



# Compiladores

## Autómatos finitos

Artur Pereira <artur@ua.pt>,  
Miguel Oliveira e Silva <mos@ua.pt>

DETI, Universidade de Aveiro

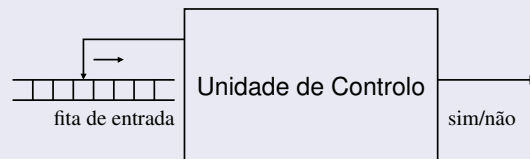
Ano letivo de 2022-2023

## Sumário

- ① Autómatos finitos deterministas (AFD)
- ② Redução de autómato finito determinista
- ③ Autómatos finitos não deterministas (AFND)
- ④ Equivalência entre AFD e AFND
- ⑤ Operações sobre autómatos finitos (AF)
- ⑥ Equivalência entre ER e AF
- ⑦ Equivalência entre GR e AF

# Autômato finito

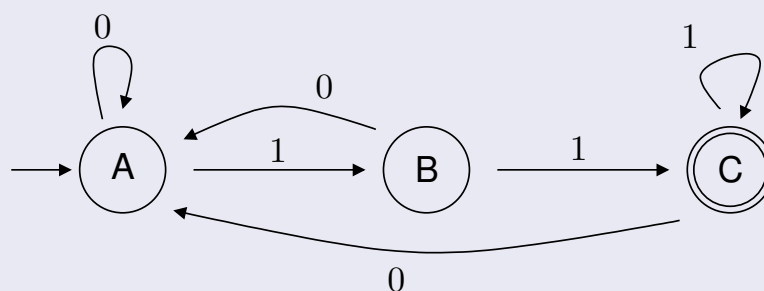
Um **autômato finito** é um mecanismo reconhecedor das palavras de uma linguagem regular



- A unidade de controlo é baseada nas noções de estado e de transição entre estados
  - número finito de estados
- A fita de entrada é só de leitura, com acesso sequencial
- A saída indica se a palavra é ou não aceite (reconhecida)
- Os autômatos finitos podem ser **deterministas**, **não deterministas** ou **generalizados**

# Autômato finito determinista

Um **autômato finito determinista** é um autômato finito

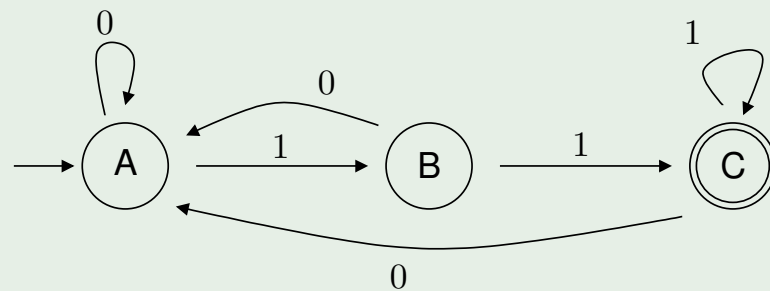


onde

- as transições estão associadas a símbolos individuais do alfabeto;
- de cada estado sai **uma e uma só** transição por cada símbolo do alfabeto;
- há um estado inicial;
- há 0 ou mais estados de aceitação, que determinam as palavras aceites;
- os caminhos que começam no estado inicial e terminam num estado de aceitação representam as palavras aceites (reconhecidas) pelo autômato.

## Autômato finito determinista: exemplo (1)

Q Que palavras binárias são reconhecidas pelo autômato seguinte?

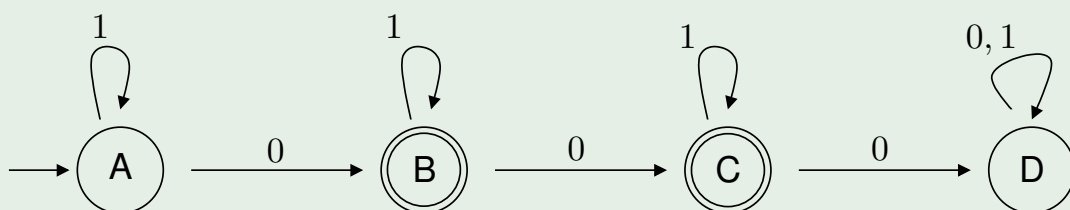


R Todas as palavras terminadas em 11.

E Obtenha uma expressão regular que represente a mesma linguagem.

## Autômato finito determinista: exemplo (2)

Q Que palavras binárias são reconhecidas pelo autômato seguinte?

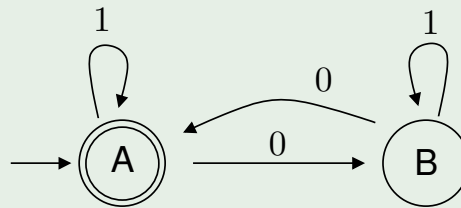


R Todas as palavras com apenas 1 ou 2 zeros.

E Obtenha uma expressão regular que represente a mesma linguagem.

## Autômato finito determinista: exemplo (3)

Q Que palavras binárias são reconhecidas pelo autômato seguinte?



R as sequências binárias com um número par de zeros.

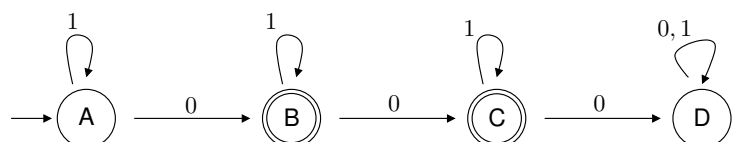
E Obtenha uma expressão regular que represente a mesma linguagem.

## Definição de autômato finito determinista

D Um **autômato finito determinista** (AFD) é um quintuplo  $M = (A, Q, q_0, \delta, F)$ , em que:

- $A$  é o alfabeto de entrada;
- $Q$  é um conjunto finito não vazio de estados;
- $q_0 \in Q$  é o estado inicial;
- $\delta : Q \times A \rightarrow Q$  é uma função que determina a transição entre estados; e
- $F \subseteq Q$  é o conjunto dos estados de aceitação.

- $A = \{0, 1\}$
- $Q = \{A, B, C, D\}$
- $q_0 = A$
- $F = \{B, C\}$
- Como representar  $\delta$ ?



## Definição de autômato finito determinista

**D** Um **autômato finito determinista** (AFD) é um quintuplo  $M = (A, Q, q_0, \delta, F)$ , em que:

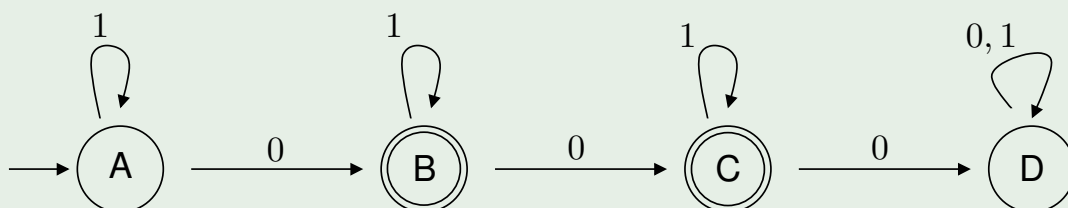
- $A$  é o alfabeto de entrada;
- $Q$  é um conjunto finito não vazio de estados;
- $q_0 \in Q$  é o estado inicial;
- $\delta : Q \times A \rightarrow Q$  é uma função que determina a transição entre estados; e
- $F \subseteq Q$  é o conjunto dos estados de aceitação.

**Q** Como representar a função  $\delta$  ?

- Matriz de  $|Q|$  linhas por  $|A|$  colunas. As células contêm elementos de  $Q$ .
- Conjunto de pares  $((q, a), q) \in (Q \times A) \times Q$ 
  - ou equivalentemente conjunto de triplos  $(q, a, q) \in Q \times A \times Q$

## Autômato finito determinista: exemplo (4)

**Q** Represente textualmente o AFD seguinte.



**R**

$M = (A, Q, q_0, \delta, F)$  com

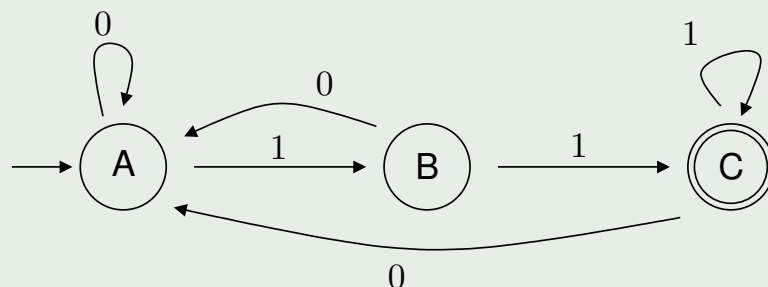
- $A = \{0, 1\}$
- $Q = \{A, B, C, D\}$
- $q_0 = A$
- $F = \{B, C\}$
- $\delta = \{$ 
  - $(A, 0, B), (A, 1, A),$
  - $(B, 0, C), (B, 1, B),$
  - $(C, 0, D), (C, 1, C),$
  - $(D, 0, D), (D, 1, D)\}$

•  $\delta =$

	0	1
A	B	A
B	C	B
C	D	C
D	D	D

## Autômato finito determinista: exemplo (5)

$\mathcal{Q}$  Represente textualmente o AFD seguinte.



$\mathcal{R}$

$M = (A, Q, q_0, \delta, F)$  com

- $A = \{0, 1\}$

- $Q = \{A, B, C\}$

- $q_0 = A$

- $F = \{C\}$

- $\delta = \{$

$(A, 0, A), (A, 1, B),$

$(B, 0, A), (B, 1, C),$

$(C, 0, A), (C, 1, C),$

- $\delta =$

	0	1
A	A	B
B	A	C
C	A	C

## Linguagem reconhecida por um AFD (1)

- Diz-se que um AFD  $M = (A, Q, q_0, \delta, F)$ , **aceita** uma palavra  $u \in A^*$  se  $u$  se puder escrever na forma  $u = u_1 u_2 \cdots u_n$  e existir uma sequência de estados  $s_0, s_1, \cdots, s_n$ , que satisfaça as seguintes condições:

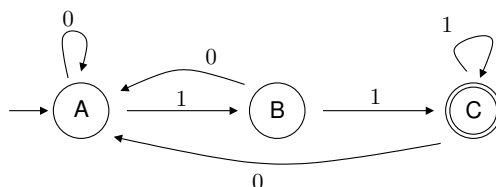
①  $s_0 = q_0$ ;

② qualquer que seja o  $i = 1, \cdots, n$ ,  $s_i = \delta(s_{i-1}, u_i)$ ;

③  $s_n \in F$ .

Caso contrário diz-se que  $M$  **rejeita** a sequência de entrada.

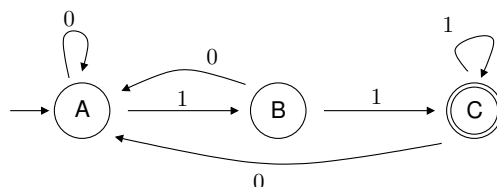
- A palavra  $\omega_1 = 0101$  faz o caminho  $A \xrightarrow{0} A \xrightarrow{1} B \xrightarrow{0} A \xrightarrow{1} B$ 
  - como  $B$  não é de aceitação,  $\omega_1$  não pertence à linguagem
- A palavra  $\omega_2 = 0011$  faz o caminho  $A \xrightarrow{0} A \xrightarrow{0} A \xrightarrow{1} B \xrightarrow{1} C$ 
  - como  $C$  é de aceitação,  $\omega_2$  pertence à linguagem



## Linguagem reconhecida por um AFD (2)

- Seja  $\delta^* : Q \times A^* \rightarrow Q$  a extensão de  $\delta$  definida indutivamente por
  - $\delta^*(q, \varepsilon) = q$
  - $\delta^*(q, av) = \delta^*(\delta(q, a), v)$ , com  $a \in A \wedge v \in A^*$
- $M$  aceita  $u$  se  $\delta^*(q_0, u) \in F$ .
- $L(M) = \{u \in A^* : M \text{ aceita } u\} = \{u \in A^* : \delta^*(q_0, u) \in F\}$

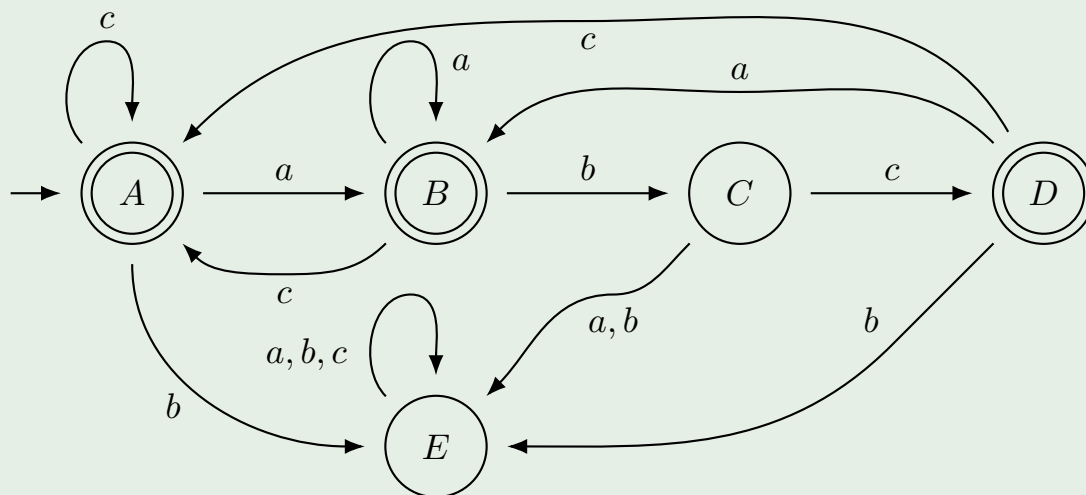
- $\delta^*(A, 0101) = \delta^*(\delta(A, 0), 101) = \delta^*(A, 101)$   
 $= \delta^*(\delta(A, 1), 01) = \delta^*(B, 01)$   
 $= \delta^*(\delta(B, 0), 1) = \delta^*(A, 1) = \delta^*(B, \varepsilon) = B$
- $\delta^*(A, 0011) = \delta^*(\delta(A, 0), 011) = \delta^*(A, 011)$   
 $= \delta^*(\delta(A, 0), 11) = \delta^*(A, 11)$   
 $= \delta^*(\delta(A, 1), 1) = \delta^*(B, 1) = \delta^*(C, \varepsilon) = C$



## Autómato finito determinista: exemplo (6)

- $\mathcal{Q}$  Sobre o alfabeto  $A = \{a, b, c\}$  considere a linguagem
- $$L = \{\omega \in A^* : (\omega_i = b) \Rightarrow ((\omega_{i-1} = a) \wedge (\omega_{i+1} = c))\}$$
- Projecte um autómato que reconheça  $L$ .

