

# Teoria da Computação

Prof. Maicon R. Zatelli

## Aula 2 - Linguagens Regulares

Universidade Federal de Santa Catarina  
Florianópolis - Brasil

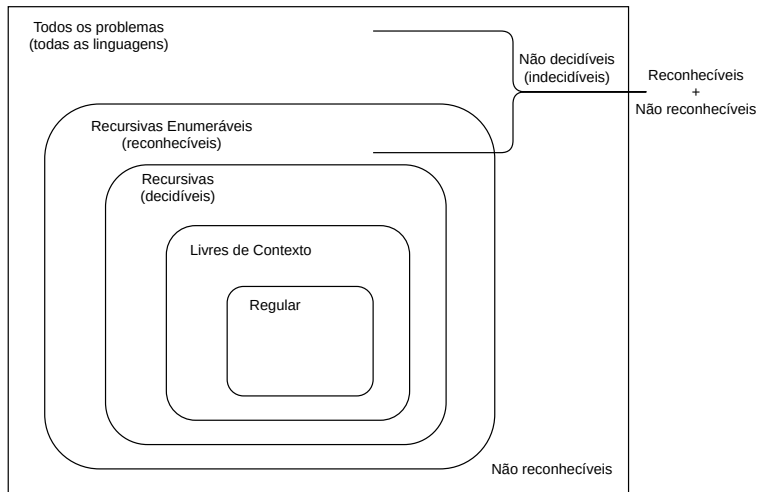
# Introdução

- Linguagens regulares
- Autômatos finitos determinísticos
- Autômatos finitos (determinísticos e não-determinísticos)
- Expressões regulares
- Linguagens não regulares e lema do bombeamento

## Material de apoio

- Livro Sipser, Capítulo 1
- Livro Hopcroft, Capítulo 2,3,4

# Hierarquia de Chomsky



Indecidíveis - são todas as linguagens Turing-reconhecíveis mas não decidíveis e também as linguagens não Turing-reconhecíveis

# Linguagens Regulares

São as linguagens mais simples na Hierarquia de Chomsky.

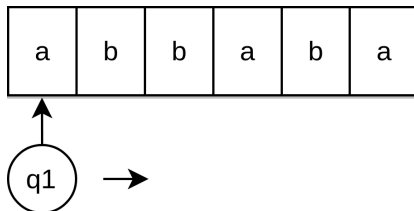
## Exemplos de Uso

- Analisador Léxico
- Verificar se tamanho de palavra é ímpar/par
- Verificar se a quantidade de dígitos 1 em uma palavra é ímpar/par
- Pesquisar por ocorrência de um padrão em uma palavra

# Autômatos Finitos

Autômato Finito (AF) ou Máquina de Estados Finita são um modelo computacional com uma quantidade extremamente limitada de memória. São reconhecedores (ou aceitadores) da classe de linguagens regulares.

- Não possuem nenhuma memória auxiliar (contador, pilha, etc)
- Apenas armazena o estado atual



Autômatos Finitos podem ser determinísticos (AFD) ou não-determinísticos (AFND).

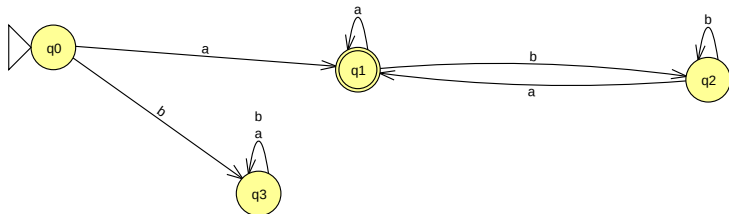
# Autômatos Finitos Determinísticos

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

- $Q$  é um conjunto finito de estados
- $\Sigma$  é um conjunto finito de símbolos (alfabeto)
- $\delta : Q \times \Sigma \rightarrow Q$  é uma função de transição
  - $\delta(\text{estado atual}, \text{símbolo lido}) \rightarrow \text{novo estado}$
- $q_0 \in Q$  é um estado inicial
- $F \subseteq Q$  é um conjunto de estados finais

# Autômatos Finitos Determinísticos - Diagrama de Estados

$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$



# Autômatos Finitos Determinísticos - Descrição Formal

$$L = \{w | w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

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

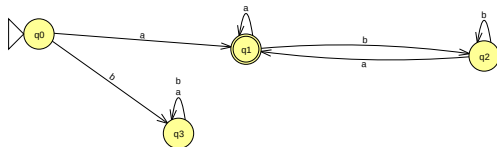
- $Q = \{q_0, q_1, q_2, q_3\}$
- $\Sigma = \{a, b\}$
- $\delta : Q \times \Sigma \rightarrow Q$ 
  - $\delta(q_0, a) \rightarrow q_1, \delta(q_0, b) \rightarrow q_3,$
  - $\delta(q_1, a) \rightarrow q_1, \delta(q_1, b) \rightarrow q_2,$
  - $\delta(q_2, a) \rightarrow q_1, \delta(q_2, b) \rightarrow q_2,$
  - $\delta(q_3, a) \rightarrow q_3, \delta(q_3, b) \rightarrow q_3$
- $q_0 = q_0$
- $F = \{q_1\}$



# Autômatos Finitos Determinísticos - Representação Tabular

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

	a	b
$\rightarrow$ q0	q1	q3
* q1	q1	q2
q2	q1	q2
q3	q3	q3

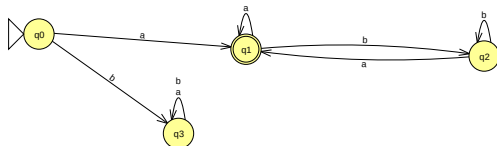


# Autômatos Finitos Determinísticos - Execução

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

Tome como exemplo a palavra: **ababa**

a	b	a	b	a	
↑					
q0					

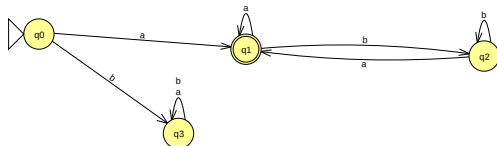


# Autômatos Finitos Determinísticos - Execução

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

Tome como exemplo a palavra: **ababa**

a	b	a	b	a	
	↑				
	q1				

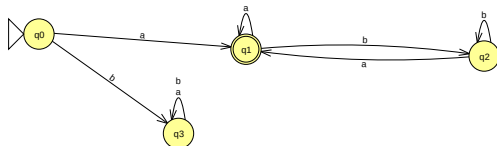


# Autômatos Finitos Determinísticos - Execução

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

Tome como exemplo a palavra: **ababa**

a	b	a	b	a	
		↑			
		q2			

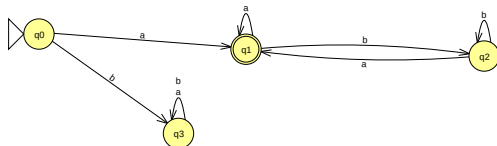


# Autômatos Finitos Determinísticos - Execução

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

Tome como exemplo a palavra: **ababa**

a	b	a	b	a	
			↑		
			q1		

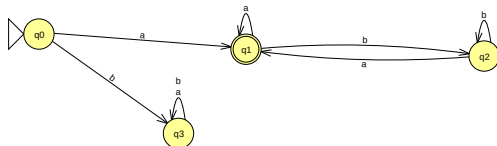


# Autômatos Finitos Determinísticos - Execução

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

Tome como exemplo a palavra: **ababa**

a	b	a	b	a	
				↑	
				q2	

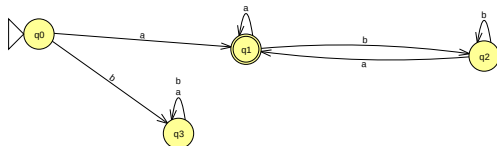


# Autômatos Finitos Determinísticos - Execução

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w \text{ começa e termina com } a\}$$

Tome como exemplo a palavra: **ababa**

a	b	a	b	a	
					↑
					q1 ✓



# Autômatos Finitos Determinísticos - Execução

- Cabeça somente move-se para direita e troca estado
- Não há memória auxiliar
- Em um AFD deve haver uma transição partindo de cada estado para cada símbolo
- Uma **computação** em um AFD  $M = (Q, \Sigma, \delta, q_0, F)$  sobre uma palavra  $w = w_1 w_2 w_3 \dots w_n$ , onde  $w_i$  é um membro do alfabeto  $\Sigma$ , é uma sequência de estados  $r_0, r_1, \dots, r_n$ , tal que  $r_0$  é igual ao estado inicial de  $M$  e  $\delta(r_i, w_{i+1}) \rightarrow r_{i+1}$  para  $i = 0$  até  $n - 1$ . A computação **aceita**  $w$  se  $r_n \in F$  e **rejeita**  $w$  se  $r_n \notin F$
- Dizemos que  $M$  reconhece uma linguagem  $A$  se  $A = \{w \mid M \text{ aceita } w\}$
- Uma linguagem  $L$  é dita linguagem regular se algum autômato finito  $M$  a reconhece, ou seja,  $L = L(M)$ , onde  $L(M)$  é a linguagem reconhecida por  $M$ .



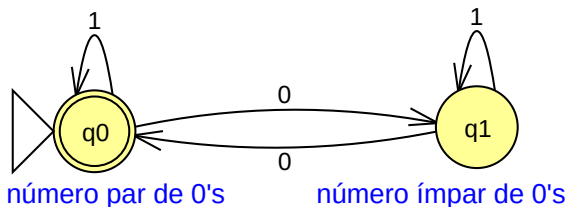
# Linguagem Regular vs Autômatos Finitos Determinísticos

Como mostrar que uma linguagem é regular?

