UDESC

LFA0001 – Linguagens Formais e Autômatos
Aula 03
Linguagens Regulares

Karina Girardi Roggia
karina.roggia@udesc.br

Departamento de Ciência da Computação
Centro de Ciências Tecnológicas
Universidade do Estado de Santa Catarina

2016

UDESC

Sumário

Linguagens Regulares

Formalismos

- Autômato Finito
  - formalismo operacional ou reconhecedor
  - sistema de estados finitos
- Expressão Regular
  - formalismo denotacional ou gerador
  - conjuntos básicos + concatenação e união
- Gramática Regular
  - formalismo axiomático ou gerador
  - gramática com restrições da forma das regras de produção

Linguagens Regulares

Dentro da Hierarquia de Chomsky

- mais simples classe de linguagens
- reconhecimento ou geração de palavras e conversão entre formalismos
  - pouca complexidade
  - grande eficiência
  - fácil implementação

Fortes limitações de expressividade

- Linguagens de programação em geral não são regulares
- Exemplo de linguagem não regular: duplo balanceamento

Complexidade de Autômatos Finitos

- Classe de algoritmos mais eficientes em tempo de processamento
  - supondo que toda a entrada deve ser lida
- **Qualquer** autômato finito é igualmente eficiente
- Qualquer solução é ótima
  - a menos de eventual redundância de estados
  - a redundância de estados não influi no tempo de execução
  - pode ser eliminada: algoritmo de Autômato Finito Mínimo

Sistema de Estados Finitos

- Modelo matemático de sistema com entradas e saídas discretas
- Número finito e predefinido de estados

Estados servirão para guardar informações do passado necessárias para determinar as ações para o próximo passo.

Exemplos

Elevador

- Não memoriza as requisições anteriores
- Estado: andar corrente + direção de movimento
- Entrada: requisições pendentes

Analisador Léxico

- Estado: memoriza a estrutura do prefixo da palavra em análise
- Entrada: texto

Contra-exemplo

Nem todos os sistemas de estados finitos são adequados para esta abordagem

Cérebro humano

- cerca de 2^{35} células
- abordagem pouco eficiente
- explosão de estados

Composição de Sistemas

Sistemas complexos construídos a partir de sistemas mais simples.

Três principais formas de composição:

- Sequencial
- Concorrente
- Não Determinista

Formas de Composição

Sequencial

- execução da próxima componente
- depende da terminação da componente anterior

Concorrente

- componentes independentes
  - ordem de execução não é importante
- podem ser processadas em paralelo

Formas de Composição

Não Determinista

- próxima componente: *escolha* entre diversas alternativas
- em oposição à determinista
  - mesmas condições
  - sempre mesma escolha
- não-determinismo pode ser
  - interno: sistema escolhe aleatoriamente
  - externo: escolha externa ao sistema

Exemplo: Banco

Sequencial

- fila: próximo cliente depende do atendimento do anterior

Concorrente

- diversos caixas atendem independentemente diversos clientes
- as ações dos clientes que estão sendo atendidos são independentes dos clientes da fila

Não-determinista

- dois ou mais caixas eletrônicos disponíveis ao mesmo tempo
- o cliente pode escolher qual das máquinas utilizará

Não Determinismo Interno

- Semântica usual para Linguagens Formais e Teoria da Computação
- Objetivo: determinar a capacidade de reconhecer linguagens e se solucionar problemas
- Difere da adotada no estudo dos Modelos para Concorrência (exemplo: Sistemas Operacionais)

Autômato Finito

Sistema de Estados Finitos

- Número finito e predefinido de estados
- Modelo computacional comum em diversos estudos teóricos-formais
  - Linguagens Formais
  - Compiladores
  - Semântica Formal
  - Modelos para Concorrência

Variantes

Determinístico

- a partir do estado corrente e do símbolo lido
- assume um **único** estado

Não-determinístico

- a partir do estado corrente e do símbolo lido
- assume um **conjunto** de estados alternativos

Com movimentos vazios

- a partir do estado corrente e **sem ler** qualquer símbolo
- pode assumir um **conjunto** de estados alternativos

Os três tipos de Autômatos Finitos são **equivalentes** em termos de poder computacional.