$\mathcal{R}$



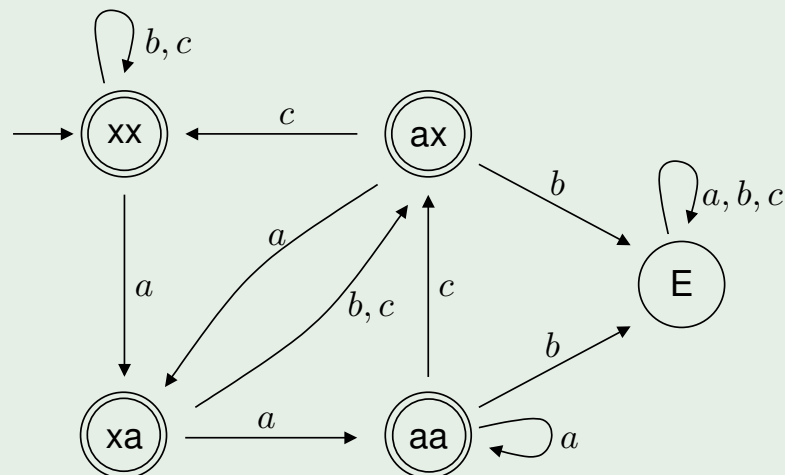
## Autómato finito determinista: exemplo (7)

Q Sobre o alfabeto  $A = \{a, b, c\}$  considere a linguagem

$$L = \{\omega \in A^* : (\omega_i = a) \Rightarrow (\omega_{i+2} \neq b)\}$$

Projecte um autómato que reconheça  $L$ .

R



## Autómato finito determinista: exemplo (8)

Q Sobre o alfabeto  $A = \{a, b, c\}$  considere a linguagem

$$L = \{\omega \in A^* : (\omega_i = a) \Rightarrow (\omega_{i+2} = b)\}$$

Projecte um autómato que reconheça  $L$ .

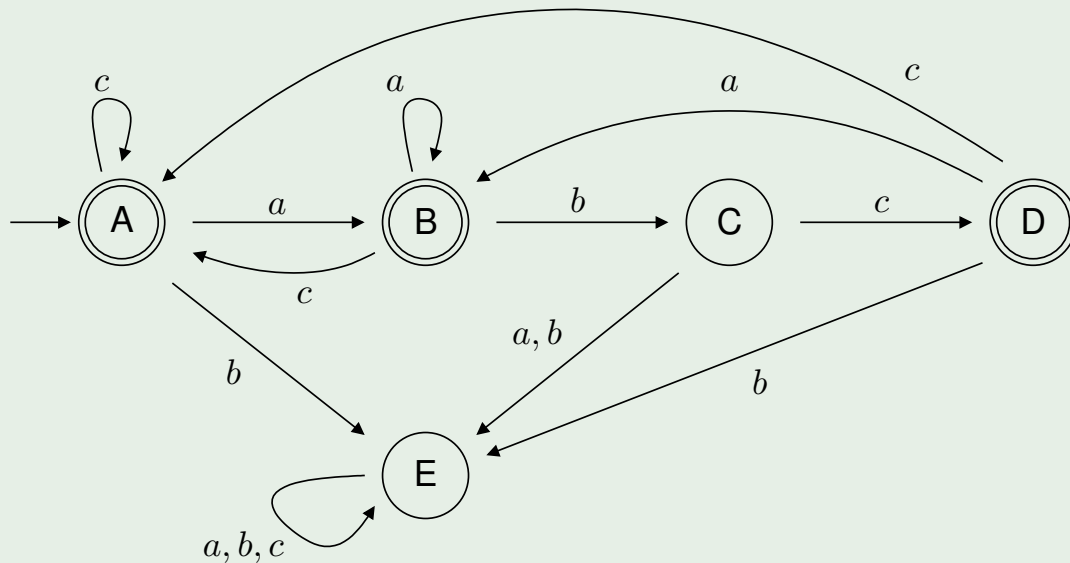
R

???



## Redução de autômato finito determinista (1)

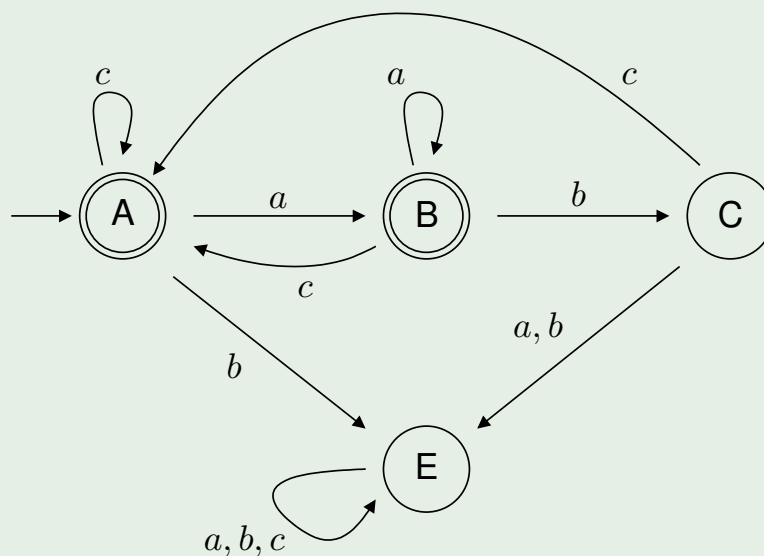
- Q Considere o autômato seguinte (o do exemplo 6) e compare os estados A e D. Que pode concluir ?



- São equivalentes. Por conseguinte, podem ser fundidos

## Redução de autômato finito determinista (2)

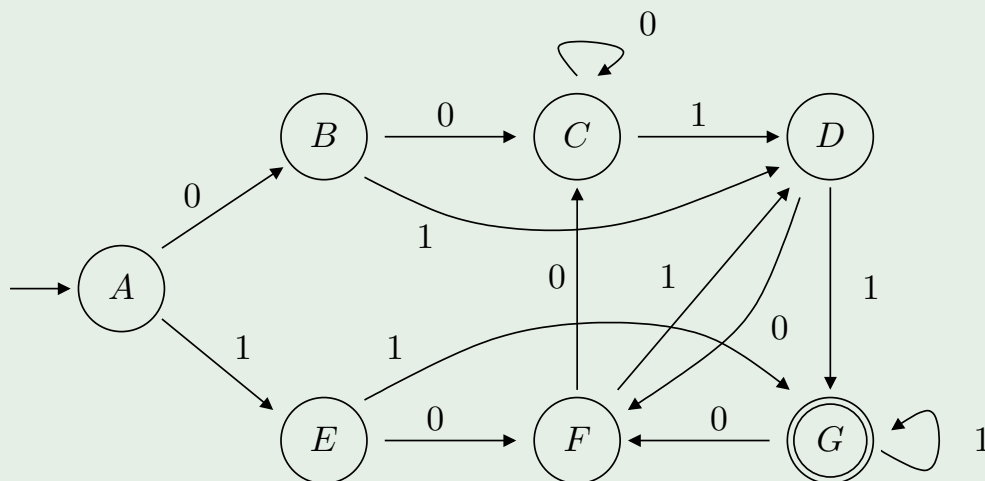
- O que resulta em



- Este, pode provar-se, não tem estados redundantes.
- Está no estado **reduzido**

## Algoritmo de Redução de AFD (1)

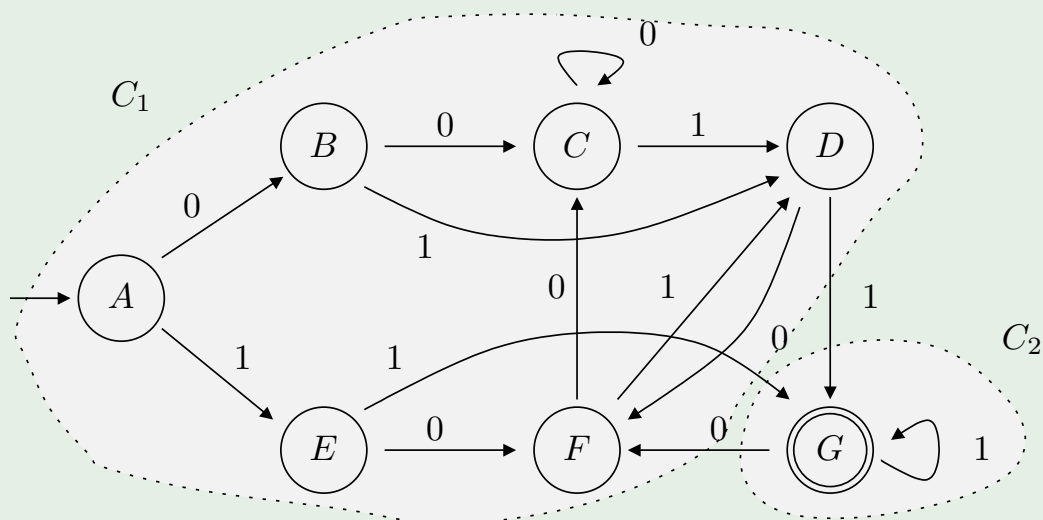
- Como proceder para reduzir um AFD?



- Primeiro, dividem-se os estados em dois conjuntos, um contendo os estados de aceitação e outro os de não-aceitação.

## Algoritmo de Redução de AFD (2)

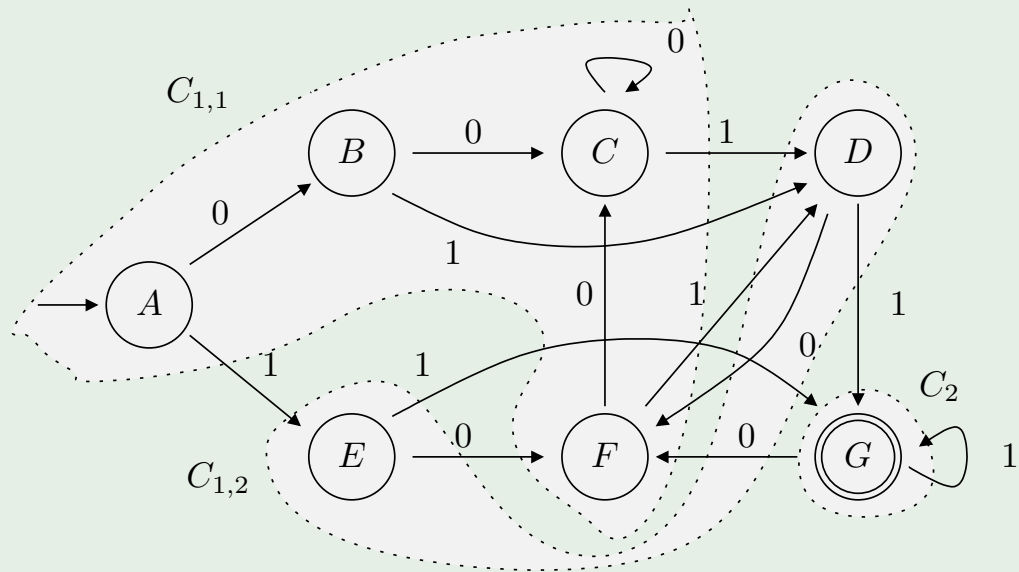
- Obtêm-se  $C_1 = \{A, B, C, D, E, F\}$  e  $C_2 = \{G\}$ .



- Em  $C_1$ , as transições em 0 são todas internas, mas as em 1 podem ser internas ou provocar uma ida para  $C_2$ . Logo, não representa uma classe de equivalência e tem de ser dividido.

## Algoritmo de Redução de AFD (3)

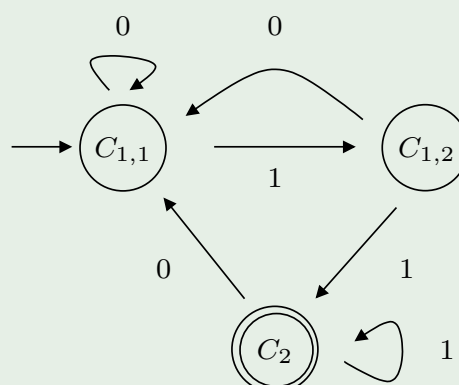
- Dividindo  $C_1$  em  $C_{1,1} = \{A, B, C, F\}$  e  $C_{1,2} = \{D, E\}$  obtém-se



- Pode verificar-se que  $C_{1,1}$ ,  $C_{1,2}$  e  $C_2$  são classes de equivalência, pelo que se chegou à versão reduzida do autômato.