- R: construindo um AFD que a reconheça

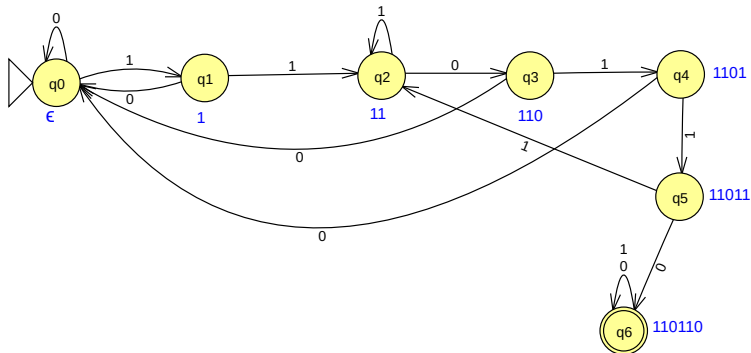
# Autômatos Finitos Determinísticos

$L = \{w \mid w \in \{0, 1\}^* \text{ e } w \text{ contém um número par de 0's}\}$



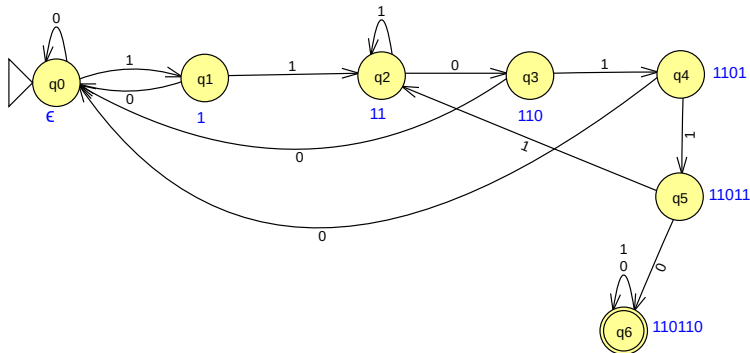
# Autômatos Finitos Determinísticos

$$L = \{w \mid w \in \{0, 1\}^* \text{ e } w \text{ contém } 110110\}$$



# Autômatos Finitos Determinísticos

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w \text{ contém } 110110\}$$



E se quero a linguagem cujas palavras não contém 110110?

- O complemento de  $L$ , escrito como  $\bar{L}$ , são as palavras que não pertencem a  $L$ .

## Linguagens Regulares - Operações - Complemento

O **complemento** é uma **operação fechada** sobre a classe das linguagens regulares, ou seja, se  $L$  é uma linguagem regular, então  $\overline{L}$  também é uma linguagem regular, e vice-versa.

- $L \subseteq \Sigma^*$
- $\overline{L} = \{w \mid w \in \Sigma^* \text{ e } w \notin L\}$ , ou seja,  $\Sigma^* - L$

## Linguagens Regulares - Operações - Complemento

**Prova:** por construção.

## Linguagens Regulares - Operações - Complemento

**Prova:** por construção.

- Se  $L$  é uma linguagem regular, então existe um AFD  $M$  que a reconhece.

## Linguagens Regulares - Operações - Complemento

**Prova:** por construção.

- Se  $L$  é uma linguagem regular, então existe um AFD  $M$  que a reconhece.
- Note agora que o complemento de  $L$ ,  $\bar{L}$ , contém todas as palavras não reconhecidas por  $M$ , ou seja, todas as palavras que ao serem dadas como entrada de  $M$  têm sua computação terminada em algum estado de  $M$  que não seja final.



# Linguagens Regulares - Operações - Complemento

**Prova:** por construção.

- Se  $L$  é uma linguagem regular, então existe um AFD  $M$  que a reconhece.
- Note agora que o complemento de  $L$ ,  $\bar{L}$ , contém todas as palavras não reconhecidas por  $M$ , ou seja, todas as palavras que ao serem dadas como entrada de  $M$  têm sua computação terminada em algum estado de  $M$  que não seja final.
- Ao inverter os estados finais e não finais de  $M$  temos um AFD  $N$  que reconhece exatamente a linguagem  $\bar{L}$ , ou seja:
  - $M = (Q, \Sigma, \delta, q_0, F)$
  - $N = (Q, \Sigma, \delta, q_0, F')$ , onde  $F' = Q - F$

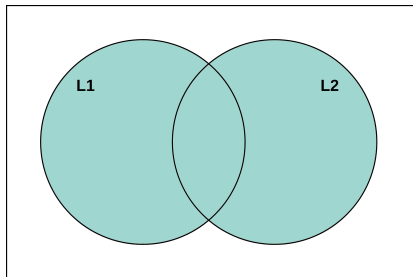
# Linguagens Regulares - Operações - Complemento

**Prova:** por construção.

- Se  $L$  é uma linguagem regular, então existe um AFD  $M$  que a reconhece.
- Note agora que o complemento de  $L$ ,  $\bar{L}$ , contém todas as palavras não reconhecidas por  $M$ , ou seja, todas as palavras que ao serem dadas como entrada de  $M$  têm sua computação terminada em algum estado de  $M$  que não seja final.
- Ao inverter os estados finais e não finais de  $M$  temos um AFD  $N$  que reconhece exatamente a linguagem  $\bar{L}$ , ou seja:
  - $M = (Q, \Sigma, \delta, q_0, F)$
  - $N = (Q, \Sigma, \delta, q_0, F')$ , onde  $F' = Q - F$
- Como temos um AFD que reconhece  $\bar{L}$ , então sabemos que ela também é uma linguagem regular.

## Linguagens Regulares - Operações - União

Seja  $L1, L2 \subseteq \Sigma^*$  e  $L1$  e  $L2$  sendo regulares, então  $L = L1 \cup L2$  também é uma linguagem regular. A classe das linguagens regulares é fechada na operação de união.



# Linguagens Regulares - Operações - União

Intuição para a prova:

- Sendo que  $L1$  e  $L2$  são linguagens regulares, existem os AFDs  $M1$  que reconhece  $L1$  e  $M2$  que reconhece  $L2$
- Pode-se rodar  $M1$  e  $M2$  em paralelo para uma mesma entrada  $w$
- $\delta((q, r), x) = (\delta_1(q, x), \delta_2(r, x))$ , onde  $q$  é um estado de  $M1$ ,  $r$  é um estado de  $M2$ ,  $x$  é o símbolo lido,  $\delta_1$  é a função de transição de  $M1$  e  $\delta_2$  é a função de transição de  $M2$
- Se  $M1$  ou  $M2$  aceitar  $w$ , então  $w \in L$ , onde  $L = L1 \cup L2$

## Linguagens Regulares - Operações - União

**Prova:** por construção.

# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$

# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cup L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde

# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cup L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$



# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cup L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$

# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cup L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$

# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cup L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$
  - $F_3 = \{(q_1, q_2) | q_1 \in F_1 \text{ ou } q_2 \in F_2\}$

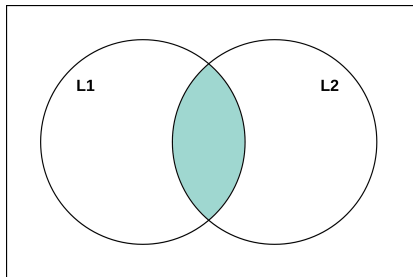
# Linguagens Regulares - Operações - União

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cup L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$
  - $F_3 = \{(q_1, q_2) | q_1 \in F_1 \text{ ou } q_2 \in F_2\}$
- Note que  $L(M3) = L1 \cup L2$

# Linguagens Regulares - Operações - Intersecção

Seja  $L1, L2 \subseteq \Sigma^*$  e  $L1$  e  $L2$  sendo regulares, então  $L = L1 \cap L2$  também é uma linguagem regular. A classe das linguagens regulares é fechada na operação de intersecção.



# Linguagens Regulares - Operações - Intersecção

Intuição para a prova:

- Sendo que  $L1$  e  $L2$  são linguagens regulares, existem os AFDs  $M1$  que reconhece  $L1$  e  $M2$  que reconhece  $L2$
- Pode-se rodar  $M1$  e  $M2$  em paralelo para uma mesma entrada  $w$
- $\delta((q, r), x) = (\delta_1(q, x), \delta_2(r, x))$ , onde  $q$  é um estado de  $M1$ ,  $r$  é um estado de  $M2$ ,  $x$  é o símbolo lido,  $\delta_1$  é a função de transição de  $M1$  e  $\delta_2$  é a função de transição de  $M2$
- Se  $M1$  e  $M2$  aceitar  $w$ , então  $w \in L$ , onde  $L = L1 \cap L2$

## Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$



# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cap L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde

# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cap L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$

# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cap L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$

# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cap L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$

# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cap L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$
  - $F_3 = \{(q_1, q_2) | q_1 \in F_1 \text{ e } q_2 \in F_2\}$

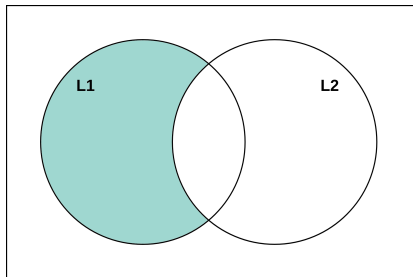
# Linguagens Regulares - Operações - Intersecção

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \cap L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$
  - $F_3 = \{(q_1, q_2) | q_1 \in F_1 \text{ e } q_2 \in F_2\}$
- Note que  $L(M3) = L1 \cap L2$

## Linguagens Regulares - Operações - Diferença

Seja  $L1, L2 \subseteq \Sigma^*$  e  $L1$  e  $L2$  sendo regulares, então  $L = L1 - L2$  também é uma linguagem regular. A classe das linguagens regulares é fechada na operação de diferença.