Componentes

Fita

- dispositivo de entrada
- contém a informação a ser processada

Unidade de Controle

- reflete o estado corrente da máquina
- possui unidade de leitura (cabeçote)
- acessa uma célula da fita de cada vez
- movimenta-se exclusivamente para a direita

Programa (Função Programa ou Função de Transição)

- comanda as leituras
- define o estado da máquina

Fita

- Finita
- Dividida em células
- Cada célula armazena um único símbolo
- Símbolos pertencem a um alfabeto de entrada
- **Não** é possível gravar sobre a fita
- Palavra a ser processada ocupa toda a fita

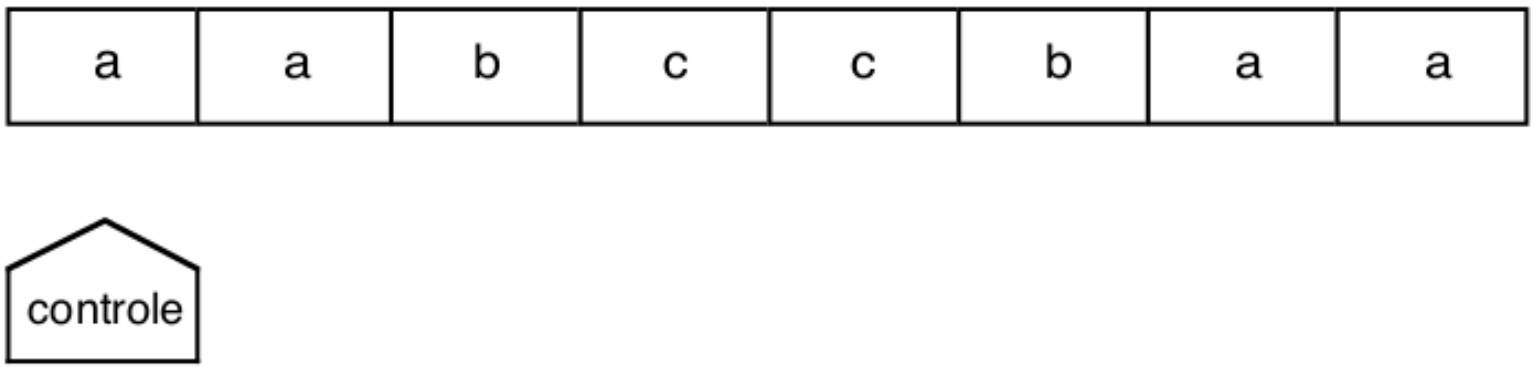
Unidade de Controle

Número finito e predefinido de estados

- origem do termo *controle finito*

Leitura

- lê o símbolo de uma célula por vez
- move a cabeça de leitura uma célula para a direita
- posição inicial: célula mais à esquerda da fita



Programa

Função Parcial

- dependendo do *estado corrente* e do *símbolo lido*
- determina o *novo estado* do autômato

Autômato Finito (Determinístico)

