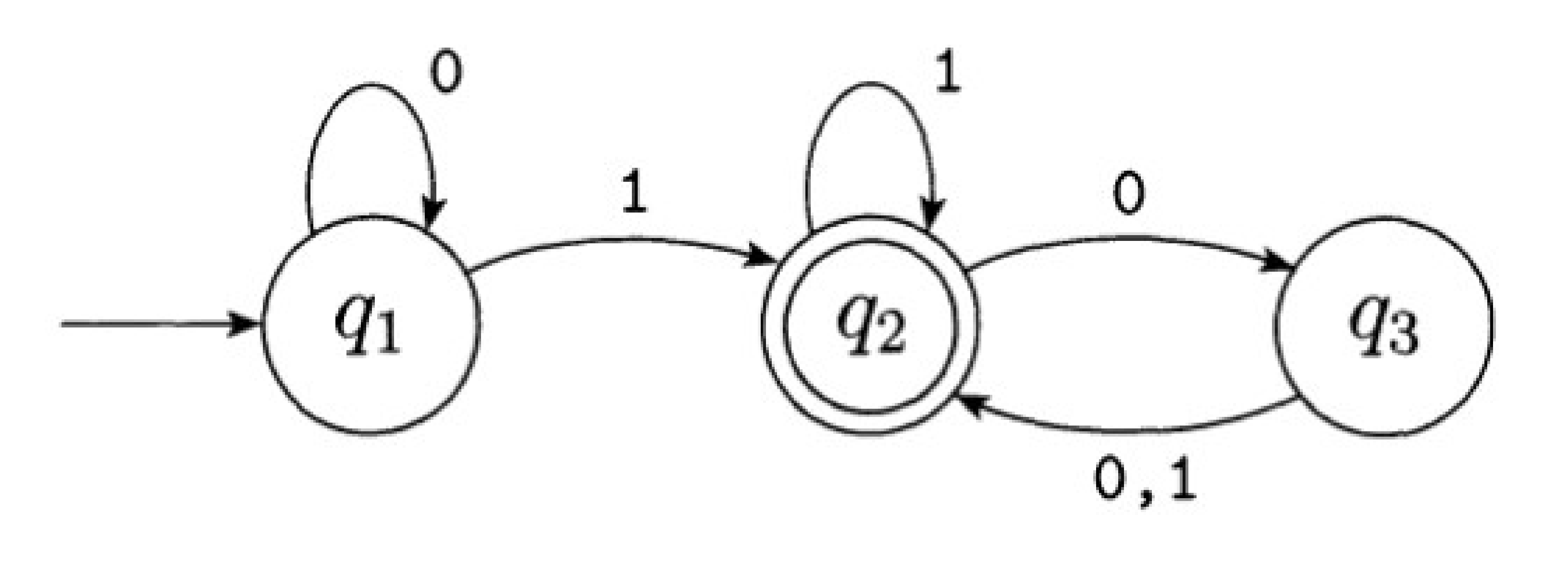
Teoria da Computação

Autômatos Finitos Determinísticos

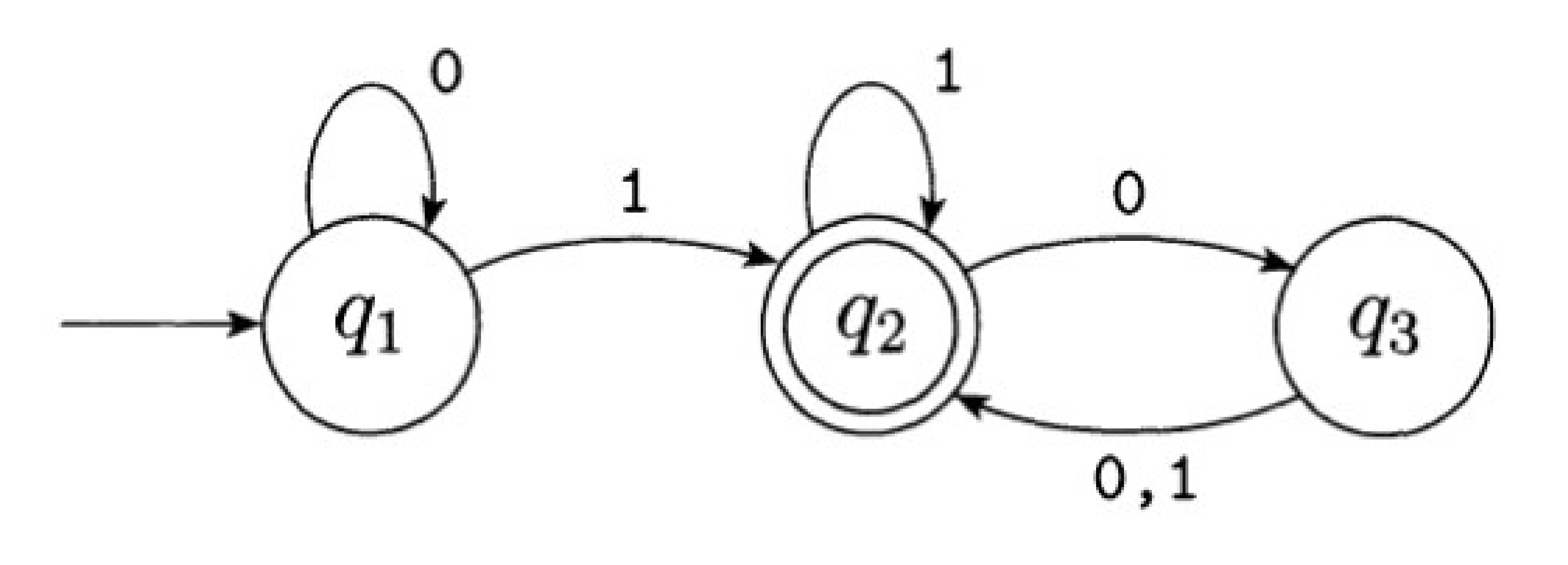
Prof. Thiago Alves

Introdução



- Três estados q_1, q_2 e q_3
- Estado inicial q_1 e final q_2
- Transições entre os estados

Introdução



- Diagrama de estados
- Processamento da string 1101
- Saída é **aceita** ou **rejeita**

Alfabeto

- Um *alfabeto* é qualquer conjunto finito de símbolos.
- Usamos o símbolo Σ
- Exemplos:

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

$\Sigma = \{a,b,c, \ldots, z\}$

Strings

- Uma *string* sobre um alfabeto Σ é uma sequência de símbolos de Σ.
  - 01101 é uma string do alfabeto Σ={0,1}.
- O *tamanho* de uma string é o número de símbolos de uma string.
  - w = 01101 é uma string de tamanho 5
  - Notação: |w| = 5
- ε indica a *string vazia*
  - |ε| = 0

Concatenação

- Se x e y são strings então xy denota a concatenação de x e y.
- Exemplo:
  - x = 01101 e y = 110
  - xy = 01101110
  - yx = 11001101
  - $\varepsilon w = w\varepsilon = w$

Strings