## Algoritmo de Redução de AFD (4)

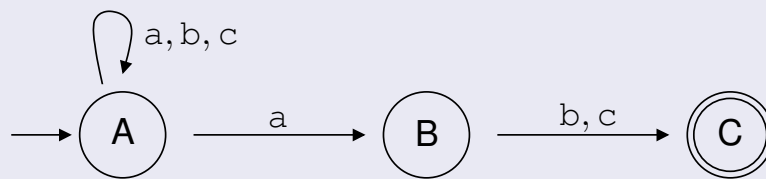
- Autômato reduzido



- Nos apontamentos encontra uma versão não gráfica do algoritmo.

# Autômato finito não determinista

Um **autômato finito não determinista** é um autômato finito

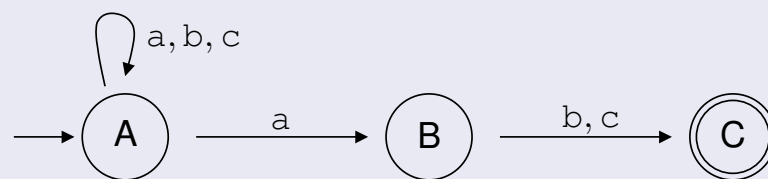


onde

- as transições estão associadas a símbolos individuais do alfabeto **ou à palavra vazia** ( $\epsilon$ );
  - de cada estado saem **zero ou mais** transições por cada símbolo do **alfabeto ou**  $\epsilon$ ;
  - há um estado inicial;
  - há 0 ou mais estados de aceitação, que determinam as palavras aceites;
  - os caminhos que começam no estado inicial e terminam num estado de aceitação representam as palavras aceites (reconhecidas) pelo autômato.
- 
- As transições múltiplas ou com  $\epsilon$  permitem alternativas de reconhecimento.
  - As transições ausentes representam quedas num estado de **morte** (estado não representado).

## AFND: caminhos alternativos

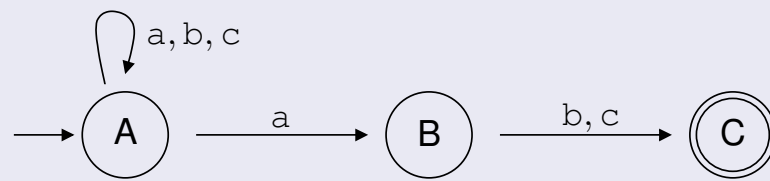
- Analise o processo de reconhecimento da palavra **abab** ?



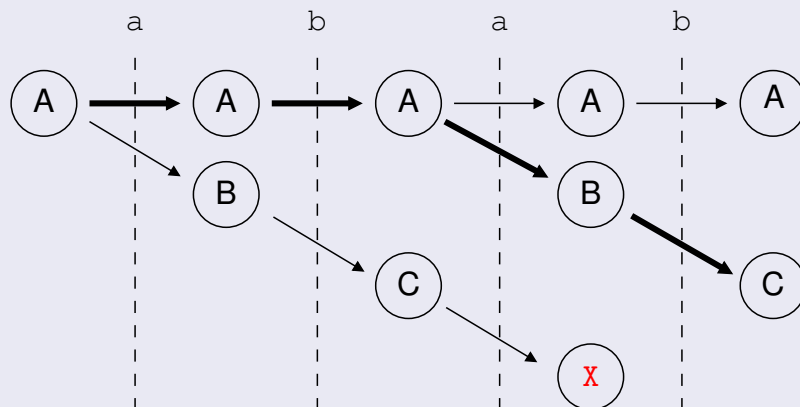
- Há 3 caminhos alternativos
  - 1  $A \xrightarrow{a} B \xrightarrow{b} C \xrightarrow{a} \text{X}$
  - 2  $A \xrightarrow{a} A \xrightarrow{b} A \xrightarrow{a} A \xrightarrow{b} A$
  - 3  $A \xrightarrow{a} A \xrightarrow{b} A \xrightarrow{a} B \xrightarrow{b} C$
- Como há um caminho que conduz a um estado de aceitação a palavra é reconhecida pelo autômato

## AFND: caminhos alternativos

- Analise o processo de reconhecimento da palavra *abab* ?

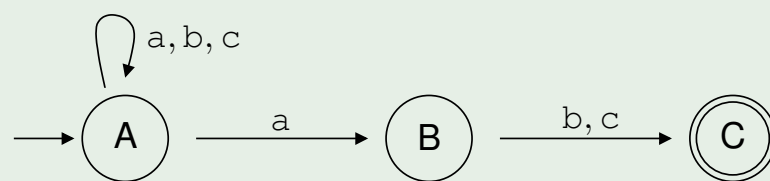


- Que se podem representar de forma arbórea



## AFND: exemplo

- Q** Que palavras são reconhecidas pelo autómato seguinte?



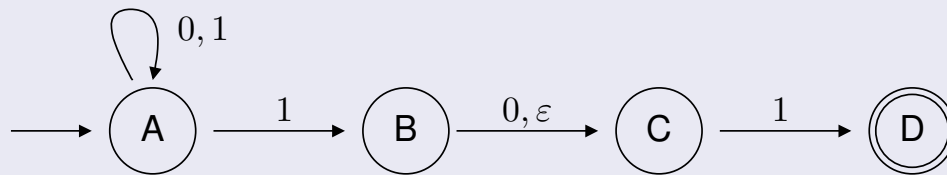
- R** Todas as palavras que terminarem em *ab* ou *ac*

$$L = \{\omega ax : \omega \in A^* \wedge x \in \{b, c\}\}.$$

- Percebe-se uma grande analogia entre este autómato e a expressão regular  $(a|b|c)^*a(b|c)$

## AFND com transições- $\epsilon$

- Considere o AFND seguinte que contém uma transição- $\epsilon$ .



- A palavra 101 é reconhecida pelo autômato através do caminho  
 $A \xrightarrow{1} B \xrightarrow{0} C \xrightarrow{1} D$
- A palavra 11 é reconhecida pelo autômato através do caminho  
 $A \xrightarrow{1} B \xrightarrow{\epsilon} C \xrightarrow{1} D$   
porque  $11 = 1\epsilon 1$
- Este autômato reconhece todas as palavras terminadas em 11 ou 101  
 $L = \{\omega_1\omega_2 : \omega_1 \in A^* \wedge \omega_2 \in \{11, 101\}\}.$

## AFND: definição

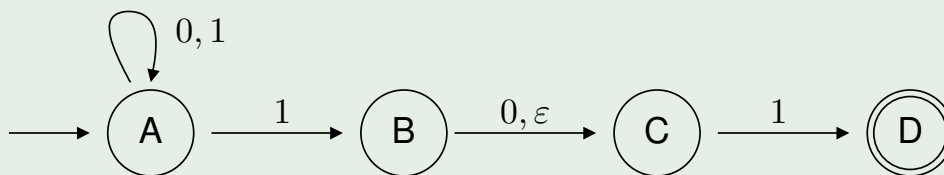
$\mathcal{D}$  Um **autômato finito não determinista** (AFND) é um quintuplo  $M = (A, Q, q_0, \delta, F)$ , em que:

- $A$  é o alfabeto de entrada;
- $Q$  é um conjunto finito não vazio de estados;
- $q_0 \in Q$  é o estado inicial;
- $\delta \subseteq (Q \times A_\epsilon \times Q)$  é a relação de transição entre estados, com  $A_\epsilon = A \cup \{\epsilon\}$ ;
- $F \subseteq Q$  é o conjunto dos estados de aceitação.

- 
- Apenas a definição de  $\delta$  difere em relação aos AFD.
  - Se se representar  $\delta$  na forma de uma tabela, as células são preenchidas com elementos de  $\wp(Q)$ , ou seja, sub-conjuntos de  $Q$ .

## AFND: Exemplo (2)

Q Represente textualmente o AFND



R  $M = (A, Q, q_0, \delta, F)$  com

- $A = \{0, 1\}$
- $Q = \{A, B, C, D\}$
- $q_0 = A$
- $F = \{D\}$
- $\delta = \{$
- $\delta =$

$(A, 0, A), (A, 1, A),$   
 $(A, 1, B), (B, 0, C),$   
 $(B, \varepsilon, C), (C, 1, D)$

	0	1	$\varepsilon$
A	{A}	{A, B}	{}
B	{C}	{}	{C}
C	{}	{D}	{}
D	{}	{}	{}

- O par  $(A, 1, A), (A, 1, B)$  faz com que  $\delta$  não seja uma função

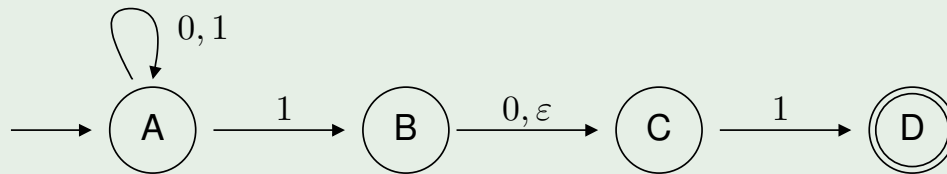
## AFND: linguagem reconhecida

- Diz-se que um AFND  $M = (A, Q, q_0, \delta, F)$ , **aceita** uma palavra  $u \in A^*$  se  $u$  se puder escrever na forma  $u = u_1 u_2 \cdots u_n$ , com  $u_i \in A_\varepsilon$ , e existir uma sequência de estados  $s_0, s_1, \dots, s_n$ , que satisfaça as seguintes condições:
  - 1  $s_0 = q_0$ ;
  - 2 qualquer que seja o  $i = 1, \dots, n$ ,  $(s_{i-1}, u_i, s_i) \in \delta$ ;
  - 3  $s_n \in F$ .
- Caso contrário diz-se que  $M$  **rejeita** a entrada.
- Note que  $n$  pode ser maior que  $|u|$ , porque alguns dos  $u_i$  podem ser  $\varepsilon$ .

- Usar-se-á a notação  $q_i \xrightarrow{u} q_j$  para indicar que a palavra  $u$  permite ir do estado  $q_i$  ao estado  $q_j$ .
- Usando esta notação tem-se  $L(M) = \{u : q_0 \xrightarrow{u} q_f \wedge q_f \in F\}$ .

## AFND: Exemplo de aplicação

Q Sobre o alfabeto  $A = \{0, 1\}$ , considere o AFND  $M$  seguinte



e a linguagem  $L = \{\omega \in A^* : \omega = (01)^n, n > 1\}$ . Mostre que  $L \subset L(M)$ .

R

## Equivalência entre AFD e AFND

- A classe das linguagens cobertas por um AFD é a mesma que a classe das linguagens cobertas por um AFND
- Isto significa que:
  - Se  $M$  é um AFD, então  $\exists_{M' \in \text{AFND}} : L(M') = L(M)$ .
  - Se  $M$  é um AFND, então  $\exists_{M' \in \text{AFD}} : L(M') = L(M)$ .

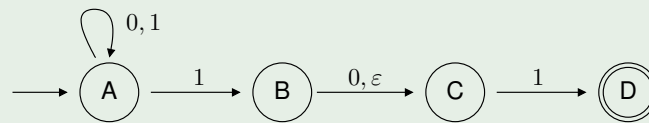
- Como determinar um AFND equivalente a um AFD dado ?
- Pelas definições de AFD e AFND, um AFD é um AFND. Porquê?
  - $Q$ ,  $q_0$  e  $F$  têm a mesma definição.
  - Nos AFD  $\delta : Q \times A \rightarrow Q$ .
  - Nos AFND  $\delta \subset Q \times A_\epsilon \times Q$
  - Mas, se  $\delta : Q \times A \rightarrow Q$  então  $\delta \subseteq Q \times A \times Q \subset Q \times A_\epsilon \times Q$
  - Logo, um AFD é um AFND



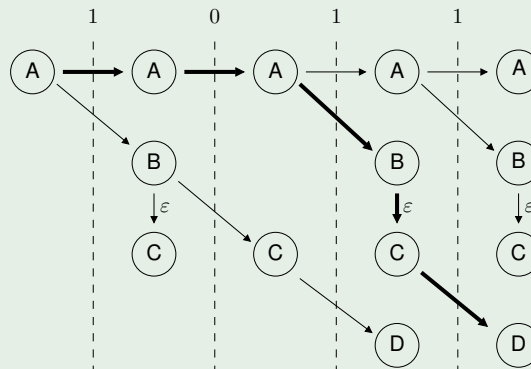
## Equivalente AFD de um AFND (1)

- Como determinar um AFD equivalente a um AFND dado ?

- No AFND



a árvore de reconhecimento da palavra 1011 sugere que a evolução se faz de sub-conjunto em sub-conjunto de estados



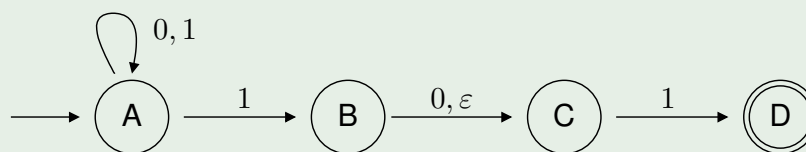
## Equivalente AFD de um AFND (2)

- Dado um AFND  $M = (A, Q, q_0, \delta, F)$ , considere o AFD  $M' = (A, Q', q'_0, \delta', F')$  onde:
  - $Q' = \wp(Q)$
  - $q'_0 = \varepsilon\text{-closure}(q_0)$
  - $F' = \{f' \in \wp(Q) : f' \cap F \neq \emptyset\}$
  - $\delta' = \wp(Q) \times A \rightarrow \wp(Q)$ ,  
 com  $\delta'(q', a) = \bigcup_{q \in q'} \{s : s \in \varepsilon\text{-closure}(s') \wedge (q, a, s') \in \delta\}$
- $M$  e  $M'$  reconhecem a mesma linguagem.