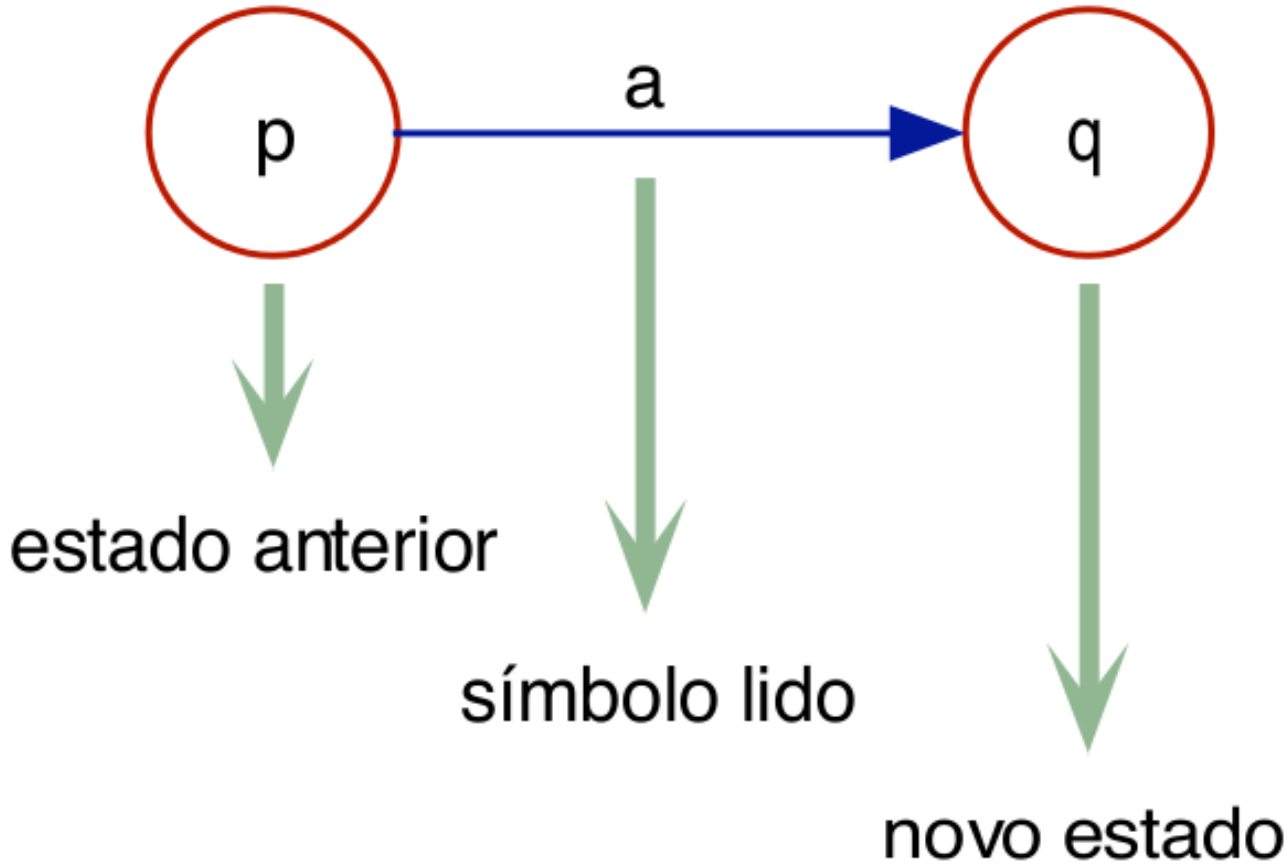
Definição (Autômato Finito Determinístico)

Um AFD é uma estrutura matemática $M = \langle \Sigma, Q, \delta, q_0, F \rangle$ tal que

- Σ é o alfabeto de entrada
- Q é um conjunto finito, denominado estados do autômato
- $\delta : Q \times \Sigma \rightarrow Q$ é a função de transição ou (função) programa
- $q_0 \in Q$ é o estado inicial do autômato
- $F \subseteq Q$ é o conjunto de estados finais