- y é substring de w quando w = xyz em que x e z são strings quaisquer.
- 001 é substring de 1001

- Σ^* é o conjunto de todas as strings sobre um alfabeto Σ.
- Com $\Sigma=\{0,1\}$, $\Sigma^* = \{\varepsilon, 0,1,00,01,\ldots\}$.

Linguagens

- Uma *linguagem* é um subconjunto de Σ^* para algum alfabeto Σ.
- Exemplo
  - O conjunto de strings de 0's e 1's sem dois 1's consecutivos.
- $L = \{\varepsilon, 0, 1, 00, 01, 10, 000, 001, 010, 100, 101, 0000, 0001, 0010, 0100, 0101, 1000, 1001, 1010, \ldots\}$

Linguagens

- $L_1 = \{w \in \{0,1\}^* \mid 010$ é substring de $w\}$
- Σ^*
- $\{\}$
- $\{\varepsilon\}$
- $L_c = \{w \in ASCII^* \mid w$ é um programa em $C\}$

Algoritmos para Linguagens

- $L_1 = \{w \in \{0,1\}^* \mid 010$ é substring de $w\}$
- Como seria um algoritmo para decidir se uma string pertence a L_1?

Algoritmos para Linguagens

- $L_1 = \{w \in \{0,1\}^* \mid 010$ é substring de $w\}$
- Como seria um algoritmo para decidir se uma string pertence a L_1?
  - Podemos usar estados para lembrar o que já foi visto na string