- $\varepsilon\text{-closure}(q)$  é o conjunto de estados constituído por  $q$  mais todos os direta ou indiretamente alcançáveis a partir de  $q$  apenas por transições- $\varepsilon$
- Note que:
  - O estado inicial ( $q'_0$ ) pode conter 1 ou mais elementos de  $Q$
  - Cada elemento do conjunto de chegada ( $f' \in F'$ ) por conter elementos de  $F$  e  $Q - F$

## Equivalente AFD de um AFND: exemplo

Q Determinar um AFD equivalente ao AFND seguinte ?



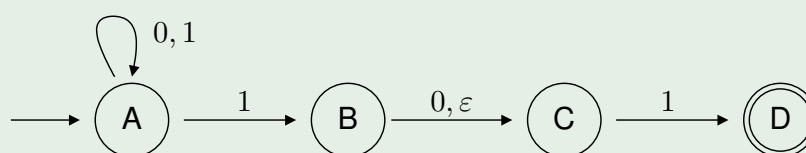
R

- $Q' = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{14}, X_{15}\}$ , com
 

$X_0 = \{\}$	$X_1 = \{A\}$	$X_2 = \{B\}$	$X_3 = \{A, B\}$
$X_4 = \{C\}$	$X_5 = \{A, C\}$	$X_6 = \{B, C\}$	$X_7 = \{A, B, C\}$
$X_8 = \{D\}$	$X_9 = \{A, D\}$	$X_{10} = \{B, D\}$	$X_{11} = \{A, B, D\}$
$X_{12} = \{C, D\}$	$X_{13} = \{A, C, D\}$	$X_{14} = \{B, C, D\}$	$X_{15} = \{A, B, C, D\}$
- $q'_0 = \varepsilon\text{-closure}(A) = \{A\} = X_1$
- $F' = \{X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{14}, X_{15}\}$

## Equivalente AFD de um AFND: exemplo

Q Determinar um AFD equivalente ao AFND seguinte ?



R

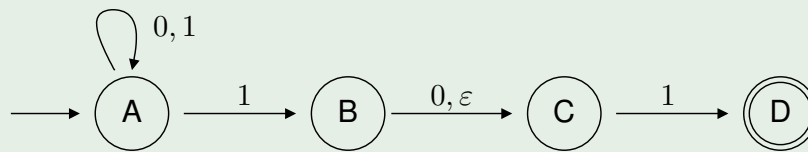
- $\delta' =$

estado	0	1		estado	0	1
$X_0 = \{\}$	$X_0$	$X_0$		$X_1 = \{A\}$	$X_1$	$X_7$
$X_2 = \{B\}$	$X_4$	$X_0$		$X_3 = \{A, B\}$	$X_5$	$X_7$
$X_4 = \{C\}$	$X_0$	$X_8$		$X_5 = \{A, C\}$	$X_1$	$X_{15}$
$X_6 = \{B, C\}$	$X_4$	$X_8$		$X_7 = \{A, B, C\}$	$X_5$	$X_{15}$
$X_8 = \{D\}$	$X_0$	$X_0$		$X_9 = \{A, D\}$	$X_1$	$X_7$
$X_{10} = \{B, D\}$	$X_4$	$X_0$		$X_{11} = \{A, B, D\}$	$X_5$	$X_7$
$X_{12} = \{C, D\}$	$X_0$	$X_8$		$X_{13} = \{A, C, D\}$	$X_1$	$X_{15}$
$X_{14} = \{B, C, D\}$	$X_4$	$X_8$		$X_{15} = \{A, B, C, D\}$	$X_5$	$X_{15}$

- Serão todos estes estados necessários?

## Equivalente AFD de um AFND: exemplo (2)

Q Determinar um AFD equivalente ao AFND seguinte ?

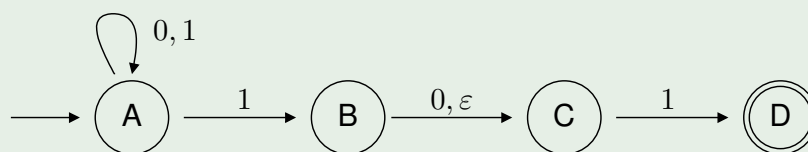


$\mathcal{R}$

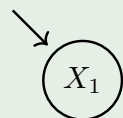
- Conseguir-se o mesmo resultado através de um processo construtivo.

## Equivalente AFD de um AFND: exemplo (2)

Q Determinar um AFD equivalente ao AFND seguinte ?



$\mathcal{R}$

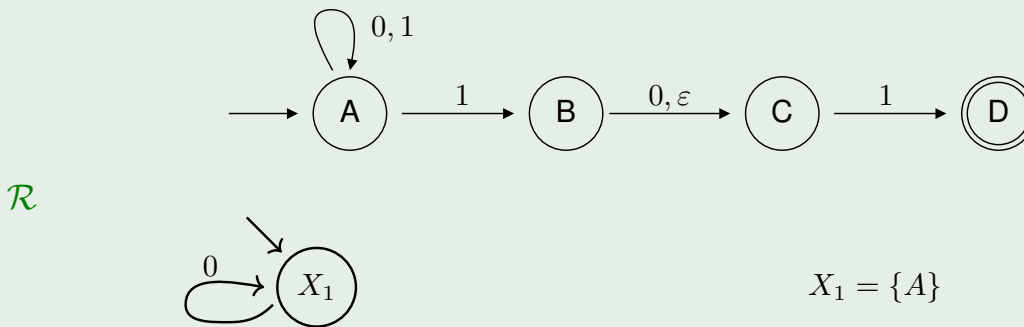


$$X_1 = \{A\}$$

- Comece-se com o estado inicial ( $X_1 = \{A\}$ )

## Equivalente AFD de um AFND: exemplo (2)

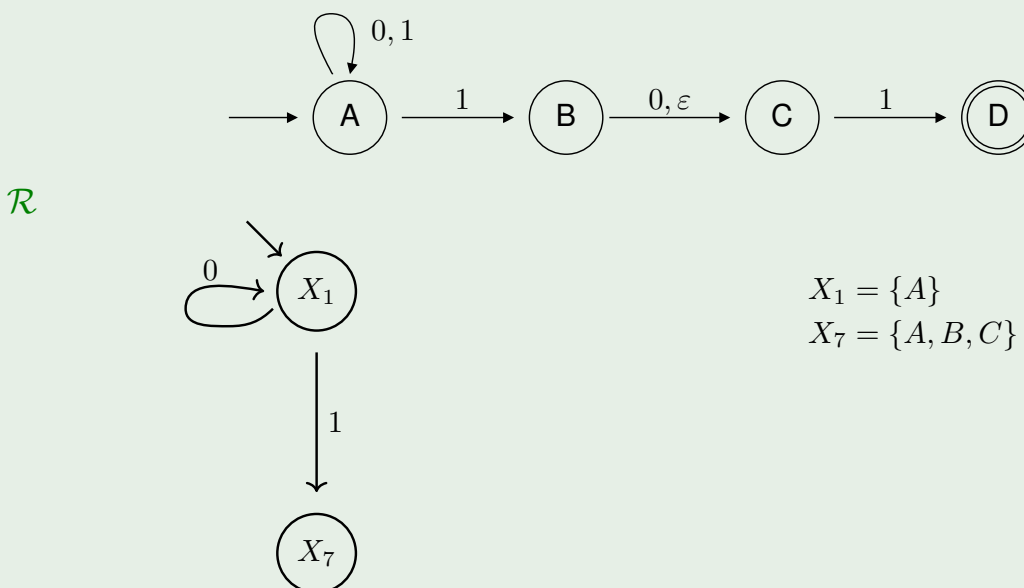
Q Determinar um AFD equivalente ao AFND seguinte ?



- $\delta'(X_1, 0) = \varepsilon\text{-closure}(A) = \{A\}$

## Equivalente AFD de um AFND: exemplo (2)

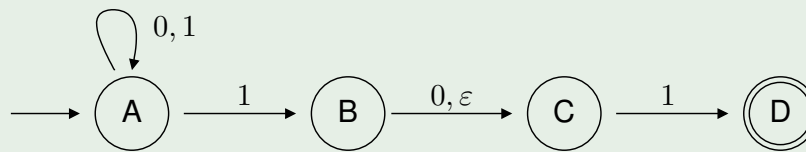
Q Determinar um AFD equivalente ao AFND seguinte ?



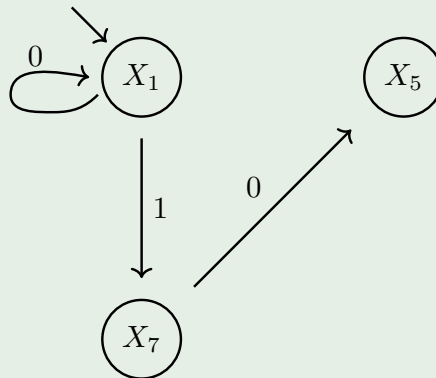
- $\delta'(X_1, 1) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(B) = \{A\} \cup \{B, C\} = \{A, B, C\}$

## Equivalente AFD de um AFND: exemplo (2)

Q Determinar um AFD equivalente ao AFND seguinte ?



$\mathcal{R}$

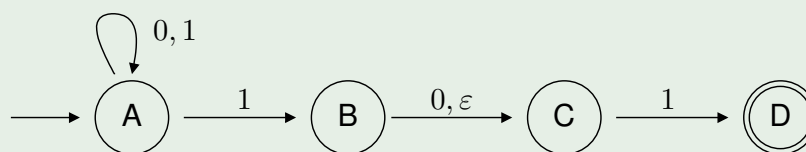


$X_1 = \{A\}$   
 $X_7 = \{A, B, C\}$   
 $X_5 = \{A, C\}$

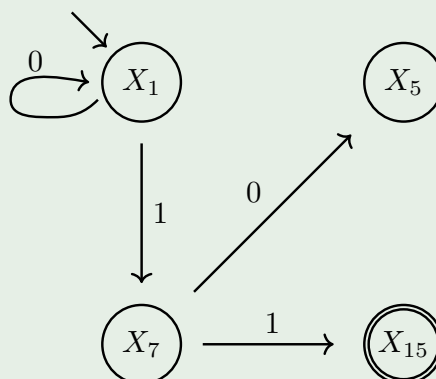
- $\delta'(X_7, 0) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(C) = \{A\} \cup \{C\} = \{A, C\}$

## Equivalente AFD de um AFND: exemplo (2)

Q Determinar um AFD equivalente ao AFND seguinte ?



$\mathcal{R}$

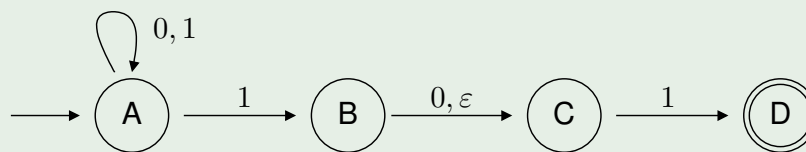


$X_1 = \{A\}$   
 $X_7 = \{A, B, C\}$   
 $X_5 = \{A, C\}$   
 $X_{15} = \{A, B, C, D\}$

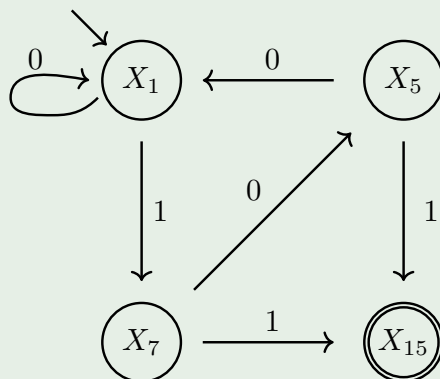
- $\delta'(X_7, 1) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(B) \cup \varepsilon\text{-closure}(D) = \{A\} \cup \{B, C\} \cup \{D\} = \{A, B, C, D\}$
- É de aceitação porque  $\{A, B, C, D\} \cap \{D\} \neq \emptyset$

## Equivalente AFD de um AFND: exemplo (2)

Q Determinar um AFD equivalente ao AFND seguinte ?



$\mathcal{R}$

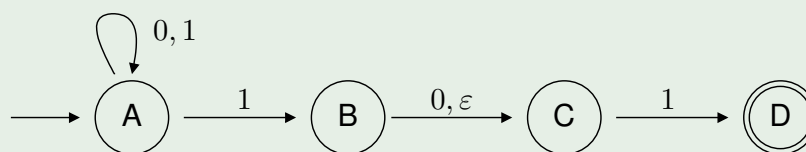


$X_1 = \{A\}$   
 $X_7 = \{A, B, C\}$   
 $X_5 = \{A, C\}$   
 $X_{15} = \{A, B, C, D\}$

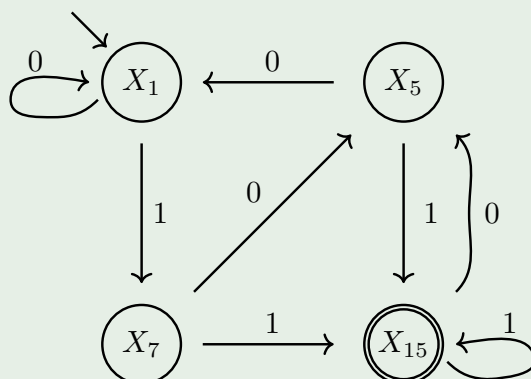
- $\delta'(X_5, 0) = \varepsilon\text{-closure}(A) = \{A\}$
- $\delta'(X_5, 1) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(B) \cup \varepsilon\text{-closure}(D) = \{A\} \cup \{B, C\} \cup \{D\} = \{A, B, C, D\}$

## Equivalente AFD de um AFND: exemplo (2)

Q Determinar um AFD equivalente ao AFND seguinte ?