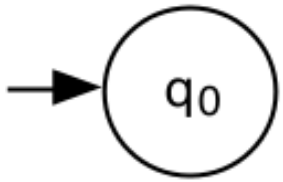
Transição

Se $\delta(p, a) = q$, podemos representar tal transição graficamente como

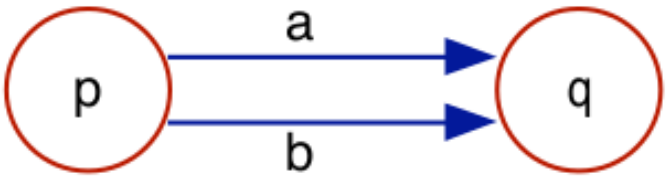


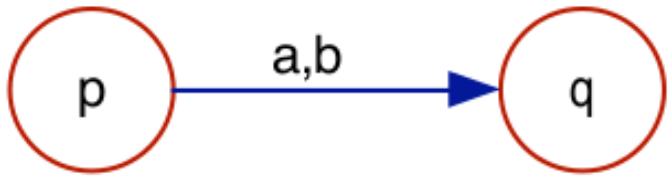
Representação Gráfica

Estados Iniciais e Finais



Transições paralelas: se $\delta(p, a) = q$ e $\delta(p, b) = q$





δ como Tabela

$$\delta(p, a) = q$$

δ	a	$\ldots$
p	q	$\ldots$
q	$\ldots$	$\ldots$

Computação

Sucessiva aplicação da função programa

- para cada símbolo de entrada
- muda-se o estado do autômato
- até ocorrer uma condição de parada

Lembrando que...

- autômato finito **não** possui memória
- estado será utilizado para guardar informações passadas

Exemplo

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

$$L_1 = \{w \mid w \text{ possui } aa \text{ ou } bb \text{ como subpalavra}\}$$

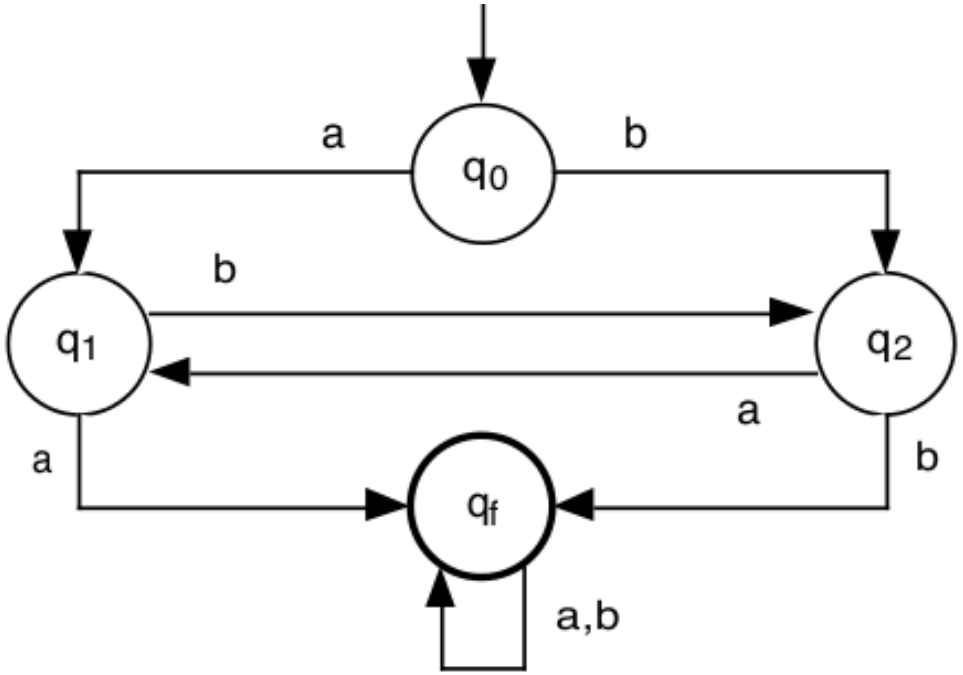
Autômato Finito

$$M_1 = \langle \{a, b\}, \{q_0, q_1, q_2, q_f\}, \delta_1, q_0, \{q_f\} \rangle$$

onde

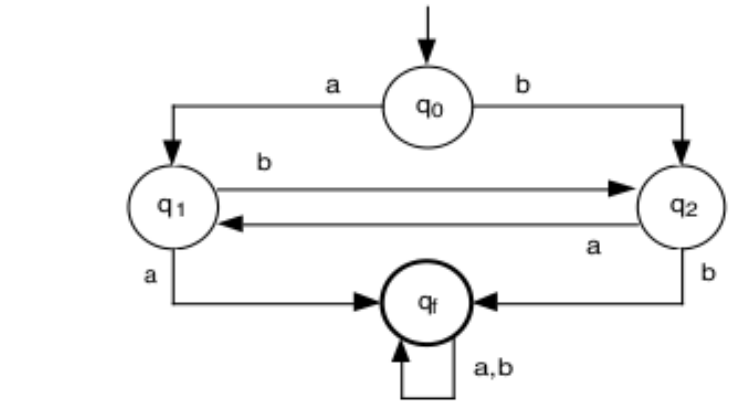
δ_1	a	b
q_0	q_1	q_2
q_1	q_f	q_2
q_2	q_1	q_f
q_f	q_f	q_f