Autômato Finito Determinístico

- Autômato finito determinístico A_1

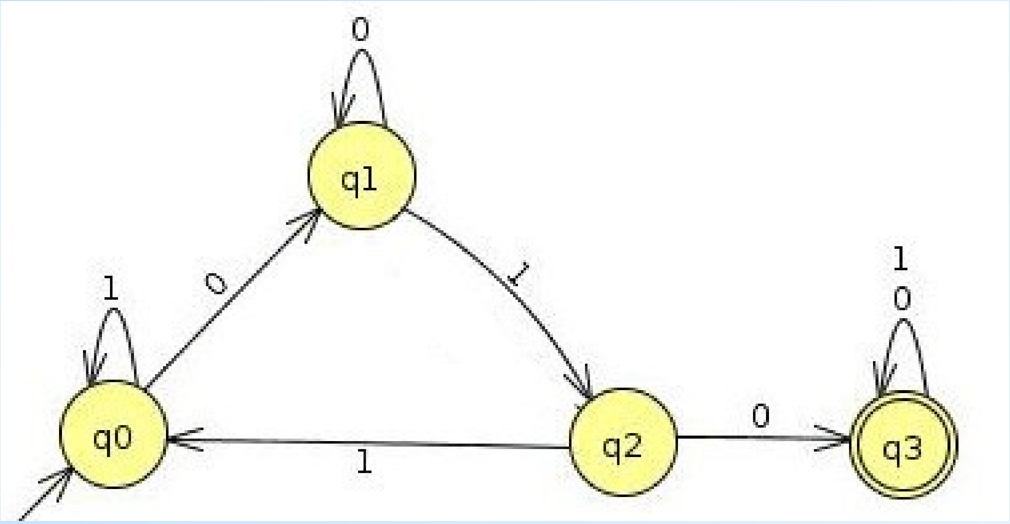


Diagrama de Transição de Estados

Autômato Finito Determinístico

- Basta processar a string de entrada de acordo com o autômato finito determinístico
  - Vamos processar a string 0110101
  - Vamos processar a string 101101

Autômato Finito Determinístico

- Basta processar a string de entrada de acordo com o autômato finito determinístico
  - Vamos processar a string 0110101
  - A_1 aceita a string
  - Vamos processar a string 101101
  - A_1 não aceita a string

Autômato Finito Determinístico

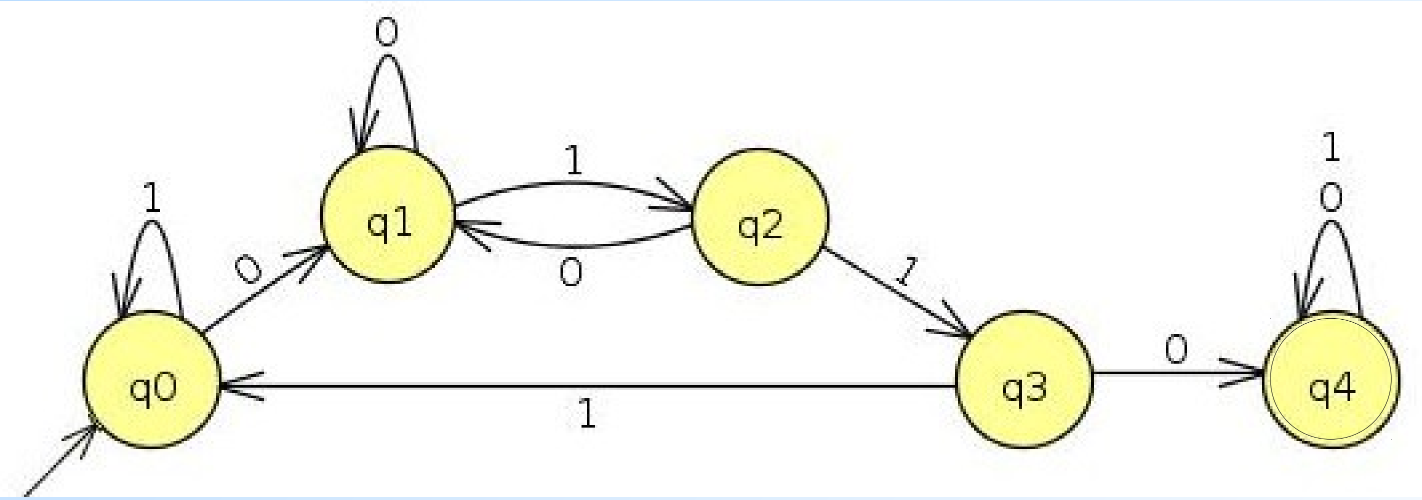
- Autômato Finito que está em apenas um estado em cada passo da execução
  - Para cada símbolo, só existe um estado para o qual o autômato possa ir a partir do estado atual
- Vamos utilizar o termo AFD

Autômato Finito Determinístico

- E para a linguagem
  - $L_2 = \{x0110y \mid x \in \{0,1\}^* \text{ e } y \in \{0,1\}^*\}$?