## Linguagens Regulares - Operações - Diferença

Intuição para a prova:

- Sendo que  $L1$  e  $L2$  são linguagens regulares, existem os AFDs  $M1$  que reconhece  $L1$  e  $M2$  que reconhece  $L2$
- Pode-se rodar  $M1$  e  $M2$  em paralelo para uma mesma entrada  $w$
- $\delta((q, r), x) = (\delta_1(q, x), \delta_2(r, x))$ , onde  $q$  é um estado de  $M1$ ,  $r$  é um estado de  $M2$ ,  $x$  é o símbolo lido,  $\delta_1$  é a função de transição de  $M1$  e  $\delta_2$  é a função de transição de  $M2$
- Se  $M1$  aceitar  $w$  e  $M2$  rejeitar  $w$ , então  $w \in L$ , onde  $L = L1 - L2$



## Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

## Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$

# Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 - L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde

## Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 - L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$

# Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 - L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$

# Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 - L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$

# Linguagens Regulares - Operações - Diferença

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 - L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$
  - $F_3 = \{(q_1, q_2) | q_1 \in F_1 \text{ e } q_2 \notin F_2\}$

# Linguagens Regulares - Operações - Diferença

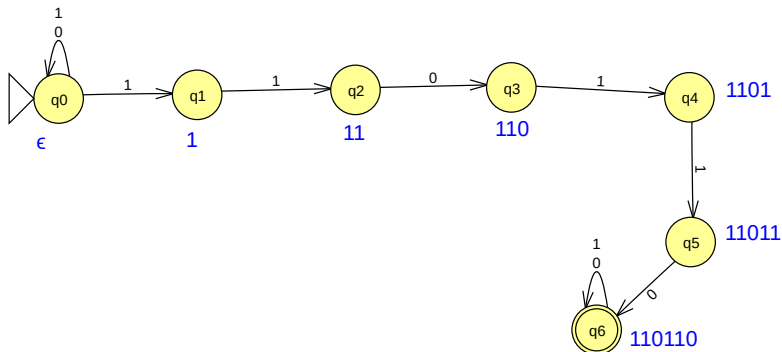
**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
  - $M1 = (Q_1, \Sigma, \delta_1, q0_1, F_1)$
  - $M2 = (Q_2, \Sigma, \delta_2, q0_2, F_2)$
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 - L2$ 
  - $M3 = (Q_3, \Sigma, \delta_3, q0_3, F_3)$ , onde
  - $Q_3 = Q_1 \times Q_2$ , ou seja,  $Q_3 = \{(q_1, q_2) | q_1 \in Q_1 \text{ e } q_2 \in Q_2\}$
  - $q0_3 = (q0_1, q0_2)$
  - $\delta_3((q_1, q_2), x) = (\delta_1(q_1, x), \delta_2(q_2, x))$ , onde  $q_1 \in Q_1$  e  $q_2 \in Q_2$
  - $F_3 = \{(q_1, q_2) | q_1 \in F_1 \text{ e } q_2 \notin F_2\}$
- Note que  $L(M3) = L1 - L2$



# Autômatos Finitos Não-Determinísticos (AFND)

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w \text{ contém } 110110\}$$



# Autômatos Finitos Não-Determinísticos (AFND)

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

- $Q$  é um conjunto finito de estados
- $\Sigma$  é um conjunto finito de símbolos (alfabeto)
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow P(Q)$  é uma função de transição, onde  $P(Q)$  é o conjunto de partes de  $Q$ 
  - $\delta(\text{estado atual}, \text{símbolo lido ou } \varepsilon) \rightarrow \text{conjunto de novos estados}$
- $q_0 \in Q$  é um estado inicial
- $F \subseteq Q$  é um conjunto de estados finais

# AFND - Execução

- Cabeça somente move-se para direita e troca estado
- Não há memória auxiliar
- Em um AFND podem haver várias transições com o mesmo símbolo a partir de um mesmo estado e para estados diferentes
- Algumas transições podem ser disparadas sem consumir nenhum símbolo da entrada, ou seja, elas podem ser transições com  $\varepsilon$ . Neste caso os AFND são chamados de  $\varepsilon$ -AFND
- Uma **computação** em um AFND (sem transições  $\varepsilon$ )  
 $M = (Q, \Sigma, \delta, q_0, F)$  sobre uma palavra  $w = w_1 w_2 w_3 \dots w_n$ , onde  $w_i$  é um membro do alfabeto  $\Sigma$ , é uma sequência de estados  $r_0, r_1, \dots, r_n$ , tal que  $r_0$  é igual ao estado inicial de  $M$  e  $r_{i+1} \in \delta(r_i, w_{i+1})$  para  $i = 0$  até  $n - 1$ . A computação **aceita**  $w$  se algum  $r_n \in F$  e **rejeita**  $w$  se todos  $r_n \notin F$
- Note que  $\delta(r_i, w_{i+1})$  é o conjunto de todos os estados possíveis de serem alcançados a partir do estado  $r_i$  com o símbolo  $w_{i+1}$

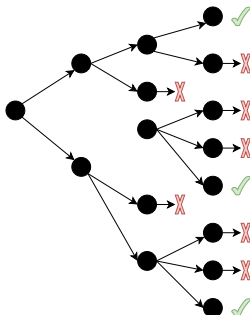
# AFND - Execução - Intuição

- Faz escolha de qual estado será o próximo a cada estado. Se não der certo com uma escolha, tenta outra. O AFND tenta todas as escolhas "ao mesmo tempo".
- O AFND aceita uma entrada se alguma escolha levar até um estado final
- O AFND rejeita uma entrada se nenhuma escolha levar até um estado final

Computação em AFD



Computação em AFND



## AFND $\times$ AFD

- Um AFND é mais poderoso que um AFD?
- Um AFND é equivalente a um AFD?

## AFND $\times$ AFD

- Um AFND é mais poderoso que um AFD?
- Um AFND é equivalente a um AFD?

**Teorema:** se uma linguagem  $L$  é aceita por um AFD, então  $L$  é aceita por um AFND

# AFND x AFD

- Um AFND é mais poderoso que um AFD?
- Um AFND é equivalente a um AFD?

**Teorema:** se uma linguagem  $L$  é aceita por um AFD, então  $L$  é aceita por um AFND

- **Prova:** note que todo AFD é um AFND que respeita as restrições impostas pelos AFDs. Assim, se  $L$  é aceita por um AFD, então  $L$  é também aceita por um AFND. Então, a classe de linguagens aceitas por um AFND inclui todas as linguagens aceitas por um AFD, portanto, todas as linguagens regulares

## AFND $\times$ AFD

**Teorema:** se uma linguagem  $L$  é aceita por um AFND, então  $L$  é aceita por um AFD



## AFND $\times$ AFD

**Teorema:** se uma linguagem  $L$  é aceita por um AFND, então  $L$  é aceita por um AFD

- **Prova:** vamos mostrar agora que se  $M$  é um AFND,  $L(M)$  é uma linguagem regular

## AFND x AFD

**Teorema:** se uma linguagem  $L$  é aceita por um AFND, então  $L$  é aceita por um AFD

- **Prova:** vamos mostrar agora que se  $M$  é um AFND,  $L(M)$  é uma linguagem regular
- Fazemos isso construindo um AFD  $M'$  a partir de um AFND  $M$ , assim mostramos também que AFND e AFD são equivalentes, ou seja,  $L(M) = L(M')$

# AFND x AFD

**Teorema:** se uma linguagem  $L$  é aceita por um AFND, então  $L$  é aceita por um AFD

- **Prova:** vamos mostrar agora que se  $M$  é um AFND,  $L(M)$  é uma linguagem regular
- Fazemos isso construindo um AFD  $M'$  a partir de um AFND  $M$ , assim mostramos também que AFND e AFD são equivalentes, ou seja,  $L(M) = L(M')$
- O AFD  $M'$  irá "lembrar" quais estados o AFND  $M$  poderia estar com a entrada  $w$  em determinado instante. Isso só funciona pois  $Q$  é finito.

# AFND $\times$ AFD

**Prova:** por construção

# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$

# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$

# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- Inicialmente vamos desconsiderar transições  $\varepsilon$

# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- Inicialmente vamos desconsiderar transições  $\varepsilon$
- $Q' = P(Q)$



# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- Inicialmente vamos desconsiderar transições  $\varepsilon$
- $Q' = P(Q)$
- $q_0' = \{q_0\}$

# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- Inicialmente vamos desconsiderar transições  $\varepsilon$
- $Q' = P(Q)$
- $q_0' = \{q_0\}$
- $\delta' =$  para cada  $R \in Q'$  e  $a \in \Sigma$ , temos que  $\delta'(R, a) = R'$ , onde  $R' = \bigcup_{r \in R} \delta(r, a)$ 
  - $R'$  conterá todos os estados alcançados a partir de cada estado  $r$  com cada símbolo  $a$  considerando o autômato  $M$

# AFND $\times$ AFD

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- Inicialmente vamos desconsiderar transições  $\varepsilon$
- $Q' = P(Q)$
- $q_0' = \{q_0\}$
- $\delta' =$  para cada  $R \in Q'$  e  $a \in \Sigma$ , temos que  $\delta'(R, a) = R'$ , onde  $R' = \bigcup_{r \in R} \delta(r, a)$ 
  - $R'$  conterá todos os estados alcançados a partir de cada estado  $r$  com cada símbolo  $a$  considerando o autômato  $M$
- $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$ , ou seja, um estado  $R$  é final se ele possui algum estado final de  $M$

# AFND $\times$ AFD

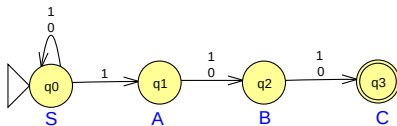
Provamos os seguintes teoremas:

- **Teorema:** se uma linguagem  $L$  é aceita por um AFND, então  $L$  é aceita por um AFD
- **Teorema:** se uma linguagem  $L$  é aceita por um AFD, então  $L$  é aceita por um AFND

Portanto, pode-se concluir também que os teoremas abaixo também estão provados

- **Teorema:** uma linguagem  $L$  é aceita por um AFND se e somente se  $L$  é aceita por um AFD
- **Teorema:** uma linguagem  $L$  é regular se e somente se existe um AFND que a reconhece

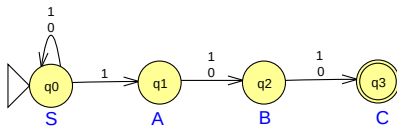
# AFND x AFD - Exemplo



	0	1
$\rightarrow S$	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{ }	{ }

	0	1
$\rightarrow \{S\}$		

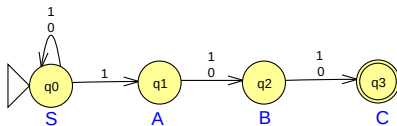
## AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}