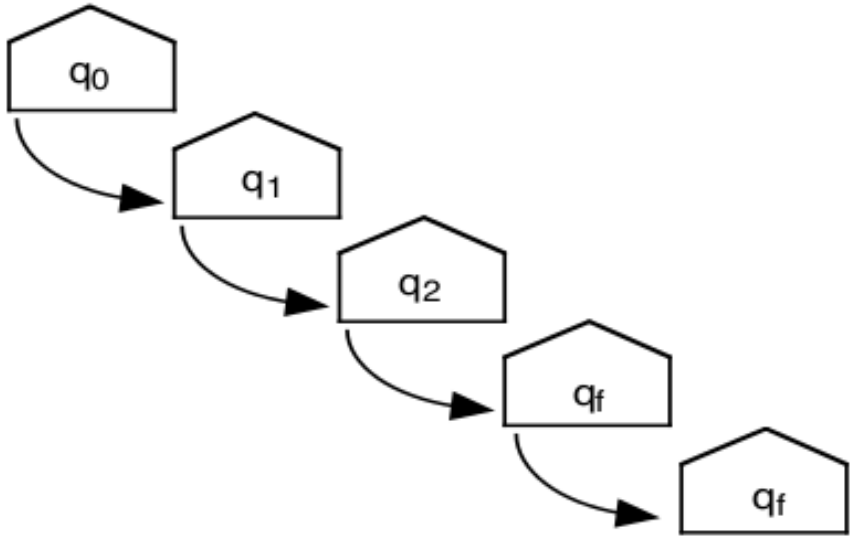
Exemplo



- q_1: símbolo anterior é a
- q_2: símbolo anterior é b
- qual a informação memorizada por q_0 e q_f?

Exemplo





Autômato Finito **sempre** para

- Toda palavra é finita
- Aplicação de δ: leitura de um símbolo de entrada
- Não há a possibilidade de ciclo infinito (*loop*)

Parada do Processamento

Aceitação

- Após processamento do último símbolo, assume um estado final

Rejeição

- Após processamento do último símbolo, assume um estado **não** final
- Programa não definido para argumento (estado e símbolo)

Comportamento de um Autômato Finito

Definição Formal?

- dar semântica à sintaxe
- extensão da função programa
- argumento: estado e *palavra*

Função Programa Estendida

Definição (Função Programa Estendida)

Dado $M = \langle \Sigma, Q, \delta, q_0, F \rangle$ um autômato finito determinístico, sua função programa estendida

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

é definida indutivamente como segue, sendo $q \in Q$, $a \in \Sigma$ e $w \in \Sigma^*$:

- $\delta^*(q, \varepsilon) = q$
- $\delta^*(q, aw) = \delta^*(\delta(q, a), w)$

Função Programa Estendida

Sucessiva aplicação da função programa

- se a entrada for vazia: fica parado
- aceitação ou rejeição: δ^* a partir do estado inicial

Definição (Linguagem Aceita/Rejeitada)