Autômato Finito Determinístico

- E para a linguagem
  - $L_2 = \{x0110y \mid x \in \{0,1\}^* \text{ e } y \in \{0,1\}^*\}$?
- Diagrama do AFD A_2:

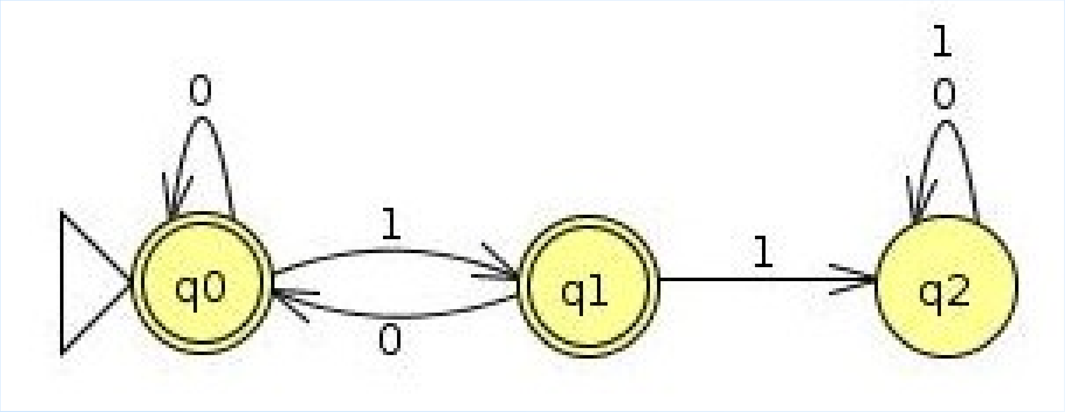


Autômato Finito Determinístico

- E para a linguagem
  - $L_3 = \{w \in \{0,1\}^* \mid \text{não ocorre 11 em } w\}$?

Autômato Finito Determinístico

- E para a linguagem
  - $L_3 = \{w \in \{0,1\}^* \mid \text{não ocorre 11 em } w\}$?
- Diagrama do AFD A_3:



Autômato Finito Determinístico

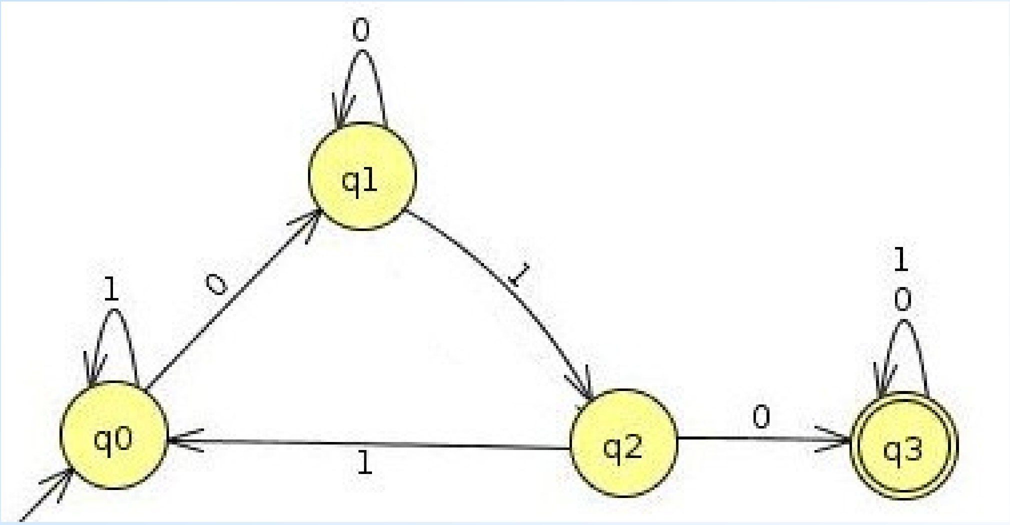
- Um AFD consiste de uma tupla $A = (Q, \Sigma, \delta, q_{0,} F)$:
  1. Um conjunto finito de *estados* Q
  2. Um *alfabeto de entrada* Σ
  3. Uma *função de transição* δ
  4. Um *estado inicial* q_0 in Q
  5. Um conjunto de *estados finais* $F \subseteq Q$
     - Também é usado estados de aceitação

A Função de Transição

- Recebe dois argumentos: um estado e um símbolo de entrada
- $\delta(q_i, a) = q_j$
  - O estado que o AFD vai quando está no estado *q* e um símbolo *a* é recebido
- $\delta: Q x \Sigma \rightarrow Q$
- Nota: sempre existe um próximo estado