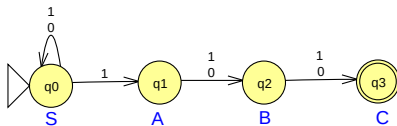
## AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}		

## AFND x AFD - Exemplo

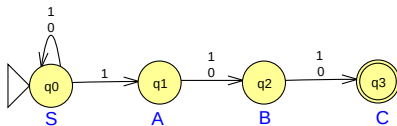


	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{ }	{ }

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}



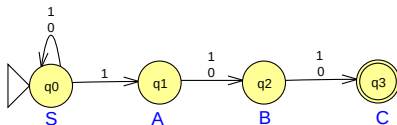
# AFND x AFD - Exemplo



	0	1
$\rightarrow S$	$\{S\}$	$\{S,A\}$
A	$\{B\}$	$\{B\}$
B	$\{C\}$	$\{C\}$
* C	$\{\}$	$\{\}$

	0	1
$\rightarrow \{S\}$	$\{S\}$	$\{S,A\}$
$\{S,A\}$	$\{S,B\}$	$\{S,A,B\}$
$\{S,B\}$		
$\{S,A,B\}$		

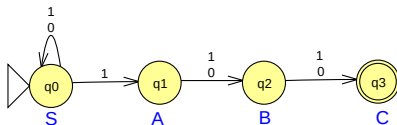
# AFND x AFD - Exemplo



	0	1
$\rightarrow S$	$\{S\}$	$\{S,A\}$
A	$\{B\}$	$\{B\}$
B	$\{C\}$	$\{C\}$
* C	$\{\}$	$\{\}$

	0	1
$\rightarrow \{S\}$	$\{S\}$	$\{S,A\}$
$\{S,A\}$	$\{S,B\}$	$\{S,A,B\}$
$\{S,B\}$	$\{S,C\}$	$\{S,A,C\}$
$\{S,A,B\}$		

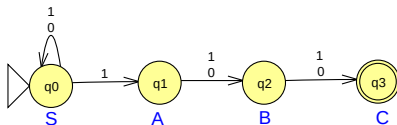
# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{ }	{ }

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}		
{S,C}		
{S,A,C}		

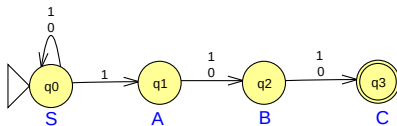
# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{ }	{ }

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
{S,C}		
{S,A,C}		

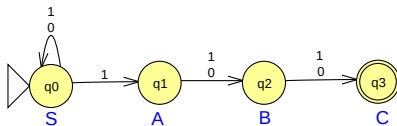
# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
{S,C}		
{S,A,C}		
{S,B,C}		
{S,A,B,C}		

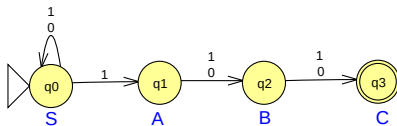
# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
{S,C}	{S}	{S,A}
{S,A,C}		
{S,B,C}		
{S,A,B,C}		

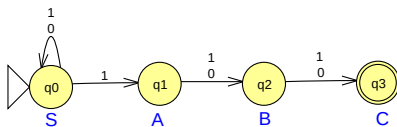
# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
{S,C}	{S}	{S,A}
{S,A,C}	{S,B}	{S,A,B}
{S,B,C}		
{S,A,B,C}		

# AFND x AFD - Exemplo

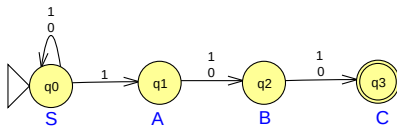


	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
{S,C}	{S}	{S,A}
{S,A,C}	{S,B}	{S,A,B}
{S,B,C}	{S,C}	{S,A,C}
{S,A,B,C}		



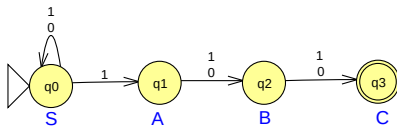
# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
{S,C}	{S}	{S,A}
{S,A,C}	{S,B}	{S,A,B}
{S,B,C}	{S,C}	{S,A,C}
{S,A,B,C}	{S,B,C}	{S,A,B,C}

# AFND x AFD - Exemplo



	0	1
→ S	{S}	{S,A}
A	{B}	{B}
B	{C}	{C}
* C	{}	{}

	0	1
→ {S}	{S}	{S,A}
{S,A}	{S,B}	{S,A,B}
{S,B}	{S,C}	{S,A,C}
{S,A,B}	{S,B,C}	{S,A,B,C}
* {S,C}	{S}	{S,A}
* {S,A,C}	{S,B}	{S,A,B}
* {S,B,C}	{S,C}	{S,A,C}
* {S,A,B,C}	{S,B,C}	{S,A,B,C}

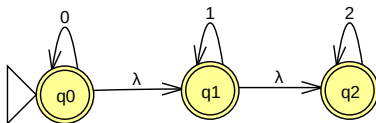
## AFND com transições $\varepsilon$ ( $\varepsilon$ -AFND)

Com transições  $\varepsilon$ , definimos o conceito de  $\varepsilon$ -Fecho( $q$ ) ou  $E(q)$ , sendo que este é o conjunto de todos os estados que podem ser alcançados a partir de um estado  $q$  com zero ou mais transições  $\varepsilon$ .

Da mesma forma,  $\varepsilon$ -Fecho( $R$ ) ou  $E(R)$  é o conjunto de todos os estados que podem ser alcançados a partir de qualquer estado de  $R$  com zero ou mais transições  $\varepsilon$ .

- **Teorema:** uma linguagem  $L$  é aceita por um  $\varepsilon$ -AFND se e somente se  $L$  é aceita por um AFD
- **Teorema:** uma linguagem  $L$  é regular se e somente se existe um  $\varepsilon$ -AFND que a reconhece

## AFND com transições $\varepsilon$ ( $\varepsilon$ -AFND)



$$E(q_0) = \{q_0, q_1, q_2\}$$

$$E(q_1) = \{q_1, q_2\}$$

$$E(q_2) = \{q_2\}$$

$E(\delta(\{q_0, q_1\}, 1)) = \{\mathbf{q_1}, q_2\}$  (alcança-se o estado  $q_1$  e então inclui-se todos os demais estados alcançados a partir de  $q_1$  com zero ou mais transições vazias)

$E(\delta(\{q_0, q_1, q_2\}, 0)) = \{\mathbf{q_0}, q_1, q_2\}$  (alcança-se o estado  $q_0$  e então inclui-se todos os demais estados alcançados a partir de  $q_0$  com zero ou mais transições vazias)

## $\varepsilon$ -AFND x AFD - Equivalência

**Teorema:** se uma linguagem  $L$  é aceita por um AFD, então  $L$  é aceita por um  $\varepsilon$ -AFND

## $\epsilon$ -AFND x AFD - Equivalência

**Teorema:** se uma linguagem  $L$  é aceita por um AFD, então  $L$  é aceita por um  $\epsilon$ -AFND

- **Prova:** note que todo AFD é um  $\epsilon$ -AFND que respeita as restrições impostas pelos AFDs. Assim, se  $L$  é aceita por um AFD, então  $L$  é também aceita por um  $\epsilon$ -AFND. Então, a classe de linguagens aceitas por um  $\epsilon$ -AFND inclui todas as linguagens aceitas por um AFD, portanto, todas as linguagens regulares

## $\epsilon$ -AFND $\times$ AFD - Equivalência

**Teorema:** se uma linguagem  $L$  é aceita por um  $\epsilon$ -AFND, então  $L$  é aceita por um AFD

## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Teorema:** se uma linguagem  $L$  é aceita por um  $\varepsilon$ -AFND, então  $L$  é aceita por um AFD

- **Prova:** vamos mostrar agora que se  $M$  é um  $\varepsilon$ -AFND,  $L(M)$  é uma linguagem regular



## $\varepsilon$ -AFND x AFD - Equivalência

**Teorema:** se uma linguagem  $L$  é aceita por um  $\varepsilon$ -AFND, então  $L$  é aceita por um AFD

- **Prova:** vamos mostrar agora que se  $M$  é um  $\varepsilon$ -AFND,  $L(M)$  é uma linguagem regular
- Fazemos isso construindo um AFD  $M'$  a partir de um  $\varepsilon$ -AFND  $M$ , assim mostramos também que  $\varepsilon$ -AFND e AFD são equivalentes, ou seja,  $L(M) = L(M')$

## $\varepsilon$ -AFND x AFD - Equivalência

**Teorema:** se uma linguagem  $L$  é aceita por um  $\varepsilon$ -AFND, então  $L$  é aceita por um AFD

- **Prova:** vamos mostrar agora que se  $M$  é um  $\varepsilon$ -AFND,  $L(M)$  é uma linguagem regular
- Fazemos isso construindo um AFD  $M'$  a partir de um  $\varepsilon$ -AFND  $M$ , assim mostramos também que  $\varepsilon$ -AFND e AFD são equivalentes, ou seja,  $L(M) = L(M')$
- O AFD  $M'$  irá "lembrar" quais estados o  $\varepsilon$ -AFND  $M$  poderia estar com a entrada  $w$  em determinado instante. Isso só funciona pois  $Q$  é finito.