$\mathcal{R}$



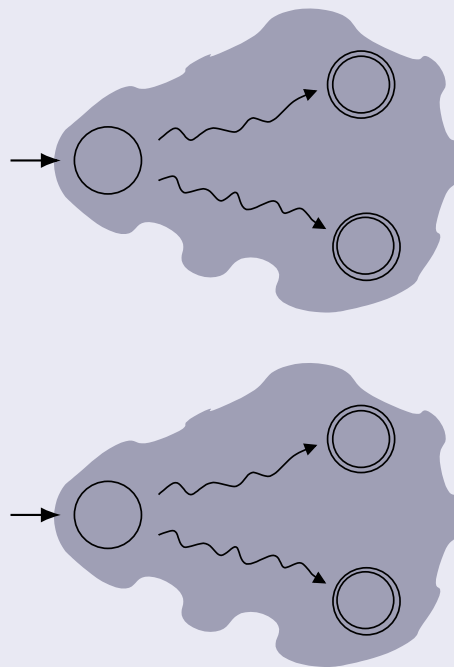
$X_1 = \{A\}$   
 $X_7 = \{A, B, C\}$   
 $X_5 = \{A, C\}$   
 $X_{15} = \{A, B, C, D\}$

- $\delta'(X_{15}, 0) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(C) = \{A\} \cup \{C\} = \{A, C\}$
- $\delta'(X_{15}, 1) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(B) \cup \varepsilon\text{-closure}(D) = \{A\} \cup \{B, C\} \cup \{D\} = \{A, B, C, D\}$

# Operações sobre AFD e AFND

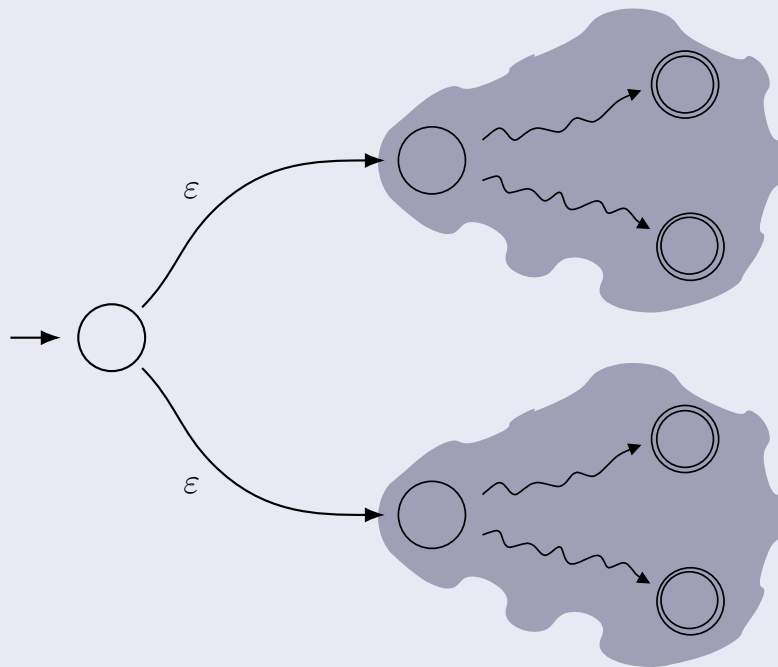
- Os automáto finitos (AF) são fechados sobre as operações de:
  - Reunião
  - Concatenação
  - Fecho
  - Intersecção
  - Complementação

## Reunião de AF



- Como criar um AF que represente a reunião destes dois AF?

## Reunião de AF



- acrescenta-se um novo estado que passa a ser o inicial
- e acrescentam-se transições- $\varepsilon$  deste novo estado para os estados iniciais originais

## Reunião de AF: definição

**D** Seja  $M_1 = (A, Q_1, q_1, \delta_1, F_1)$  e  $M_2 = (A, Q_2, q_2, \delta_2, F_2)$  dois autómatos (AFD ou AFND) quaisquer.

O AFND  $M = (A, Q, q_0, \delta, F)$ , onde

$$Q = Q_1 \cup Q_2 \cup \{q_0\}, \quad \text{com } q_0 \notin Q_1 \wedge q_0 \notin Q_2$$

$$F = F_1 \cup F_2$$

$$\delta = \delta_1 \cup \delta_2 \cup \{(q_0, \varepsilon, q_1), (q_0, \varepsilon, q_2)\}$$

implementa a reunião de  $M_1$  e  $M_2$ , ou seja,  $L(M) = L(M_1) \cup L(M_2)$ .



## Reunião de AF: exemplo (1)

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça  $L = L_1 \cup L_2$ .

R

- Como criar um AF que represente a reunião de  $L_1$  e  $L_2$ ?

## Reunião de AF: exemplo (1)

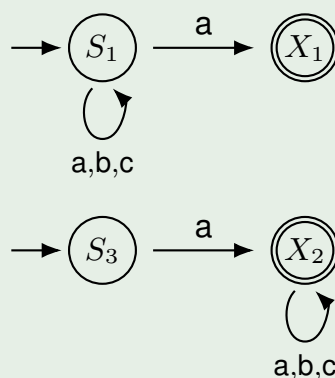
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça  $L = L_1 \cup L_2$ .

R



- Constroi-se um AF para a linguagem  $L_1$
- Constroi-se um AF para a linguagem  $L_2$

## Reunião de AF: exemplo (1)

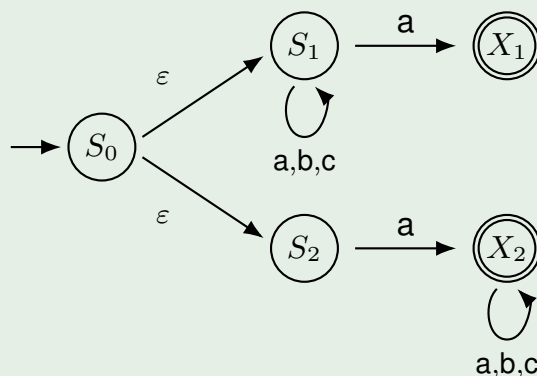
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça  $L = L_1 \cup L_2$ .

R



- Acrescenta-se um novo estado ( $S_0$ ), que passa a ser o inicial
- E acrescentam-se transições- $\epsilon$  de  $S_0$  (novo estado inicial) para  $S_1$  e  $S_2$  (os estados iniciais originais)

## Reunião de AF: exemplo (1)

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça  $L = L_1 \cup L_2$ .

R

$M_1 = (A, Q_1, q_1, \delta_1, F_1)$  com

$$Q_1 = \{S_1, X_1\}, \quad q_1 = S_1, \quad F_1 = \{X_1\}$$

$$\delta_1 = \{(S_1, a, S_1), (S_1, b, S_1), (S_1, c, S_1), (S_1, a, X_1)\}$$

$M_2 = (A, Q_2, q_2, \delta_2, F_2)$  com

$$Q_2 = \{S_2, X_2\}, \quad q_2 = S_2, \quad F_2 = \{X_2\}$$

$$\delta_2 = \{(S_2, a, X_2), (X_2, a, X_2), (X_2, b, X_2), (X_2, c, X_2)\}$$

$M = M_1 \cup M_2 = (A, Q, q_0, \delta, F)$  com

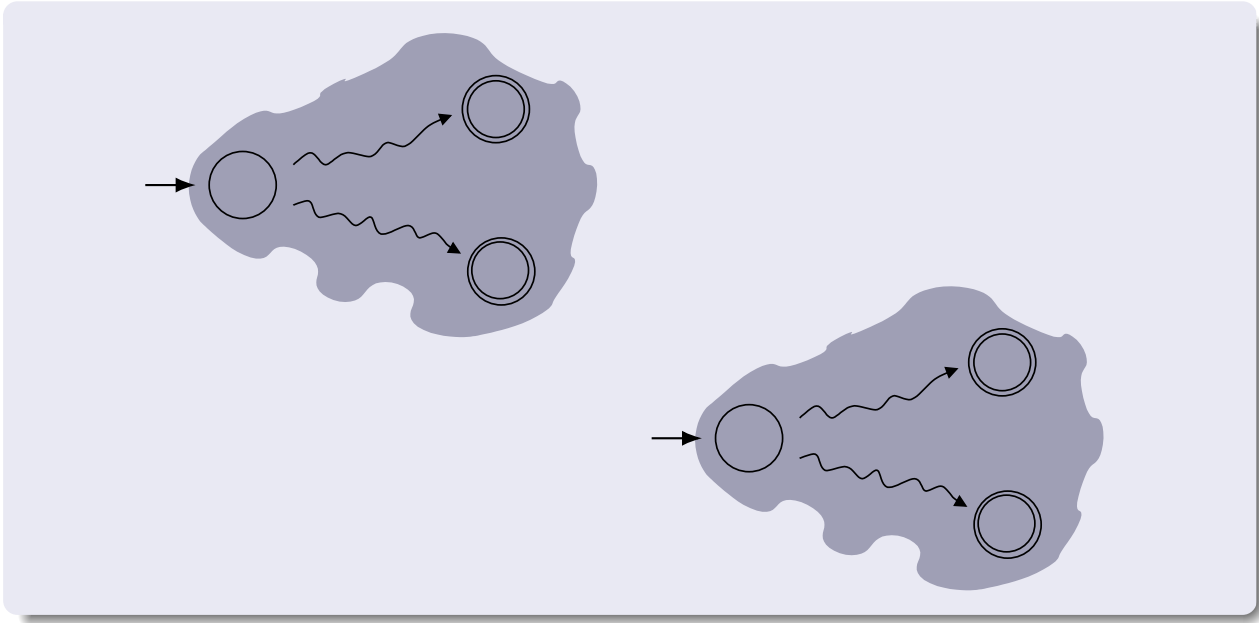
$$Q = \{S_0, S_1, X_1, S_2, X_2\}, \quad q_0 = S_0, \quad F = \{X_1, X_2\},$$

$$\delta = \{(S_0, \epsilon, S_1), (S_0, \epsilon, S_2), (S_1, a, S_1), (S_1, b, S_1), (S_1, c, S_1),$$

$$(S_1, a, X_1), (S_2, a, X_2), (X_2, a, X_2), (X_2, b, X_2), (X_2, c, X_2)\}$$

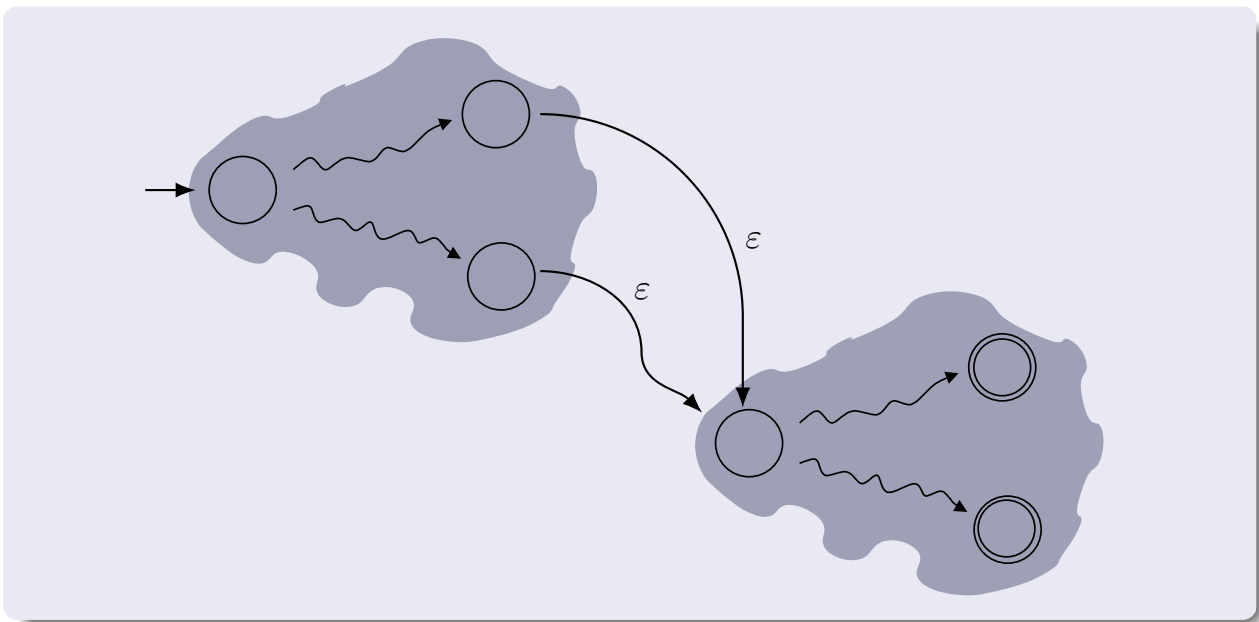
- Alternativamente, pode ser escrito de forma textual

## Concatenação de AF



- Como criar um AF que represente a concatenação destes dois AF?