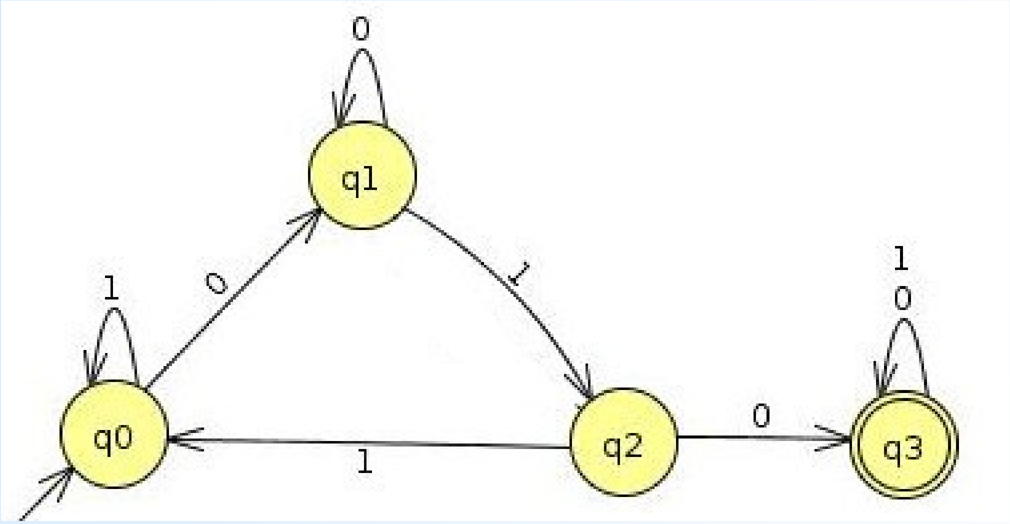
Exemplo

- $A_1 = (Q, \Sigma, \delta, q_0, F)$
- $Q = \{q0,q1,q2,q3\}$
- $\Sigma = \{0,1\}$
- $q_0 = q0$
- $F = \{q3\}$



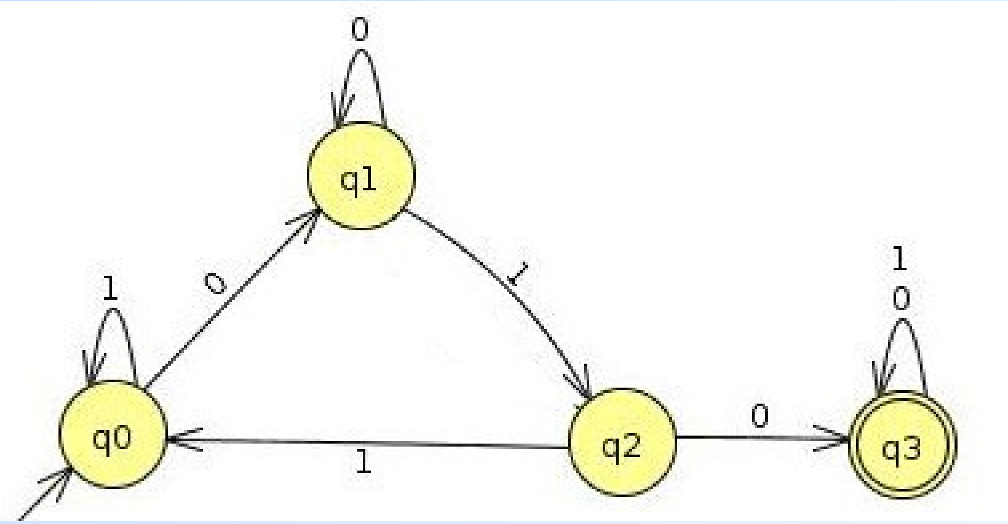
Exemplo

- $\delta = \{(q0,0,q1),(q0,1,q0),$
- $(q1,0,q1),(q1,1,q2)$
- $(q2,0,q3),(q2,1,q0)$
- $(q3,0,q3),(q3,1,q3)\}$



Exemplo

Estado/Símbolo	0	1
→q0	q1	q0
q1	q1	q2
q2	q3	q0
*q3	q3	q3



Função de Transição Estendida

- Vamos formalizar a definição de processamento de uma string por um AFD
- Podemos obter um algoritmo a partir da formalização
- Vamos estender a função de transição para poder receber strings como entrada

Função de Transição Estendida

- Vamos chamar a função estendida de δ de δ^*
- Vamos definir δ^* indutivamente na string de entrada
- $\delta^*(q, \varepsilon) = ?$
- $\delta^*(q,ua) = ?$
  - Nota: 'u' é uma string e 'a' é um símbolo pela nossa convenção