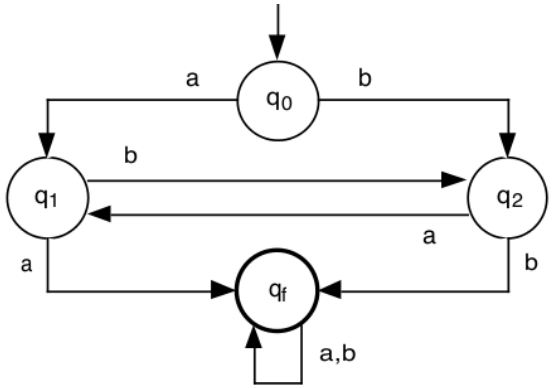
Dado $M = \langle \Sigma, Q, \delta, q_0, F \rangle$ um autômato finito determinístico, a **linguagem aceita** ou **linguagem reconhecida** por M é o conjunto

$$L(M) = \text{ ACEITA}(M) = \{w \mid \delta^*(q_0, w) \in F\}$$

e a **linguagem rejeitada** por M é

$$\text{REJEITA}(M) = \{w \mid \delta^*(q_0, w) \notin F \text{ ou } \delta^*(q_0, w) \text{ é indefinida } \}$$

Exemplo: Função Programa Estendida

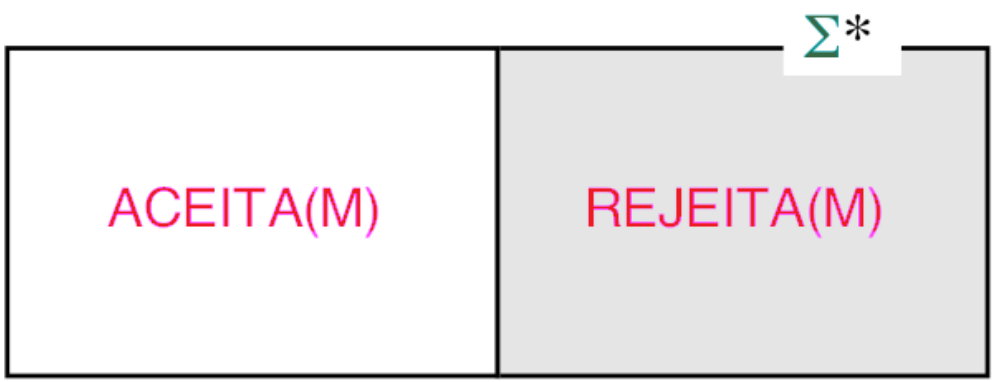


$$
\begin{aligned}
\delta^*(q_0, abba) &= \delta^*(\delta(q_0, a), baa) = \\
\delta^*(q_1, baa) &= \delta^*(\delta(q_1, b), aa) = \\
\delta^*(q_2, aa) &= \delta^*(\delta(q_2, a), a) = \\
\delta^*(q_1, a) &= \delta^*(\delta(q_1, a), \varepsilon) = \\
\delta^*(q_f, \varepsilon) &= q_f
\end{aligned}
$$

Partição de Σ^*

Supondo Σ^* o conjunto universo

- $\text{ACEITA}(M) \cap \text{REJEITA}(M) = \varnothing$
- $\text{ACEITA}(M) \cup \text{REJEITA}(M) = \Sigma^*$
- $\overline{\text{ACEITA}(M)} = \text{REJEITA}(M)$
- $\overline{\text{REJEITA}(M)} = \text{ACEITA}(M)$



Autômatos Equivalentes

Definição (Autômatos Finitos Equivalentes)

$M_1 = \langle \Sigma, Q_1, \delta_1, q_{0_1}, F_1 \rangle$ e $M_2 = \langle \Sigma, Q_2, \delta_2, q_{0_2}, F_2 \rangle$ são dois autômatos finitos equivalentes se e somente se

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

Linguagem Regular

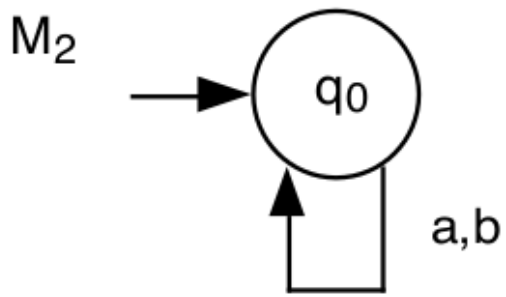
Definição (Linguagem Regular)