## $\varepsilon$ -AFND x AFD - Equivalência

**Prova:** por construção

## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um  $\varepsilon$ -AFND que reconhece alguma linguagem  $L$

## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um  $\varepsilon$ -AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$

## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um  $\varepsilon$ -AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- $Q' = P(Q)$

## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um  $\varepsilon$ -AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- $Q' = P(Q)$
- $q_0' = \mathbf{E}(\{q_0\})$

## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Prova:** por construção

- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um  $\varepsilon$ -AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- $Q' = P(Q)$
- $q_0' = E(\{q_0\})$
- $\delta' =$  para cada  $R \in Q'$  e  $a \in \Sigma$ , temos que  $\delta'(R, a) = R'$ , onde  $R' = \bigcup_{r \in R} E(\delta(r, a))$ 
  - $R'$  conterá todos os estados alcançados a partir de cada estado  $r$  com cada símbolo  $a$  ou transições  $\varepsilon$  considerando o autômato  $M$

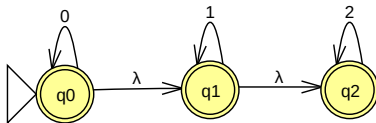


## $\varepsilon$ -AFND $\times$ AFD - Equivalência

**Prova:** por construção

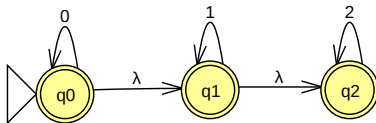
- Seja  $M = (Q, \Sigma, \delta, q_0, F)$  um  $\varepsilon$ -AFND que reconhece alguma linguagem  $L$
- Vamos construir um AFD  $M' = (Q', \Sigma, \delta', q_0', F')$  a partir de  $M$  e que reconheça  $L$
- $Q' = P(Q)$
- $q_0' = \mathbf{E}(\{q_0\})$
- $\delta' =$  para cada  $R \in Q'$  e  $a \in \Sigma$ , temos que  $\delta'(R, a) = R'$ , onde  $R' = \bigcup_{r \in R} \mathbf{E}(\delta(r, a))$ 
  - $R'$  conterá todos os estados alcançados a partir de cada estado  $r$  com cada símbolo  $a$  ou transições  $\varepsilon$  considerando o autômato  $M$
- $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$ , ou seja, um estado  $R$  é final se ele possui algum estado final de  $M$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



Para facilitar, primeiro computamos o  $\varepsilon$ -Fecho de todos os estados do  $\varepsilon$ -AFND

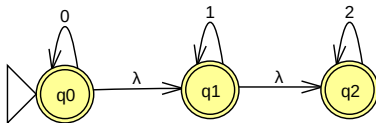
## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



Para facilitar, primeiro computamos o  $\varepsilon$ -Fecho de todos os estados do  $\varepsilon$ -AFND

$$E(q_0) = \{q_0, q_1, q_2\}, E(q_1) = \{q_1, q_2\}, E(q_2) = \{q_2\}$$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo

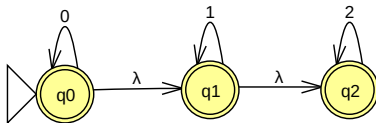


Para facilitar, primeiro computamos o  $\varepsilon$ -Fecho de todos os estados do  $\varepsilon$ -AFND

$$E(q_0) = \{q_0, q_1, q_2\}, E(q_1) = \{q_1, q_2\}, E(q_2) = \{q_2\}$$

Identificamos agora o estado inicial do AFD  $M'$ , que é dado por  $E(q_0)$ ,

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo

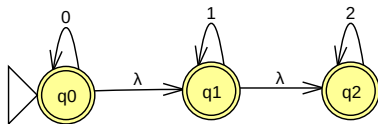


Para facilitar, primeiro computamos o  $\varepsilon$ -Fecho de todos os estados do  $\varepsilon$ -AFND

$$E(q_0) = \{q_0, q_1, q_2\}, E(q_1) = \{q_1, q_2\}, E(q_2) = \{q_2\}$$

Identificamos agora o estado inicial do AFD  $M'$ , que é dado por  $E(q_0)$ , assim,  $q_0' = E(q_0) = \{q_0, q_1, q_2\}$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q_0) = \{q_0, q_1, q_2\}$$

$$E(q_1) = \{q_1, q_2\}$$

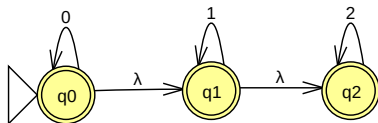
$$E(q_2) = \{q_2\}$$

$$q_0' = E(q_0) = \{q_0, q_1, q_2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q_0, q_1, q_2\}$			

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q_0) = \{q_0, q_1, q_2\}$$

$$E(q_1) = \{q_1, q_2\}$$

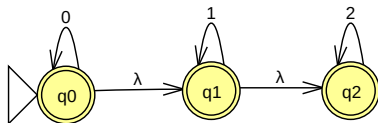
$$E(q_2) = \{q_2\}$$

$$q_0' = E(q_0) = \{q_0, q_1, q_2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	
$\rightarrow \{q_0, q_1, q_2\}$	$E(\delta(\{q_0, q_1, q_2\}, 0))$	$E(\delta(\{q_0, q_1, q_2\}, 1))$	$E(\delta(\{q_0, q_1, q_2\}, \lambda))$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q_0) = \{q_0, q_1, q_2\}$$

$$E(q_1) = \{q_1, q_2\}$$

$$E(q_2) = \{q_2\}$$

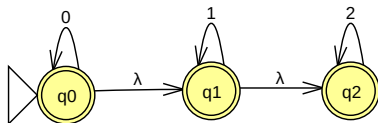
$$q_0' = E(q_0) = \{q_0, q_1, q_2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$E(\delta(\{q_0, q_1, q_2\}, 1))$	$E(\delta(\{q_0, q_1, q_2\}, 2))$



## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q_0) = \{q_0, q_1, q_2\}$$

$$E(q_1) = \{q_1, q_2\}$$

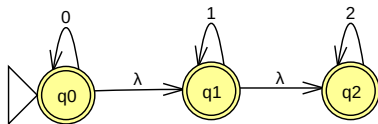
$$E(q_2) = \{q_2\}$$

$$q_0' = E(q_0) = \{q_0, q_1, q_2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$E(\delta(\{q_0, q_1, q_2\}, 2))$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q_0) = \{q_0, q_1, q_2\}$$

$$E(q_1) = \{q_1, q_2\}$$

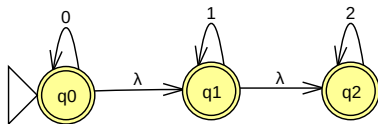
$$E(q_2) = \{q_2\}$$

$$q_0' = E(q_0) = \{q_0, q_1, q_2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q_0, q_1, q_2\}$	$\{\mathbf{q_0}, q_1, q_2\}$	$\{\mathbf{q_1}, q_2\}$	$\{\mathbf{q_2}\}$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

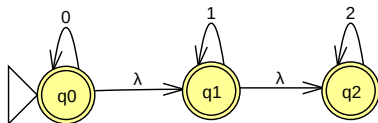
$$E(q2) = \{q2\}$$

$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q0, q1, q2\}$	$\{q0, q1, q2\}$	$\{q1, q2\}$	$\{q2\}$
$\{q1, q2\}$			
$\{q2\}$			

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

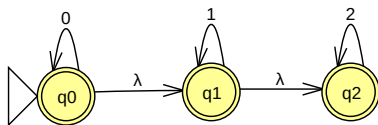
$$E(q2) = \{q2\}$$

$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q0, q1, q2\}$	$\{q0, q1, q2\}$	$\{q1, q2\}$	$\{q2\}$
$\{q1, q2\}$	$E(\delta(\{q1, q2\}, 0))$	$E(\delta(\{q1, q2\}, 1))$	$E(\delta(\{q1, q2\}, 2))$
$\{q2\}$	$E(\delta(\{q2\}, 0))$	$E(\delta(\{q2\}, 1))$	$E(\delta(\{q2\}, 2))$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

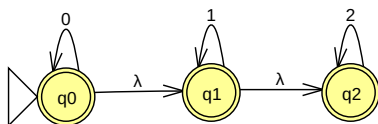
$$E(q2) = \{q2\}$$

$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q0, q1, q2\}$	$\{q0, q1, q2\}$	$\{q1, q2\}$	$\{q2\}$
$\{q1, q2\}$	$\{\}$	$\{q1, q2\}$	$\{q2\}$
$\{q2\}$	$E(\delta(\{q2\}, 0))$	$E(\delta(\{q2\}, 1))$	$E(\delta(\{q2\}, 2))$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

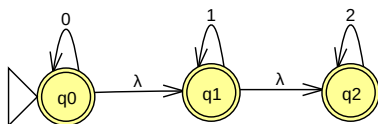
$$E(q2) = \{q2\}$$

$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q0, q1, q2\}$	$\{q0, q1, q2\}$	$\{q1, q2\}$	$\{q2\}$
$\{q1, q2\}$	$\{ \}$	$\{q1, q2\}$	$\{q2\}$
$\{q2\}$	$E(\delta(\{q2\}, 0))$	$E(\delta(\{q2\}, 1))$	$E(\delta(\{q2\}, 2))$
$\{ \}$	$E(\delta(\{ \}, 0))$	$E(\delta(\{ \}, 1))$	$E(\delta(\{ \}, 2))$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

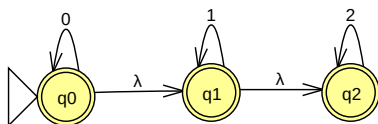
$$E(q2) = \{q2\}$$

$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q0, q1, q2\}$	<b><math>\{q0, q1, q2\}</math></b>	<b><math>\{q1, q2\}</math></b>	<b><math>\{q2\}</math></b>
$\{q1, q2\}$	$\{\}$	<b><math>\{q1, q2\}</math></b>	<b><math>\{q2\}</math></b>
$\{q2\}$	$\{\}$	$\{\}$	<b><math>\{q2\}</math></b>
$\{\}$	$E(\delta(\{\}, 0))$	$E(\delta(\{\}, 1))$	$E(\delta(\{\}, 2))$

## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

$$E(q2) = \{q2\}$$

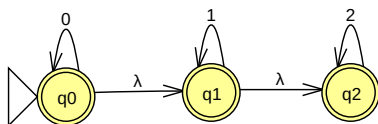
$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow \{q0, q1, q2\}$	<b><math>\{q0, q1, q2\}</math></b>	<b><math>\{q1, q2\}</math></b>	<b><math>\{q2\}</math></b>
$\{q1, q2\}$	$\{\}$	<b><math>\{q1, q2\}</math></b>	<b><math>\{q2\}</math></b>
$\{q2\}$	$\{\}$	$\{\}$	<b><math>\{q2\}</math></b>
$\{\}$	$\{\}$	$\{\}$	$\{\}$