Função de Transição Estendida

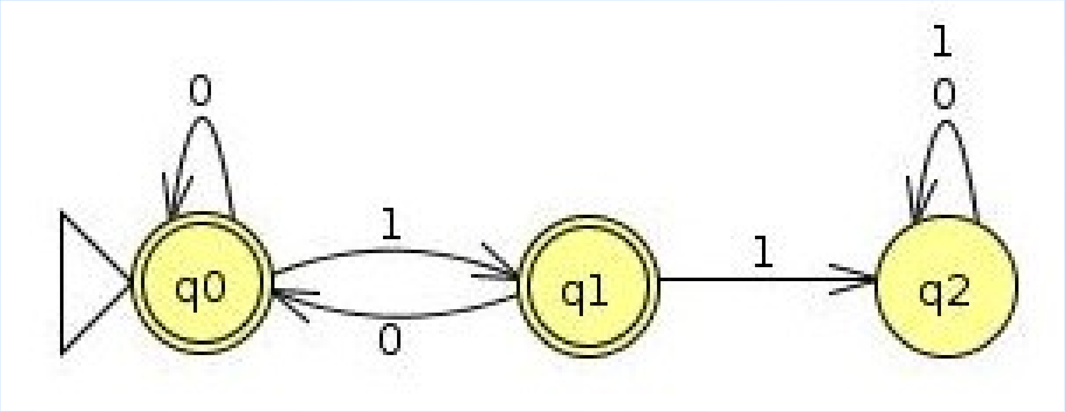
- $\delta^*(q, \varepsilon) = q$
- $\delta^*(q,ua) = ?$
  - Computar $\delta^*(q,u)$
  - Resultado é um estado que vamos chamar de p
  - Agora computar $\delta(p, a)$

Função de Transição Estendida

- Seja q um estado
- $\delta^*(q, \varepsilon) = q$
- $\delta^*(q,ua) = \delta(\delta^*(q,u),a)$

Exemplo

- Diagrama do AFD A_3



- Computar $\delta^*(q_0, 1010)$
- Computar $\delta^*(q_0, 1101)$

Aceitação de String

- Seja $A = (Q, \Sigma, \delta, q_{0,}$ F) um AFD
- A aceita uma string w se $\delta^*(q_0, w) \in F$
- Quando $\delta^*(q_0, w) \notin F$ dizemos que A rejeita w
  - Também é usado o termo A não aceita w

Exemplo

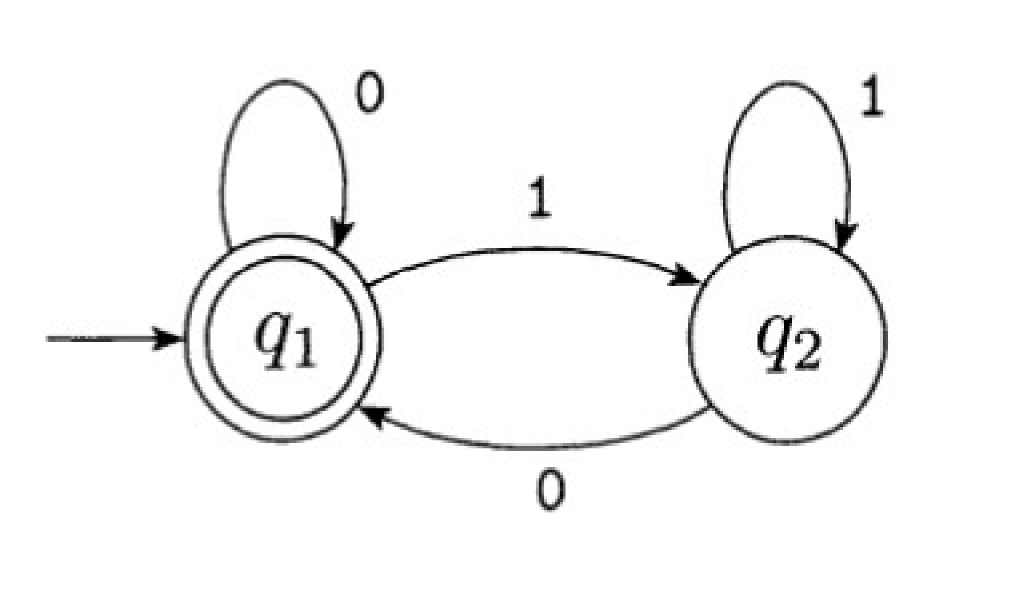
- A_3 aceita 1010 pois $\delta^*(q_0, 1010) \in F$