## Concatenação de AF



- O estado inicial passa a ser o estado inicial do AF da esquerda
- Os estados de aceitação são apenas os estados de aceitação do AF da direita
- acrescentam-se transições- $\epsilon$  dos (antigos) estados de aceitação do AF da esquerda para o estado inicial do AF da direita

## Concatenação de AF: definição

$\mathcal{D}$  Seja  $M_1 = (A, Q_1, q_1, \delta_1, F_1)$  e  $M_2 = (A, Q_2, q_2, \delta_2, F_2)$  dois autómatos (AFD ou AFND) quaisquer.

O AFND  $M = (A, Q, q_0, \delta, F)$ , onde

$$Q = Q_1 \cup Q_2$$

$$q_0 = q_1$$

$$F = F_2$$

$$\delta = \delta_1 \cup \delta_2 \cup (F_1 \times \{\varepsilon\} \times \{q_2\})$$

implementa a concatenação de  $M_1$  e  $M_2$ , ou seja,  
 $L(M) = L(M_1) \cdot L(M_2)$ .

## Concatenação de AF: exemplo

$\mathcal{Q}$  Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça  $L = L_1 \cdot L_2$ .

$\mathcal{R}$

- 
- Como criar um AF que represente a concatenação de  $L_1$  com  $L_2$ ?

## Concatenação de AF: exemplo

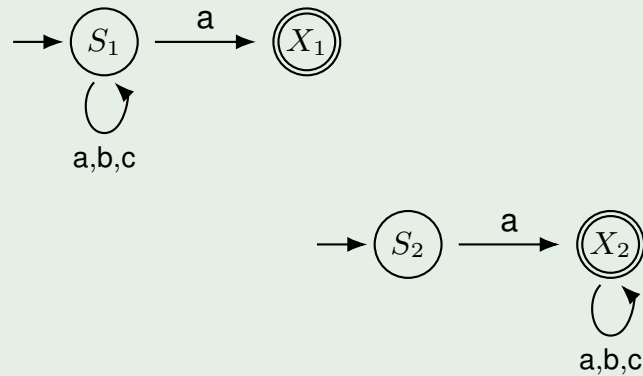
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça  $L = L_1 \cdot L_2$ .

R



- Constroi-se AF para as linguagens  $L_1$  e  $L_2$

## Concatenação de AF: exemplo

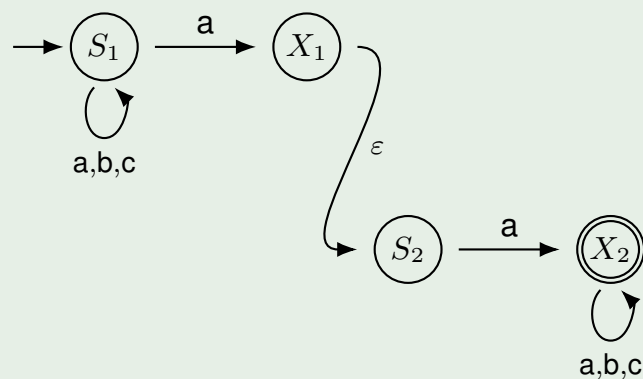
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

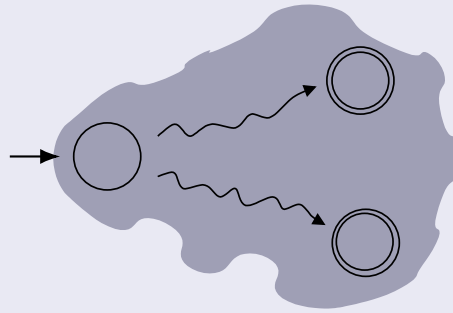
Determine um AF que reconheça  $L = L_1 \cdot L_2$ .

R



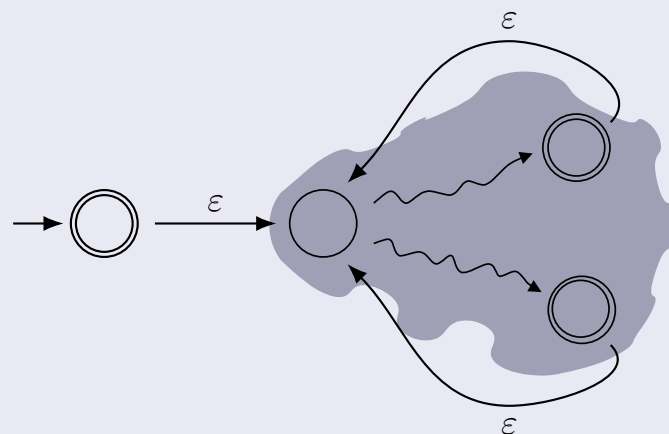
- $X_1$  deixa de ser de aceitação;  $S_2$  deixa de ser de entrada
- acrescenta-se uma transição- $\epsilon$  de  $X_1$  para  $S_2$

## Fecho de AF



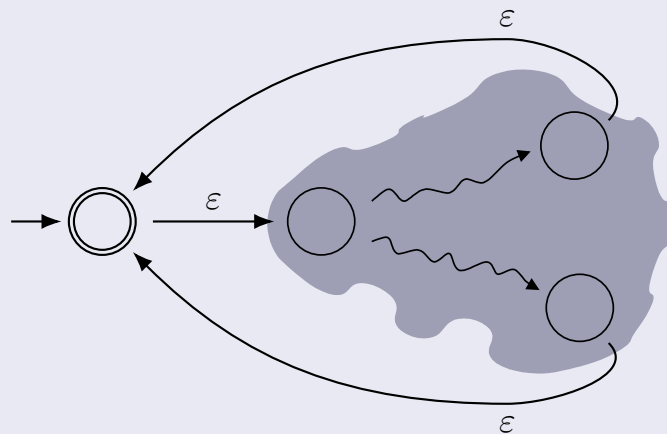
- Como criar um AF que represente o fecho deste AF?

## Fecho de AF



- acrescenta-se um novo estado que passa a ser o inicial
- o novo estado inicial é de aceitação
- acrescentam-se transições- $\epsilon$  dos estados de aceitação do AF para o estado inicial original

## Fecho de AF



- acrescenta-se um novo estado que passa a ser o inicial
  - o novo estado inicial é de aceitação
  - ou acrescentam-se transições- $\epsilon$  dos estados de aceitação do AF para o novo estado inicial (caso em que antigos estados de aceitação podem deixar de o ser)
- ◇ Note que em geral não se pode fundir o novo estado inicial com o antigo

## Fecho de AF: definição

**D** Seja  $M_1 = (A, Q_1, q_1, \delta_1, F_1)$  um autómato (AFD ou AFND) qualquer. O AFND  $M = (A, Q, q_0, \delta, F)$ , onde

$$Q = Q_1 \cup \{q_0\}$$

$$F = \{q_0\}$$

$$\delta = \delta_1 \cup (F_1 \times \{\epsilon\} \times \{q_0\}) \cup \{(q_0, \epsilon, q_1)\}$$

implementa o fecho de  $M_1$ , ou seja,  $L(M) = L(M_1)^*$ .

- Em alternativa poder-se-á considerar que  $F = F_1 \cup \{q_0\}$  e que de  $F_1$  as novas transições- $\epsilon$  se dirigem a  $q_1$

## Fecho de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , seja

$$L_1 = \{a\omega \mid \omega \in A^*\}$$

Determine o AFND que reconhece a linguagem  $L_1^*$ .

R

- Como criar um AF que represente o fecho de  $L_1$ ?

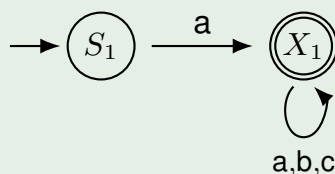
## Fecho de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , seja

$$L_1 = \{a\omega \mid \omega \in A^*\}$$

Determine o AFND que reconhece a linguagem  $L_1^*$ .

R



- Constroi-se um AF para  $L_1$



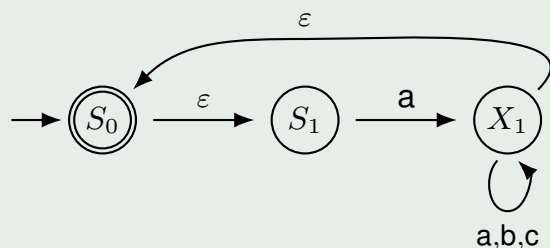
## Fecho de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , seja

$$L_1 = \{a\omega \mid \omega \in A^*\}$$

Determine o AFND que reconhece a linguagem  $L_1^*$ .

R



- acrescenta-se um novo estado ( $S_0$ ), que passa a ser o inicial e é de aceitação
- liga-se este estado ao  $S_1$  (inicial anterior) por uma transição- $\varepsilon$
- liga-se o estado  $X_1$  (aceitação anterior) ao  $S_0$  (novo inicial)
- $X_1$  deixa (pode deixar) de ser de aceitação

## Intersecção de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = L_1 \cap L_2$ .

R

- Como criar um AF que represente a intersecção de  $L_1$  e  $L_2$ ?

## Intersecção de AF: exemplo

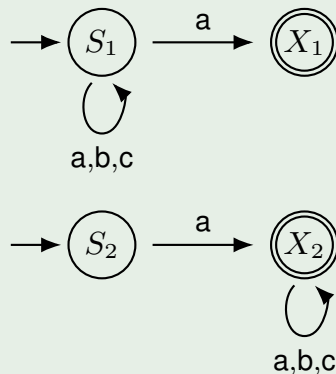
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = L_1 \cap L_2$ .

R



- Constroi-se AF para as linguagens  $L_1$  e  $L_2$

## Intersecção de AF: exemplo

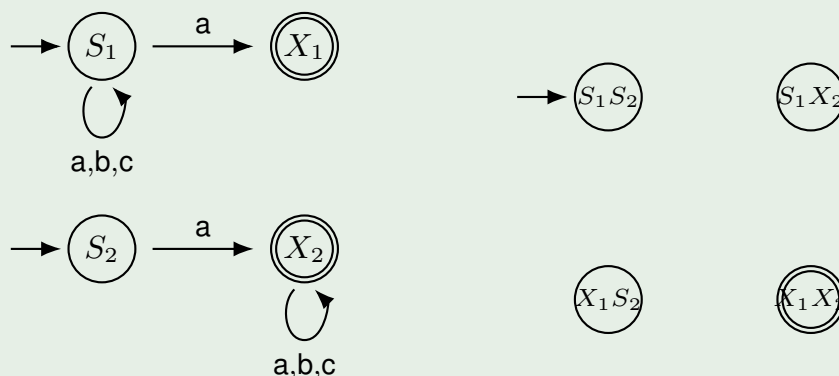
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = L_1 \cap L_2$ .

R



- Definem-se os estados que resultam do produto cartesiano  $\{S_1, X_1\} \times \{S_2, X_2\}$
- Mas, alguns podem não ser alcançáveis

## Intersecção de AF: exemplo

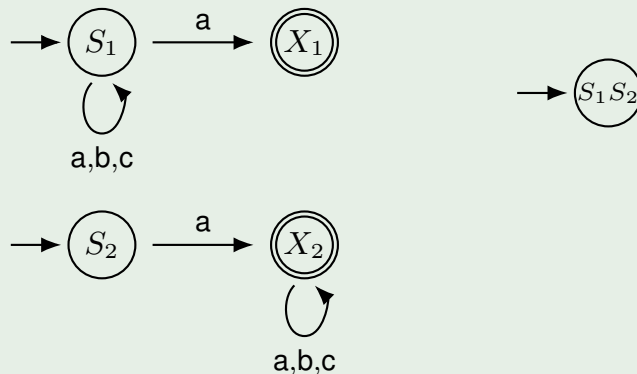
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = L_1 \cap L_2$ .

R



- Pelo que começemos apenas pelo  $S_1S_2$ , que corresponde ao estado inicial

## Intersecção de AF: exemplo

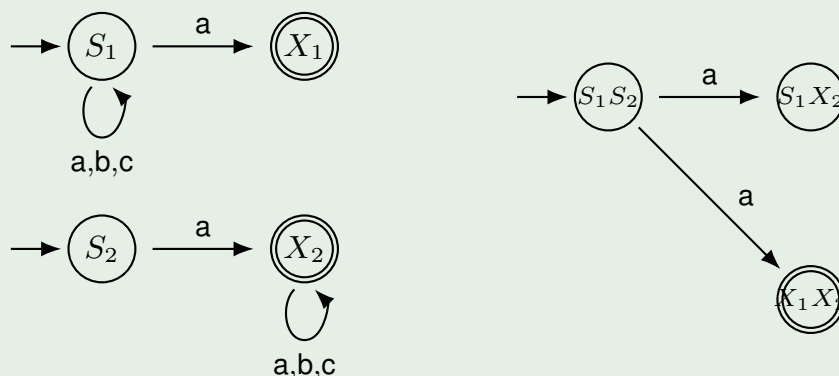
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = L_1 \cap L_2$ .