## $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



$$E(q0) = \{q0, q1, q2\}$$

$$E(q1) = \{q1, q2\}$$

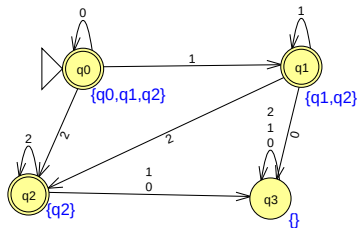
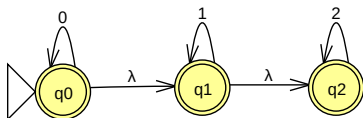
$$E(q2) = \{q2\}$$

$$q0' = E(q0) = \{q0, q1, q2\}$$

Agora podemos construir o AFD  $M'$  seguindo o procedimento apresentado anteriormente

	0	1	2
$\rightarrow *$ $\{q0, q1, q2\}$	<b><math>\{q0, q1, q2\}</math></b>	$\{q1, q2\}$	$\{q2\}$
$\quad *$ $\{q1, q2\}$	$\{\}$	$\{q1, q2\}$	$\{q2\}$
$\quad \quad *$ $\{q2\}$	$\{\}$	$\{\}$	$\{q2\}$
$\quad \quad \quad \{\}$	$\{\}$	$\{\}$	$\{\}$

# $\varepsilon$ -AFND x AFD - Equivalência - Exemplo



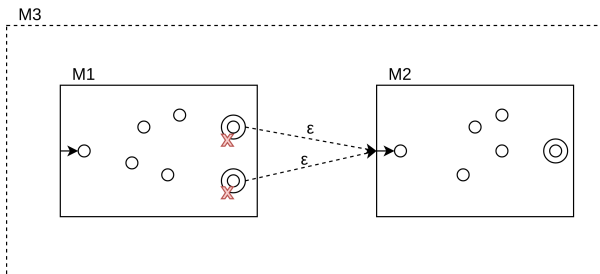
	0	1	2
$\rightarrow *$ {q0,q1,q2}	{ <b>q0</b> ,q1,q2}	{q1,q2}	{q2}
* {q1,q2}	{ }	{q1,q2}	{q2}
* {q2}	{ }	{ }	{q2}
{ }	{ }	{ }	{ }

# Linguagens Regulares - Operações - Concatenação

Seja  $L1, L2 \subseteq \Sigma^*$  e  $L1$  e  $L2$  sendo regulares, então  $L = L1 \circ L2$  também é uma linguagem regular. A classe das linguagens regulares é fechada na operação de concatenação.

**Prova:** por construção.

- Sejam os AFDs  $M1$  e  $M2$  que reconhecem  $L1$  e  $L2$ , respectivamente
- Vamos construir o autômato  $M3$  que reconhece a linguagem  $L = L1 \circ L2$  conforme o esquema abaixo

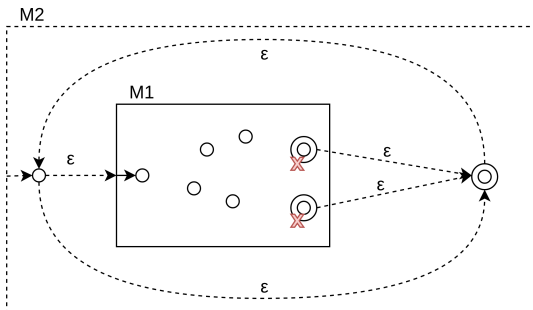


# Linguagens Regulares - Operações - Fecho de Kleene

Seja  $A \subseteq \Sigma^*$  e  $A$  sendo regular, então  $L = A^*$  também é uma linguagem regular. A classe das linguagens regulares é fechada na operação de fecho de Kleene.

**Prova:** por construção.

- Seja o AFD  $M1$  que reconhece  $A$
- Vamos construir o autômato  $M2$  que reconhece a linguagem  $L = A^*$  conforme o esquema abaixo



# Linguagens Regulares - Expressões Regulares

Expressão regular (ER) é outra forma de representar linguagens regulares

- Utiliza os símbolos do alfabeto e alguns caracteres especiais, por exemplo, para representar a quantidade de vezes que um terminal ou grupo de terminais se repetem dentro da formação de uma palavra
- Uma ER é expressa por meio de operações de concatenação, união e fecho de Kleene sobre linguagens

# Linguagens Regulares - Expressões Regulares

Uma expressão regular  $R$  sobre um alfabeto  $\Sigma$  é definida como:

- 1  $a$ , onde  $a \in \Sigma$ . Representa a linguagem  $\{a\} = L(a)$
- 2  $\varepsilon$  representa a linguagem  $\{\varepsilon\} = L(\varepsilon)$
- 3  $\emptyset$  representa a linguagem  $\{\} = L(\emptyset)$
- 4  $R_1 + R_2$ , onde  $R_1$  e  $R_2$  são ER e representa a linguagem resultante de  $L(R_1) \cup L(R_2)$
- 5  $R_1 \circ R_2$  ou  $R_1 R_2$ , onde  $R_1$  e  $R_2$  são ER e representa a linguagem resultante de  $L(R_1) \circ L(R_2)$
- 6  $R_1^*$ , onde  $R_1$  é uma ER e representa a linguagem resultante de  $L(R_1)^*$

# Linguagens Regulares - Expressões Regulares

Uma expressão regular  $R$  sobre um alfabeto  $\Sigma$  é definida como:

- 1  $a$ , onde  $a \in \Sigma$ . Representa a linguagem  $\{a\} = L(a)$
- 2  $\varepsilon$  representa a linguagem  $\{\varepsilon\} = L(\varepsilon)$
- 3  $\emptyset$  representa a linguagem  $\{\} = L(\emptyset)$
- 4  $R_1 + R_2$ , onde  $R_1$  e  $R_2$  são ER e representa a linguagem resultante de  $L(R_1) \cup L(R_2)$
- 5  $R_1 \circ R_2$  ou  $R_1 R_2$ , onde  $R_1$  e  $R_2$  são ER e representa a linguagem resultante de  $L(R_1) \circ L(R_2)$
- 6  $R_1^*$ , onde  $R_1$  é uma ER e representa a linguagem resultante de  $L(R_1)^*$

**Exemplo:** seja o alfabeto  $\Sigma = \{a, b\}$

$a$  e  $b$  são ER então pela regra 4  $a + b$  também é uma ER.

# Linguagens Regulares - Expressões Regulares - Exemplo

- $a = \{a\}$
- $b = \{b\}$
- $a + b = \{a\} \cup \{b\} = \{a, b\}$
- $(a + b)^* = \{a, b\}^* = \{\varepsilon, a, b, aa, ab, ba, bb, \dots\}$
- $(ab)^* = \{\varepsilon, ab, abab, \dots\}$

Precedência:  $*$   $>$   $\circ$   $>$   $+$



## Linguagens Regulares - Expressões Regulares - Exemplo

$$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$$

## Linguagens Regulares - Expressões Regulares - Exemplo

$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$

- $a|a(a+b)^*a = a + a(a+b)^*a$

## Linguagens Regulares - Expressões Regulares - Exemplo

$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$

- $a|a(a+b)^*a = a + a(a+b)^*a$

$$(a+b)^*bbb(a+b)^*$$

## Linguagens Regulares - Expressões Regulares - Exemplo

$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$

- $a|a(a+b)^*a = a + a(a+b)^*a$

$(a+b)^*bbb(a+b)^*$

- $L2 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ contém a subcadeia } bbb\}$

## Linguagens Regulares - Expressões Regulares - Exemplo

$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$

- $a|a(a+b)^*a = a + a(a+b)^*a$

$(a+b)^*bbb(a+b)^*$

- $L2 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ contém a subcadeia } bbb\}$

$((a+b)(a+b))^*$

# Linguagens Regulares - Expressões Regulares - Exemplo

$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$

- $a|a(a+b)^*a = a + a(a+b)^*a$

$(a+b)^*bbb(a+b)^*$

- $L2 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ contém a subcadeia } bbb\}$

$((a+b)(a+b))^*$

- $L3 = \{w \mid w \in \{a, b\}^* \text{ e } |w| \text{ é par } \}$

# Linguagens Regulares - Expressões Regulares - Exemplo

$L1 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com } a\}$

- $a|a(a+b)^*a = a + a(a+b)^*a$

$(a+b)^*bbb(a+b)^*$

- $L2 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ contém a subcadeia } bbb\}$

$((a+b)(a+b))^*$

- $L3 = \{w \mid w \in \{a, b\}^* \text{ e } |w| \text{ é par } \}$

Tente:

- $L4 = \{w \mid w \in \{a, b\}^* \text{ e o número de } a\text{'s é par } \}$
- $L5 = \{w \mid w \in \{a, b\}^* \text{ e } w \text{ começa e termina com a mesma letra } \}$

# Autômatos Finitos x Expressões Regulares

**Teorema:** uma linguagem é regular se e somente se alguma expressão regular a descreve

**Prova:** precisamos provar as duas direções

- Lemma 1: Se uma linguagem é descrita por uma expressão regular, então ela é regular
- Lemma 2: Se uma linguagem é regular, então ela é descrita por uma expressão regular



# Autômatos Finitos x Expressões Regulares

**Lemma 1:** Se uma linguagem é descrita por uma expressão regular, então ela é regular

**Prova:** por construção

- Já sabemos que uma linguagem é regular se e somente se algum AFD a reconhece. Sabemos também que AFND,  $\epsilon$ -AFND e AFD são equivalentes. Assim, podemos mostrar que ER e AFs são equivalentes para tanto provar o Lemma 1 como o Lemma 2.

