• Se o elevador está parado e o andar requisitado é menor que o andar corrente, então o elevador deve descer para o andar desejado.

• Se o elevador está parado e o andar requisitado é maior que o andar corrente, então o elevador deve subir para o andar desejado. Para mais informações consulte o artigo **"Modelos Orientados a Estado na Especificação de Software"** no endereço http://www.uel.br/revistas/uel/index.php/semexatas/article/view/1571/1322

Sendo assim:

a) Desenhe o diagrama da máquina de estados que realiza esta operação.

b) Esta máquina é Moore ou Mealy? Por quê ?

Exercícios

Enviar todos os exercícios até o dia 30/10