R



- de  $S_1 \xrightarrow{a} S_1$  e  $S_2 \xrightarrow{a} X_2$  aparece  $S_1S_2 \xrightarrow{a} S_1X_2$
- de  $S_1 \xrightarrow{a} X_1$  e  $S_2 \xrightarrow{a} X_2$  aparece  $S_1S_2 \xrightarrow{a} X_1X_2$

## Intersecção de AF: exemplo

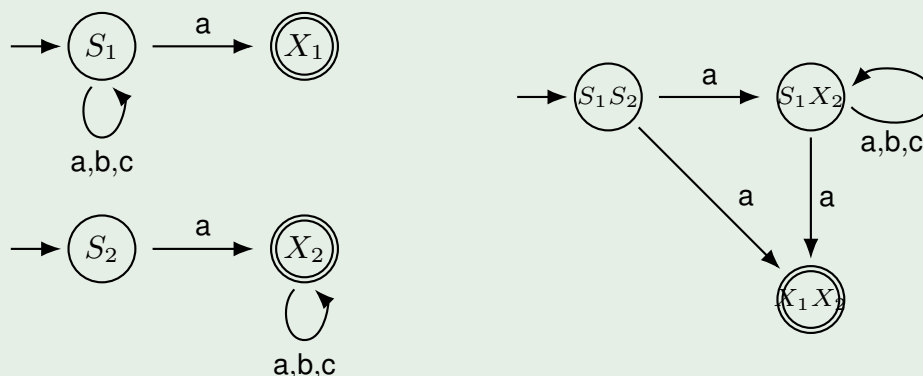
Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

$$L_2 = \{a\omega \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = L_1 \cap L_2$ .

R



- de  $S_1 \xrightarrow{x} S_1$  e  $X_2 \xrightarrow{x} X_2$  aparece  $S_1X_2 \xrightarrow{x} S_1X_2$ , para  $x \in \{a, b, c\}$
- de  $S_1 \xrightarrow{a} X_1$  e  $X_2 \xrightarrow{a} X_2$  aparece  $S_1X_2 \xrightarrow{a} X_1X_2$

## Intersecção de AF: definição

D Seja  $M_1 = (A, Q_1, q_1, \delta_1, F_1)$  e  $M_2 = (A, Q_2, q_2, \delta_2, F_2)$  dois autómatos (AFD ou AFND) quaisquer.

O AFND  $M = (A, Q, q_0, \delta, F)$ , onde

$$Q = Q_1 \times Q_2$$

$$q_0 = (q_1, q_2)$$

$$F = F_1 \times F_2$$

$$\delta \subseteq (Q_1 \times Q_2) \times A_\epsilon \times (Q_1 \times Q_2)$$

sendo  $\delta$  definido de modo que

$((q_i, q_j), a, (q'_i, q'_j)) \in \delta$  se e só se  $(q_i, a, q'_i) \in \delta_1$  e  $(q_j, a, q'_j) \in \delta_2$ ,  
implementa intersecção de  $M_1$  e  $M_2$ , ie.,  $L(M) = L(M_1) \cap L(M_2)$ .

## Complementação de AF

Q Sobre o alfabeto  $A = \{a, b, c\}$ , seja

$$L_1 = \{a\omega \mid \omega \in A^*\}$$

Determine um AF que reconheça a linguagem  $\overline{L_1}$ .

R

- Para se obter o complementar de um autômato finito determinista (em sentido estrito, ie. com todos os estados representados) basta complementar o conjunto de aceitação
- Para o caso de um autômato finito não determinista **é preciso** calcular o determinista equivalente e complementá-lo.

## Complementação de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = \overline{L_1}$ .

R

- 
- Como criar um AF que represente a intersecção de  $L_1$  e  $L_2$ ?

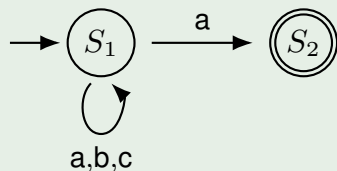
## Complementação de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = \overline{L_1}$ .

R



- Considere-se um AFND para a linguagem  $L_1$

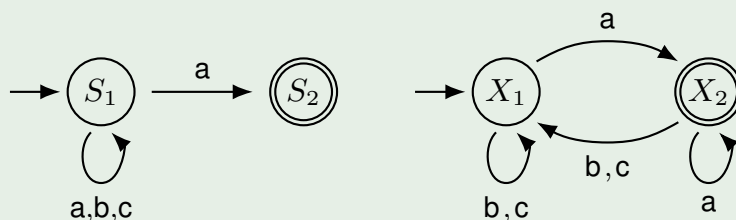
## Complementação de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = \overline{L_1}$ .

R



- Obtenha-se um determinista equivalente

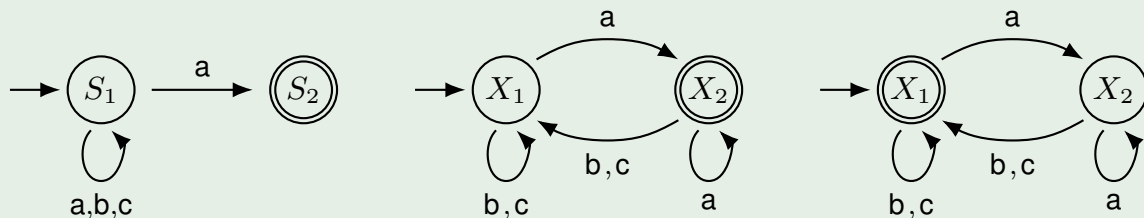
## Complementação de AF: exemplo

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{\omega a \mid \omega \in A^*\}$$

Determine um AFD ou AFND que reconheça  $L = \overline{L_1}$ .

R



- Complemente-se os estados de aceitação

## Operações sobre AF: exercício

Q Sobre o alfabeto  $A = \{a, b, c\}$ , sejam  $L_1$  e  $L_2$  as duas linguagens seguintes:

$$L_1 = \{v\omega \mid v \in \{a, b\} \wedge \omega \in A^*\} \quad (\text{palavras começadas por } a \text{ ou } b)$$

$$L_2 = \{\omega \in A^* \mid \#(a, \omega) \bmod 2 = 0\} \quad (\text{palavras com um número par de } a)$$

Determine AF que reconheça a linguagem

- $L_1$
- $L_2$
- $L_3 = L_1 \cup L_2$
- $L_4 = L_1 \cdot L_2$
- $L_6 = \overline{L_1} \cap L_2$
- $L_7 = \overline{L_2}$
- $L_8 = (L_4 \cup L_3)^*$

## Equivalência entre ER e AF

- A classe das linguagens cobertas por expressões regulares (ER) é a mesma que a classe das linguagens cobertas por autómatos finitos (AF)
- Logo:
  - Se  $e$  é uma ER, então  $\exists M \in AF : L(M) = L(e)$
  - Se  $M$  é um AF, então  $\exists e \in ER : L(e) = L(M)$
- Isto introduz duas operações:
  - Como converter uma ER num AF equivalente
  - Como converter um AF numa ER equivalente

## Conversão de uma ER num AF

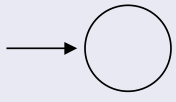
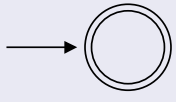
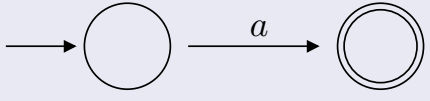
### Abordagem

- Já se viu anteriormente que uma expressão regular qualquer é:
  - ou um elemento primitivo;
  - ou uma expressão do tipo  $e_1|e_2$ , sendo  $e_1$  e  $e_2$  duas expressões regulares quaisquer
  - ou uma expressão do tipo  $e_1e_2$ , sendo  $e_1$  e  $e_2$  duas expressões regulares quaisquer
  - ou uma expressão do tipo  $e^*$ , sendo  $e$  uma expressão regular qualquer
- Já se viu anteriormente como realizar a **reunião**, a **concatenação** e o **fecho** de autómatos finitos
- Então, se se identificar autómatos finitos equivalentes às expressões regulares primitivas, tem-se o problema da conversão de uma expressão regular para um autómato finito resolvido



# Conversão de uma ER num AF

## Autômatos dos elementos primitivos

expressão regular	autômato finito
$\emptyset$	
$\varepsilon$	
$a$	

- Na realidade, o autômato referente a  $\varepsilon$  pode ser obtido aplicando o fecho ao autômato de  $\emptyset$

# Conversão de uma ER num AF

## Algoritmo de conversão

- Se a expressão regular é do tipo primitivo, o autômato correspondente pode ser obtido da tabela anterior
- Se é do tipo  $e^*$ , aplica-se este mesmo algoritmo na obtenção de um autômato equivalente à expressão regular  $e$  e, de seguida, aplica-se o fecho de autômatos
- Se é do tipo  $e_1e_2$ , aplica-se este mesmo algoritmo na obtenção de autômatos para as expressões  $e_1$  e  $e_2$  e, de seguida, aplica-se a concatenação de autômatos
- Finalmente, se é do tipo  $e_1|e_2$ , aplica-se este mesmo algoritmo na obtenção de autômatos para as expressões  $e_1$  e  $e_2$  e, de seguida, aplica-se a reunião de autômatos

- Na realidade, o algoritmo corresponde a um processo de decomposição arbórea a partir da raiz seguido de um processo de construção arbórea a partir das folhas

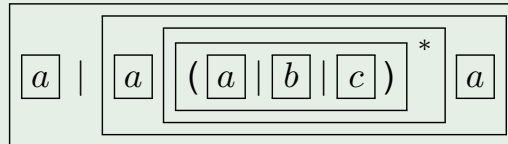
# Conversão de uma ER num AF

## Exemplo

Q Construa um autômato equivalente à expressão regular  $e = a|a(a|b|c)^*a$

R

1 Decomposição



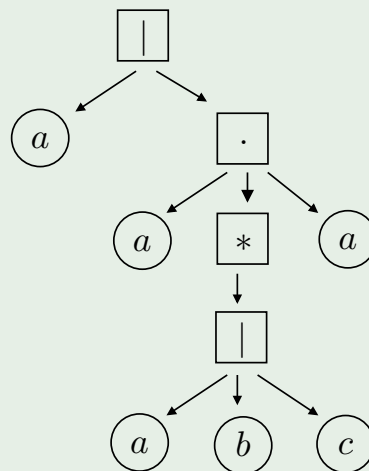
# Conversão de uma ER num AF

## Exemplo

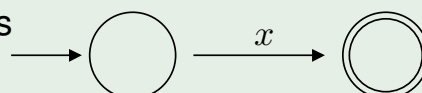
Q Construa um autômato equivalente à expressão regular  $e = a|a(a|b|c)^*a$

R

1 Decomposição



2 Autômatos primitivos

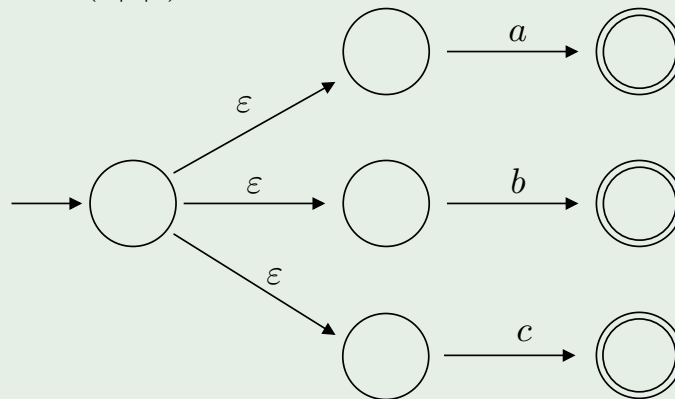


com  $x = \{a, b, c\}$

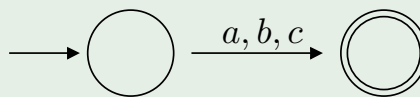
## Conversão de uma ER num AF

Exemplo

3 Reunião para obter  $(a|b|c)$



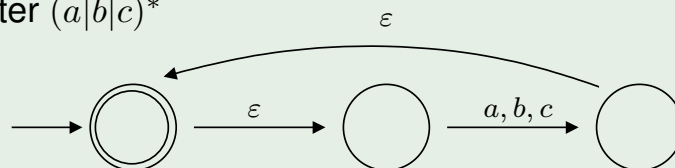
4 Simplificando



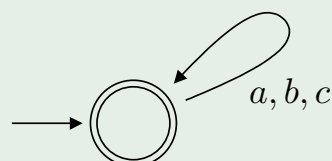
## Conversão de uma ER num AF

Exemplo

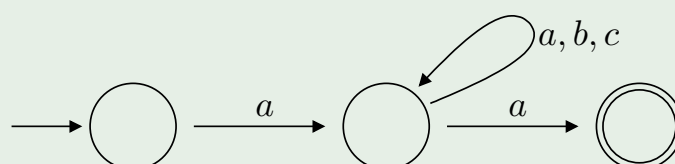
5 Fecho para obter  $(a|b|c)^*$



6 Simplificando



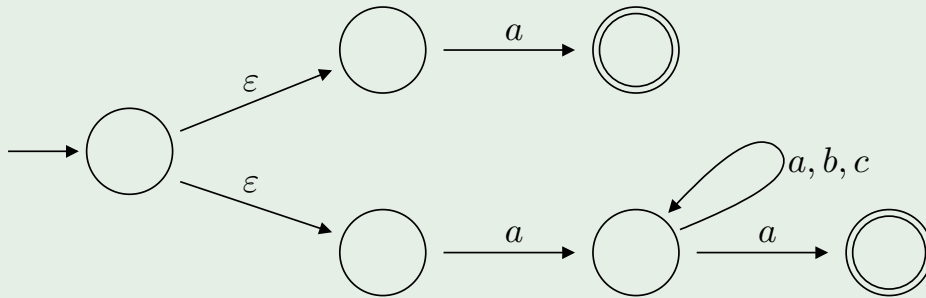
7 Concatenação (já com simplificação) para obter  $a(a|b|c)^*a$