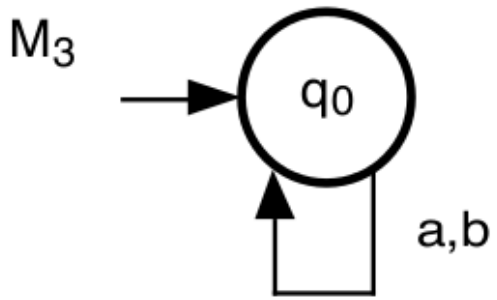
Uma linguagem $L \subseteq \Sigma^*$ é dita **Regular** ou Tipo 3 se existe pelo menos um autômato finito $M = \langle \Sigma, Q, \delta, q_0, F \rangle$ tal que $L(M) = L$.

Exemplo

Linguagem Vazia e Linguagem de Todas as Palavras sobre $\{a, b\}$

$$L_2 = \varnothing \text{ e } L_3 = \Sigma^*$$





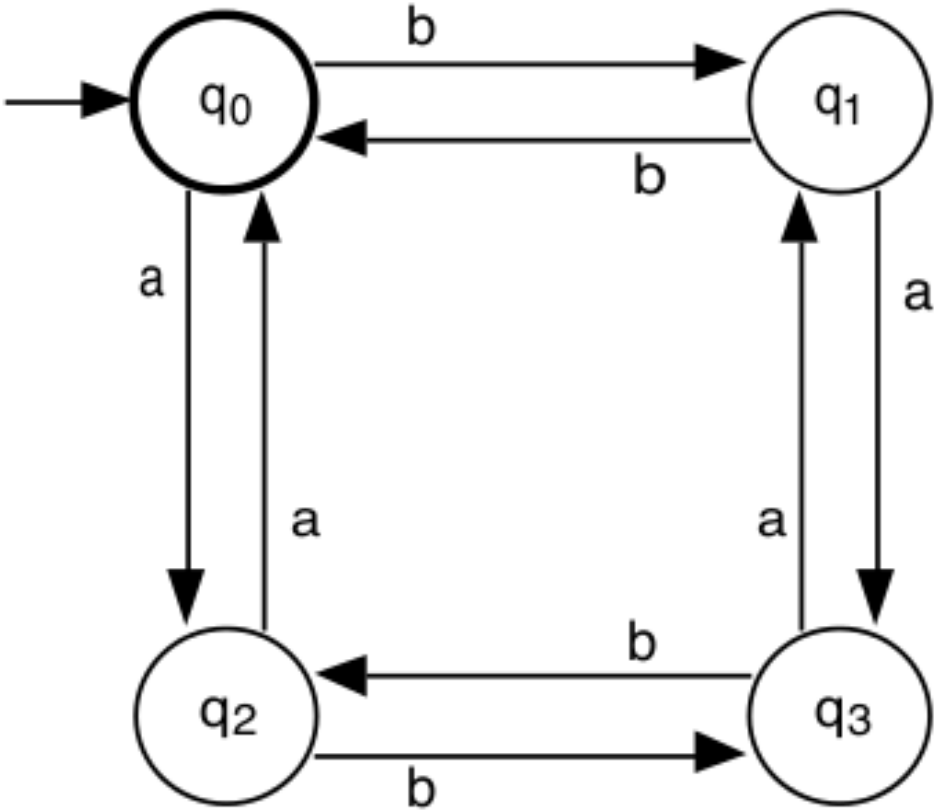
δ_2	a	b
q_0	q_0	q_0

δ_3	a	b
q_0	q_0	q_0

Exemplo

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

$L_4 = \{w \mid w \text{ possui um número par de } a \text{ e um número par de } b\}$



Exercícios

Para cada linguagem a seguir, todas sobre $\Sigma = \{a, b\}$, defina um autômato finito determinístico.

$L_{38} = \{w \mid w \text{ possui } aaa \text{ como subpalavra}\}$
$L_{42} = \{w \mid \text{o sufixo de } w \text{ é } ba\}$
$L_{731} = \{w \mid \text{o quarto símbolo da direita para a esquerda é } a\}$