# Autômatos Finitos x Expressões Regulares

**Lemma 1:** Se uma linguagem é descrita por uma expressão regular, então ela é regular

**Prova:** por construção

- Já sabemos que uma linguagem é regular se e somente se algum AFD a reconhece. Sabemos também que AFND,  $\varepsilon$ -AFND e AFD são equivalentes. Assim, podemos mostrar que ER e AFs são equivalentes para tanto provar o Lemma 1 como o Lemma 2.
- Mostraremos aqui como provar o Lemma 1, construindo um  $\varepsilon$ -AFND a partir de uma ER

# Autômatos Finitos x Expressões Regulares

**Lemma 1:** Se uma linguagem é descrita por uma expressão regular, então ela é regular

**Prova:** por construção

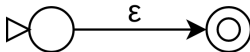
- Já sabemos que uma linguagem é regular se e somente se algum AFD a reconhece. Sabemos também que AFND,  $\varepsilon$ -AFND e AFD são equivalentes. Assim, podemos mostrar que ER e AFs são equivalentes para tanto provar o Lemma 1 como o Lemma 2.
- Mostraremos aqui como provar o Lemma 1, construindo um  $\varepsilon$ -AFND a partir de uma ER
- Para isso, basta criarmos para cada regra de construção de ER um  $\varepsilon$ -AFND equivalente

# Autômatos Finitos x Expressões Regulares

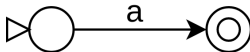
$\emptyset$



$\varepsilon$

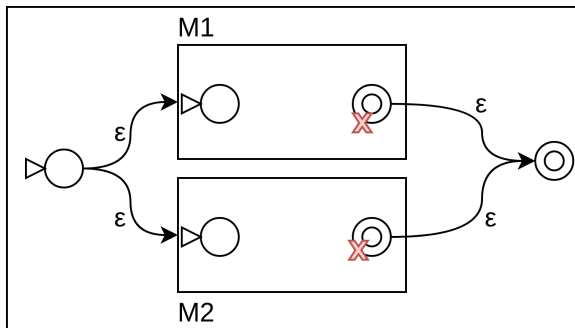


$a$



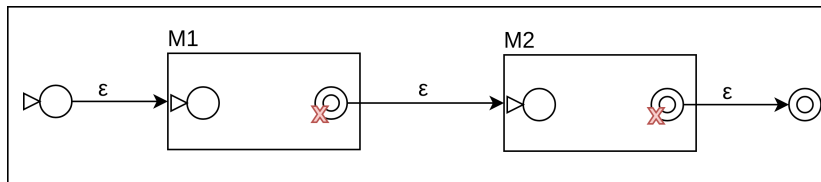
# Autômatos Finitos x Expressões Regulares

$$R_1 + R_2$$

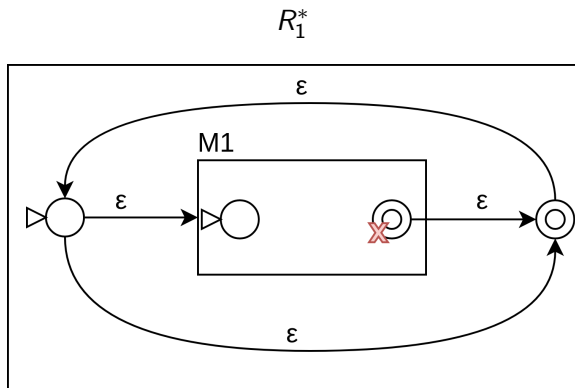


# Autômatos Finitos x Expressões Regulares

$$R_1 \circ R_2$$

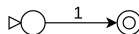
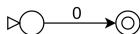
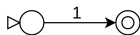


# Autômatos Finitos x Expressões Regulares



# Autômatos Finitos x Expressões Regulares - Exemplo

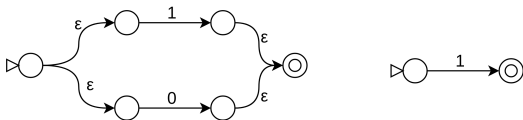
$(1 + 0)^*1$





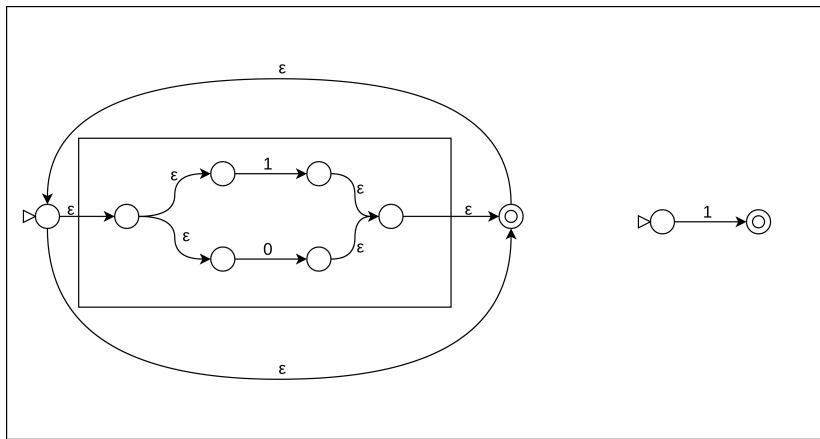
# Autômatos Finitos x Expressões Regulares - Exemplo

$(1 + 0)^*1$



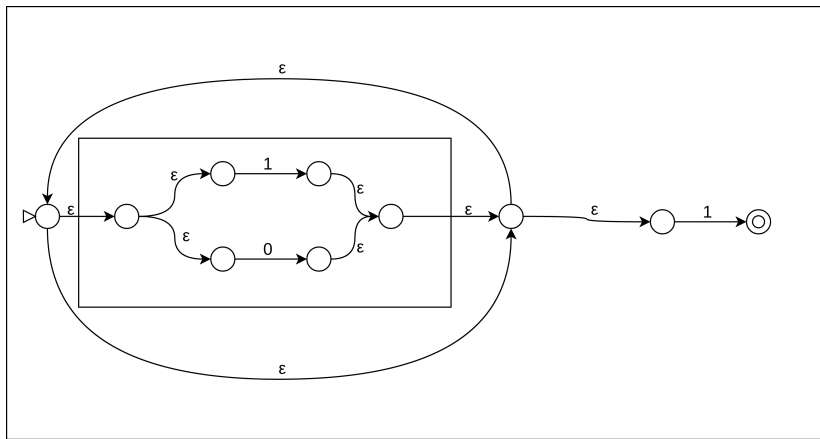
# Autômatos Finitos x Expressões Regulares - Exemplo

$(1 + 0)^*1$



# Autômatos Finitos x Expressões Regulares - Exemplo

$(1 + 0)^*1$



# Autômatos Finitos x Expressões Regulares

- Determine o autômato anterior
- Tente criar o autômato equivalente para a ER  $(10)^*(1 + 0)1$
- Para a prova do Lemma 2, veja o livro do Sipser, Cap 1 (Lemma 1.60)

## Linguagens Regulares - Propriedades de Fechamento

Podemos usar expressões regulares também para provar que as linguagens regulares são fechadas em certas operações, por exemplo, a operação de concatenação, onde se  $L_1$  e  $L_2$  são linguagens regulares, então  $L = L_1 \circ L_2$  também é regular.

**Prova:** se  $L_1$  e  $L_2$  são linguagens regulares então existem expressões regulares que descrevem as mesmas. Sejam elas,  $R_1$  e  $R_2$ , respectivamente, assim  $L(R_1) = L_1$  e  $L(R_2) = L_2$ .

## Linguagens Regulares - Propriedades de Fechamento

Podemos usar expressões regulares também para provar que as linguagens regulares são fechadas em certas operações, por exemplo, a operação de concatenação, onde se  $L_1$  e  $L_2$  são linguagens regulares, então  $L = L_1 \circ L_2$  também é regular.

**Prova:** se  $L_1$  e  $L_2$  são linguagens regulares então existem expressões regulares que descrevem as mesmas. Sejam elas,  $R_1$  e  $R_2$ , respectivamente, assim  $L(R_1) = L_1$  e  $L(R_2) = L_2$ .

- Note agora que  $L(R_1 \circ R_2) = L(R_1) \circ L(R_2)$ , ou seja, a linguagem resultante da concatenação das ERs  $R_1$  e  $R_2$  é exatamente a mesma linguagem resultante da concatenação das linguagens descritas por  $R_1$  e  $R_2$

# Linguagens Regulares - Propriedades de Fechamento

As propriedades de fechamento podem ser utilizadas, dentre outras coisas, para determinar que uma linguagem não pertence a certa classe de linguagens. Por exemplo, podemos facilmente mostrar que  $L$  não é uma linguagem regular simplesmente provando que  $\bar{L}$  não é uma linguagem regular.

- Note que as linguagens regulares são fechadas em complemento, então se  $L$  e  $\bar{L}$  devem ser regulares, assim, se  $\bar{L}$  não é regular, então  $L$  também não é regular.

# Linguagens Não Regulares e Lema do Bombeamento

Como mostrar que uma linguagem é regular?



# Linguagens Não Regulares e Lema do Bombeamento

Como mostrar que uma linguagem é regular?

- Construir um autômato finito que a reconheça ou uma expressão regular que a descreve

# Linguagens Não Regulares e Lema do Bombeamento

Como mostrar que uma linguagem é regular?

- Construir um autômato finito que a reconheça ou uma expressão regular que a descreve

Como mostrar que uma linguagem não é regular?

# Linguagens Não Regulares e Lema do Bombeamento

Como mostrar que uma linguagem é regular?

- Construir um autômato finito que a reconheça ou uma expressão regular que a descreve

Como mostrar que uma linguagem não é regular?