- A_3 não aceita 1101 pois $\delta^*(q_0, 1101) \notin F$

Linguagem de um AFD

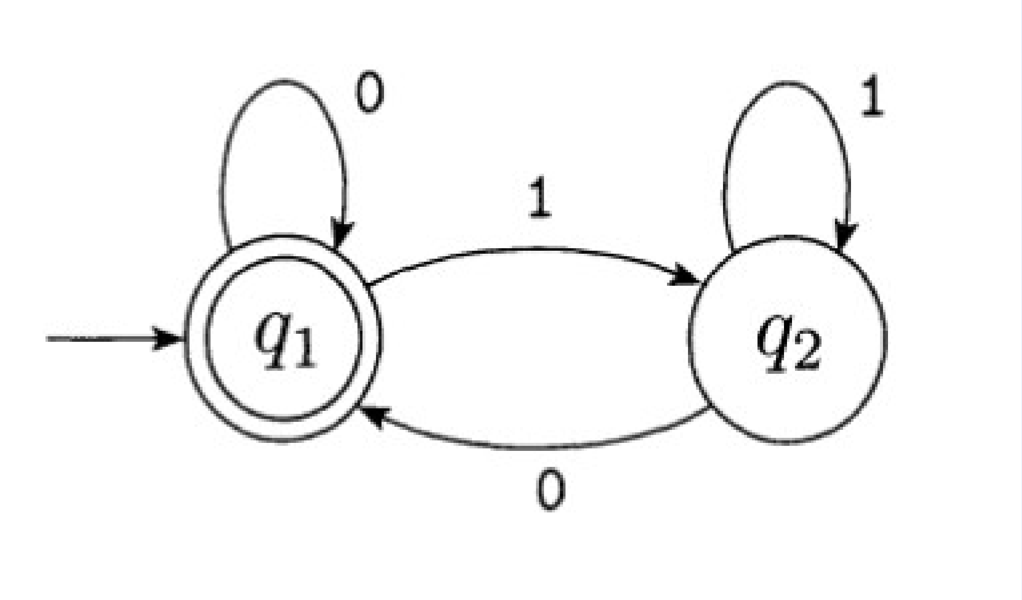
- Autômatos finitos determinísticos definem linguagens
- Se A é um AFD então L(A) é a linguagem de A
- $L(A) = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in F\}$

Exemplo



L(A) = ?

Exemplo



L(A) = {w ∈ {0,1}* | w = ε ou w termina em 0}

Linguagens Regulares

- Se B é uma linguagem e A é um AFD tal que L(A) = B então dizemos que A aceita B
  - Também é usado A reconhece B
- Uma linguagem B é regular se existe um AFD A que aceita B

Exemplo

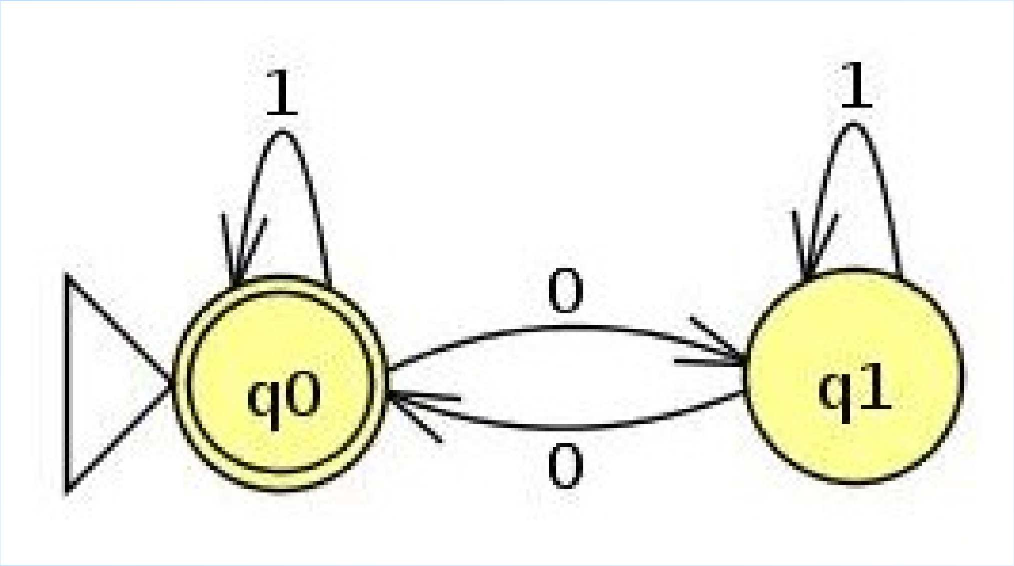
- $L_1 = \{w \in \{0,1\}^* \mid 010$ é substring de $w\}$ é regular
- $L_2 = \{x0110y \mid x \in \{0,1\}^*$ e $y \in \{0,1\}^*\}$ é regular
- $L_3 = \{w \in \{0,1\}^* \mid w$ não possui $11\}$ é regular

Exemplo

- Vamos mostrar que $L_4 = \{w \in \{0,1\}^* \mid w$ tem quantidade par de 0's$\}$ é regular.