- Usar propriedades de fechamento
- Usar o lema do bombeamento (pumping lemma)

## Linguagens Não Regulares e Lema do Bombeamento

O lema do bombeamento baseia-se na ideia de que se uma linguagem  $L$  é regular, logo, existe um AFD com  $n$  estados que a reconhece, sendo  $n$  finito

# Linguagens Não Regulares e Lema do Bombeamento

O lema do bombeamento baseia-se na ideia de que se uma linguagem  $L$  é regular, logo, existe um AFD com  $n$  estados que a reconhece, sendo  $n$  finito

- Para toda palavra  $w \in L$  que tem comprimento maior ou igual a  $n$ , ou seja,  $|w| \geq n$ , então o AFD assume algum estado mais de uma vez durante o processo de reconhecimento de  $w$  e portanto há um ciclo no AFD

# Linguagens Não Regulares e Lema do Bombeamento

O lema do bombeamento baseia-se na ideia de que se uma linguagem  $L$  é regular, logo, existe um AFD com  $n$  estados que a reconhece, sendo  $n$  finito

- Para toda palavra  $w \in L$  que tem comprimento maior ou igual a  $n$ , ou seja,  $|w| \geq n$ , então o AFD assume algum estado mais de uma vez durante o processo de reconhecimento de  $w$  e portanto há um ciclo no AFD
- Então, existe uma forma de dividir  $w$  da seguinte forma  $w = uvz$ , tal que  $|uv| \leq n$  e  $|v| \geq 1$ , onde a subcadeia  $v$  é a parte de  $w$  que é reconhecida pelo ciclo

# Linguagens Não Regulares e Lema do Bombeamento

O lema do bombeamento baseia-se na ideia de que se uma linguagem  $L$  é regular, logo, existe um AFD com  $n$  estados que a reconhece, sendo  $n$  finito

- Para toda palavra  $w \in L$  que tem comprimento maior ou igual a  $n$ , ou seja,  $|w| \geq n$ , então o AFD assume algum estado mais de uma vez durante o processo de reconhecimento de  $w$  e portanto há um ciclo no AFD
- Então, existe uma forma de dividir  $w$  da seguinte forma  $w = uvz$ , tal que  $|uv| \leq n$  e  $|v| \geq 1$ , onde a subcadeia  $v$  é a parte de  $w$  que é reconhecida pelo ciclo
- Portanto, temos que  $uv^iz \in L$  para todo  $i \geq 0$

Se uma linguagem é regular, então ela certamente satisfaz o lema do bombeamento, porém nem toda linguagem que satisfaz o lema do bombeamento é regular

## Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular



# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = a^k b^k \in L$ . Note que  $|w| \geq k$ .

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = a^k b^k \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = a^k b^k \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^i z \in L$  para todo  $i \geq 0$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = a^k b^k \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^i z \in L$  para todo  $i \geq 0$
- Porém, note que se  $|uv| \leq k$  temos que a subcadeia  $uv$  conterá apenas símbolos  $a$ , inclusive  $v$ , visto que  $|v| \geq 1$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = a^k b^k \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^i z \in L$  para todo  $i \geq 0$
- Porém, note que se  $|uv| \leq k$  temos que a subcadeia  $uv$  conterá apenas símbolos  $a$ , inclusive  $v$ , visto que  $|v| \geq 1$
- Então, se fizermos  $i = 0$  temos que  $uv^0 z \notin L$ , pois certamente estaremos removendo ao menos um símbolo  $a$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{a, b\}^+ \text{ e } w = a^n b^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = a^k b^k \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^i z \in L$  para todo  $i \geq 0$
- Porém, note que se  $|uv| \leq k$  temos que a subcadeia  $uv$  conterá apenas símbolos  $a$ , inclusive  $v$ , visto que  $|v| \geq 1$
- Então, se fizermos  $i = 0$  temos que  $uv^0 z \notin L$ , pois certamente estaremos removendo ao menos um símbolo  $a$
- $L$  não pode ser regular!



# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = 1^{k^2} \in L$ . Note que  $|w| \geq k$ .

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = 1^{k^2} \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = 1^{k^2} \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^iz \in L$  para todo  $i \geq 0$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = 1^{k^2} \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^iz \in L$  para todo  $i \geq 0$
- Porém, note que se  $|w| = k^2$  e  $|uv| \leq k$  e fizermos  $i = 2$  temos que  $uv^2z \notin L$ , pois  $|uv^2z| = k^2 + k$  no máximo e  $|uv^2z| = k^2 + 1$  no mínimo

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = 1^{k^2} \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^i z \in L$  para todo  $i \geq 0$
- Porém, note que se  $|w| = k^2$  e  $|uv| \leq k$  e fizermos  $i = 2$  temos que  $uv^2 z \notin L$ , pois  $|uv^2 z| = k^2 + k$  no máximo e  $|uv^2 z| = k^2 + 1$  no mínimo
- Mas veja que o próximo quadrado perfeito após  $k^2$  seria  $(k+1)^2$ , portanto
$$k^2 < k^2 + 1 < k^2 + k < (k+1)^2 = k^2 + 2k + 1$$



# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{1\}^* \text{ e } w = 1^{n^2} \text{ e } n \geq 0\}$$

Vamos mostrar que  $L$  não é regular

**Prova:** por contradição

- Suponha que  $L$  é regular, então existe um AFD de tamanho  $k$  que a reconhece
- Seja agora  $w = 1^{k^2} \in L$ . Note que  $|w| \geq k$ .
- Então, podemos dividir  $w = uvz$ , tal que  $|uv| \leq k$  e  $|v| \geq 1$
- Deveríamos poder verificar que  $uv^i z \in L$  para todo  $i \geq 0$
- Porém, note que se  $|w| = k^2$  e  $|uv| \leq k$  e fizermos  $i = 2$  temos que  $uv^2 z \notin L$ , pois  $|uv^2 z| = k^2 + k$  no máximo e  $|uv^2 z| = k^2 + 1$  no mínimo
- Mas veja que o próximo quadrado perfeito após  $k^2$  seria  $(k+1)^2$ , portanto
$$k^2 < k^2 + 1 < k^2 + k < (k+1)^2 = k^2 + 2k + 1$$
- $L$  não pode ser regular!

# Linguagens Não Regulares e Lema do Bombeamento

Paradigma do adversário.

Vamos inicialmente reescrever o lema do bombeamento em expressões lógicas para o caso de  $L$  ser uma linguagem regular.

- 1  $\exists n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$
- 2  $\forall w \in L$ , onde  $|w| \geq n$
- 3  $\exists u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$
- 4  $\forall i \geq 0$ ,  $uv^iz \in L$

# Linguagens Não Regulares e Lema do Bombeamento

Paradigma do adversário.

Vamos agora reescrever o paradigma do adversário considerando  $L$  uma linguagem que não seja uma linguagem regular.

- ❶  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$
- ❷  $\exists w \in L$ , onde  $|w| \geq n$
- ❸  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$
- ❹  $\exists i \geq 0$ ,  $uv^iz \notin L$

# Linguagens Não Regulares e Lema do Bombeamento

Paradigma do adversário.

Vamos agora reescrever o paradigma do adversário considerando  $L$  uma linguagem que não seja uma linguagem regular.

- ❶  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$
- ❷  $\exists w \in L$ , onde  $|w| \geq n$
- ❸  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$
- ❹  $\exists i \geq 0$ ,  $uv^iz \notin L$

Durante o processo de prova utilizando o paradigma do adversário, faremos uma espécie de "jogo", onde o adversário escolhe sempre  $\forall$  e o provador (você) escolhe  $\exists$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- 1 Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- ① Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ② Proveedor:  $\exists w \in L$ , onde  $|w| \geq k$



# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^{\lceil k/2 \rceil} 10^{\lceil k/2 \rceil}$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^{\lceil k/2 \rceil} 10^{\lceil k/2 \rceil}$
- ❸ Adversário:  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^{\lceil k/2 \rceil} 10^{\lceil k/2 \rceil}$
- ❸ Adversário:  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$ 
  - $u = 0^{\lceil k/2 \rceil}$
  - $v = 1$
  - $z = 0^{\lceil k/2 \rceil}$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Proveedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^{\lceil k/2 \rceil} 10^{\lceil k/2 \rceil}$
- ❸ Adversário:  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$ 
  - $u = 0^{\lceil k/2 \rceil}$
  - $v = 1$
  - $z = 0^{\lceil k/2 \rceil}$
- ❹ Proveedor:  $\exists i \geq 0$ ,  $uv^iz \notin L$ 
  - Note que qualquer  $i$  que o proveedor tentar escolher, tem-se que  $v$  será repetido zero ou mais vezes, mas a palavra resultante sempre será palíndromo.

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^n \text{ e } n \geq 1\}$$

Vamos mostrar que  $L$  não é regular utilizando o paradigma do adversário

...

Por fim, o adversário venceu o jogo, o que significa que as escolhas feitas pelo provador foram ruins ou a linguagem é realmente uma linguagem regular

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- 1 Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$



# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^k 10^k$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^k 10^k$
- ❸ Adversário:  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Provedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^k 10^k$
- ❸ Adversário:  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$ 
  - Note que independentemente da escolha do adversário,  $uv$  sempre conterá apenas 0's e  $v$  conterá ao menos um 0

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w | w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

- ❶ Adversário:  $\forall n$ , onde  $n$  é o número de estados de um AFD que reconheça  $L$ 
  - $n = k$
- ❷ Proveedor:  $\exists w \in L$ , onde  $|w| \geq k$ 
  - $w = 0^k 10^k$
- ❸ Adversário:  $\forall u, v, z$ , tal que  $w = uvz$ ,  $|uv| \leq n$ ,  $|v| \geq 1$ 
  - Note que independentemente da escolha do adversário,  $uv$  sempre conterá apenas 0's e  $v$  conterá ao menos um 0
- ❹ Proveedor:  $\exists i \geq 0$ ,  $uv^i z \notin L$ 
  - $i = 0$

# Linguagens Não Regulares e Lema do Bombeamento

$$L = \{w \mid w \in \{0,1\}^* \text{ e } w = w^R \text{ e } n \geq 1\}$$

Vamos tentar novamente.

...

Por fim, o provador venceu o jogo, pois ao escolher  $i = 0$  e sabendo que  $v$  contém ao menos um 0 estaremos removendo um símbolo 0 da palavra  $w$ . Veja que ao fazer isso, o número de 0's antes do 1 é menor que o número de 0's depois do 1, ou seja, a palavra resultante não é palíndromo e a linguagem não é uma linguagem regular

# Conclusão

- Linguagens regulares
- Autômatos finitos determinísticos
- Autômatos finitos (determinísticos e não-determinísticos)
- Expressões regulares
- Linguagens não regulares e lema do bombeamento

## Material de apoio

- Livro Sipser, Capítulo 1
- Livro Hopcroft, Capítulo 2,3,4