Exemplo

- $L_4 = \{w \in \{0,1\}^* \mid w$ tem quantidade par de 0's$\}$ é regular.



Strings

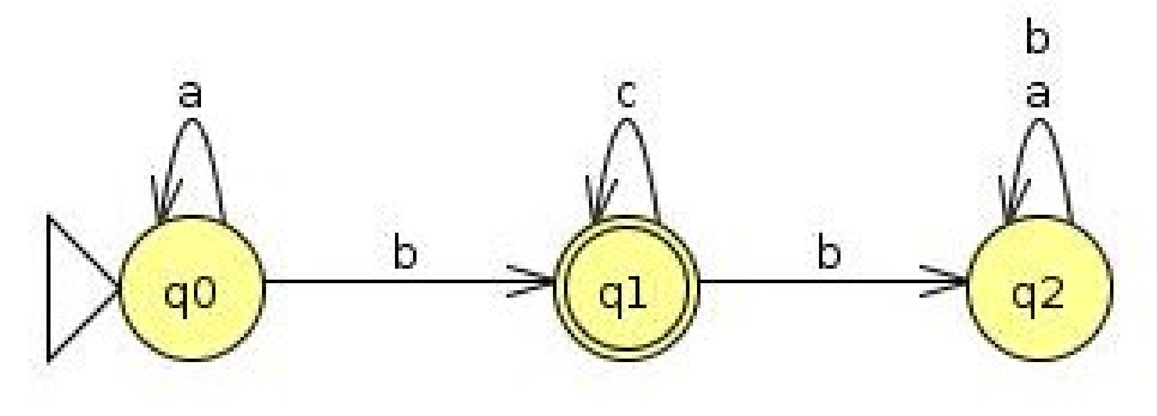
- Para concatenar uma string com ela mesma várias vezes usamos a notação w^n
- Vamos definir w^n de forma indutiva
  - $w^0 = \varepsilon$
  - $w^{i+1} = w^i w$
- Exemplo
  - Se $w = 010$ então $w^2 = 010010$

Exemplo

- Mostre que $L_b = \{a^n b c^m \mid n,m \geq 0\}$ é regular.

Exemplo

- Mostre que $L_b = \{a^n b c^m \mid n,m \geq 0\}$ é regular.



Strings

- O reverso de w, notação w^r, é a string obtida escrevendo w na ordem inversa
- Também vamos definir indutivamente
  - $\varepsilon^r = \varepsilon$
  - Se $w = ua$ para algum $a \in \Sigma$ então $w^r = au^r$
- Exemplo
  - $(110)^r = 0(11)^r = 011^r = 011\varepsilon^r = 011$

Exemplo

- $(110001)^r = ?$
- $(001)^r(110)^r = ?$

Strings

- Teorema: Para quaisquer strings x e w temos que $(wx)^r = x^r w^r$
- Prova por indução em x
- Base: $|x| = 0$
  - $x = \varepsilon$
  - $(w\varepsilon)^r = \varepsilon w^r = \varepsilon^r w^r$
- Hipótese de Indução: para quaisquer x tal que $0 \leq |x| \leq n$ temos que $(wx)^r = x^r w^r$

Strings

- Teorema: Para quaisquer strings x e w temos que $(wx)^r = x^r w^r$
- Prova por indução em x
- Passo de Indução: seja x qualquer tal que $|x| = n+1$
  - $x = ua$ tal que $|u| = n$
  - Pela definição: $(wua)^r = a(wu)^r$
  - Pela hipótese de indução: $a(wu)^r = au^r w^r$
  - Pela definição: $au^r w^r = (ua)^r w^r$

Complemento de Linguagem Regular

- Mostre que se A é um AFD, então trocando os estados finais pelos não-finais em A obtemos um novo AFD A' tal que L(A') $= \overline{L(A)}$.

Complemento de Linguagem Regular

- Prove que se B é uma linguagem regular então $\overline{B}$ é regular.