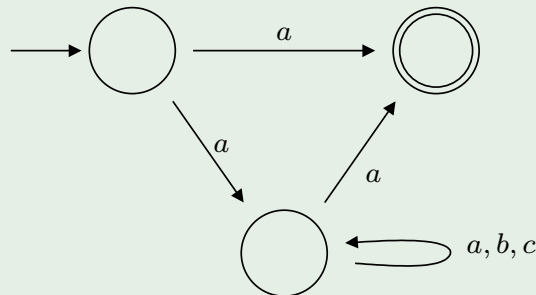
# Conversão de uma ER num AF

## Exemplo

8 Finalmente obtenção de  $a|a(a|b|c)^*a$



9 Simplificando



# Autômato finito generalizado (AFG)

## Definição

$\mathcal{D}$  Um **autômato finito generalizado** (AFG) é um quintuplo

$M = (A, Q, q_0, \delta, F)$ , em que:

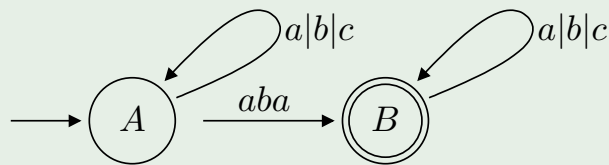
- $A$  é o alfabeto de entrada
- $Q$  é um conjunto finito não vazio de estados
- $q_0 \in Q$  é o estado inicial
- $\delta \subseteq (Q \times E \times Q)$  é a relação de transição entre estados, sendo  $E$  o conjunto das expressões regulares definidas sobre  $A$
- $F \subseteq Q$  é o conjunto dos estados de aceitação

- A diferença em relação ao AFD e AFND está na definição da relação  $\delta$ . Neste caso as etiquetas são *expressões regulares*
- Com base nesta definição os AFD e os AFND são autômatos finitos generalizados

## Autômato finito generalizado (AFG)

### Exemplo

- O AFG seguinte representa o conjunto das palavras, definidas sobre o alfabeto  $A = \{a, b, c\}$ , que contêm a sub-palavra  $aba$

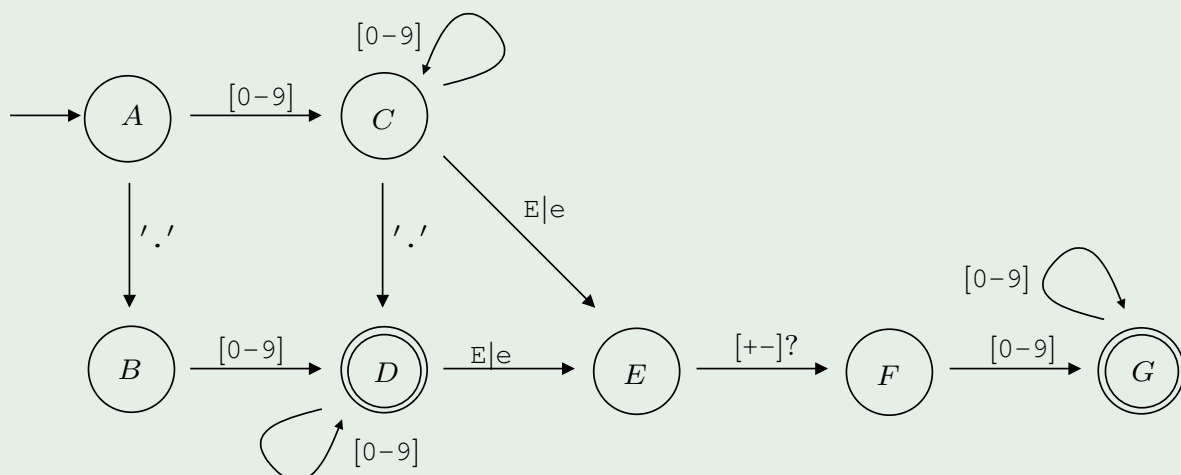


- Note que a etiqueta das transições  $A \rightarrow A$  e  $B \rightarrow B$  é  $a|b|c$  (uma expressão regular) e não  $a, b, c$  (que representa 3 transições, uma em  $a$ , uma em  $b$  e uma em  $c$ )

## Autômato finito generalizado (AFG)

### Exemplo

- O AFG seguinte representa as constantes reais em  $\mathbb{C}$

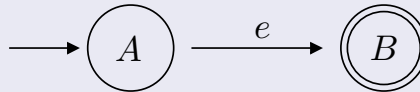


- Note que se usou  $.,$  e não  $.$ , porque o último é uma expressão regular que representa qualquer letra do alfabeto

# Conversão de um AFG numa ER

## Abordagem

$\mathcal{D}$  UM AFG com a forma



designa-se por **autômato finito generalizado reduzido**

- Note que:
  - O estado  $A$  não é de aceitação e não tem transições a chegar
  - O estado  $B$  é de aceitação e não tem transições a sair
- Se se reduzir um AFG à forma anterior,  $e$  é uma expressão regular equivalente ao autômato
- O processo de conversão resume-se assim à conversão de AFG à forma reduzida

# Conversão de um AFG numa ER

## Algoritmo de conversão

- 1 transformação de um AFG noutro cujo estado inicial **não tenha transições a chegar**
  - Se necessário, acrescenta-se um novo estado inicial com uma transição em  $\varepsilon$  para o antigo
- 2 transformação de um AFG noutro com **um único estado de aceitação, sem transições de saída**
  - Se necessário, acrescenta-se um novo estado, que passa a ser o único de aceitação, que recebe transições em  $\varepsilon$  dos anteriores estados de aceitação, que deixam de o ser
- 3 Eliminação dos estados intermédios
  - Os estados são eliminados um a um, em processos de transformação que mantêm a equivalência

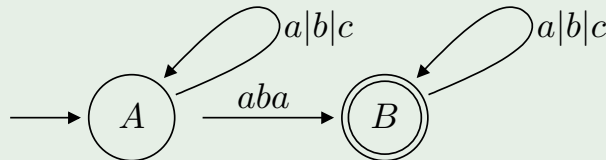
## Conversão de um AFG numa ER

Ilustração com um exemplo

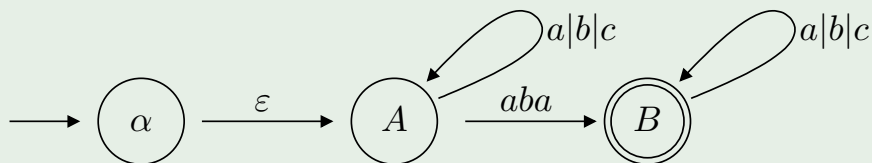
- 1 transformação de um AFG noutro cujo estado inicial **não tenha transições a chegar**

- Se necessário, acrescenta-se um novo estado inicial com uma transição em  $\varepsilon$  para o antigo

antes



depois



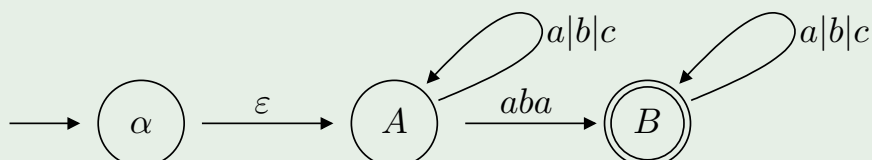
## Conversão de um AFG numa ER

Ilustração com um exemplo

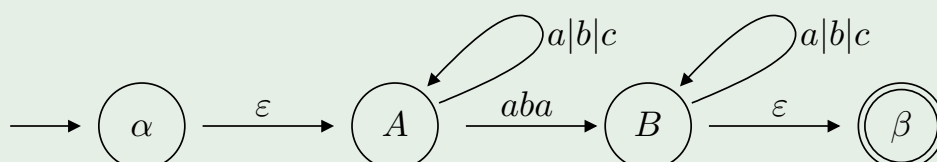
- 2 transformação de um AFG noutro com **um único estado de aceitação e sem transições de saída**

- Se necessário, acrescenta-se um novo estado, que passa a ser o único de aceitação, que recebe transições em  $\varepsilon$  dos anteriores estados de aceitação, que deixam de o ser

antes



depois



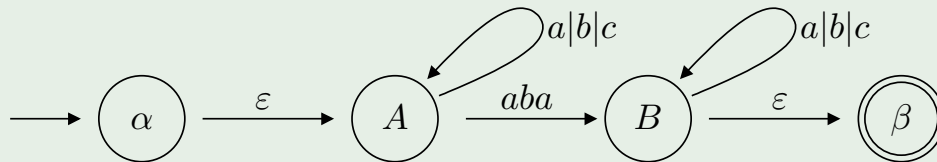
## Conversão de um AFG numa ER

### Ilustração com um exemplo

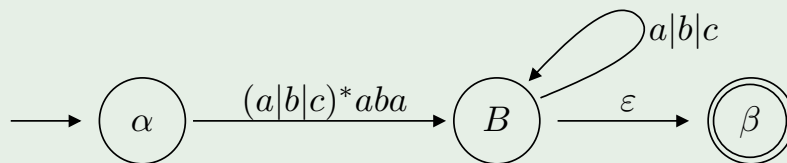
#### ③ Eliminação dos restantes estados

- Os estados são eliminados um a um, em processos de transformação que mantêm a equivalência
- Comece-se pelo estado A

antes



depois da eliminação de A



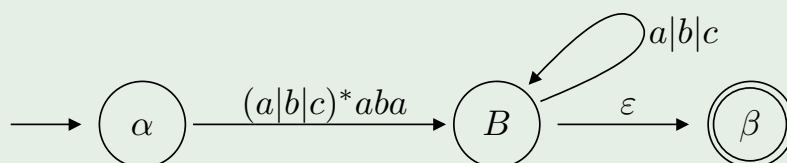
## Conversão de um AFG numa ER

### Ilustração com um exemplo

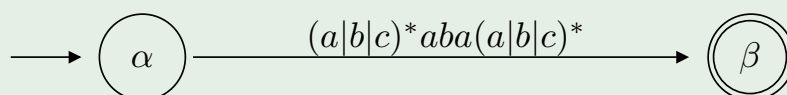
#### ③ Eliminação dos restantes estados

- Os estados são eliminados um a um, em processos de transformação que mantêm a equivalência
- Remova-se agora o estado B

depois da eliminação de A



depois da eliminação de A, seguido da eliminação de B



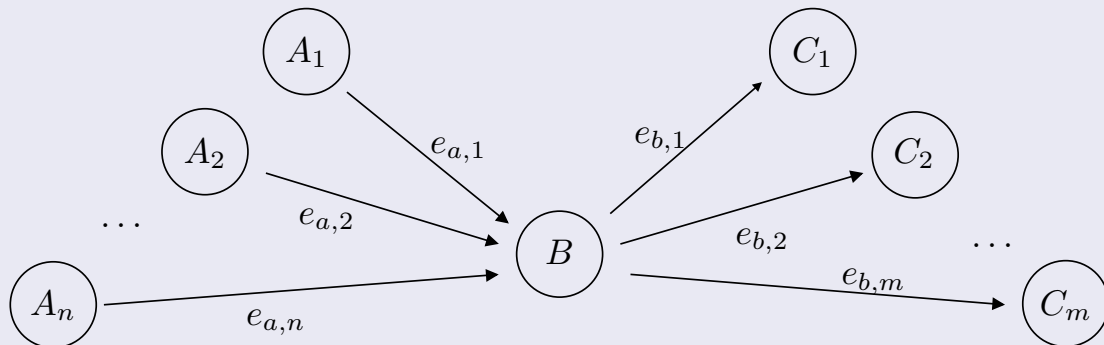
- Sendo  $(a|b|c)^*aba(a|b|c)^*$  a expressão regular pretendida



## Conversão de um AFG numa ER

### Algoritmo de eliminação de um estado

- Caso em que o estado a eliminar ( $B$ ) **não tem** transições de si para si

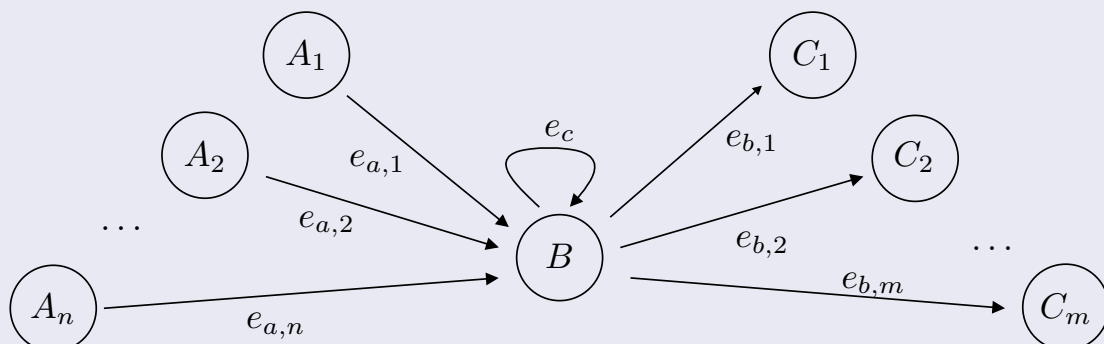


- Pode acontecer que haja  $A_i = C_j$
- Para ir de  $A_i$  para  $C_j$  através de  $B$ , para  $i = 1, 2, \dots, n$  e  $j = 1, 2, \dots, m$ , é preciso uma palavra que encaixe na expressão regular  $(e_{a,i})(e_{b,j})$
- Então, se se retirar  $B$ , é preciso acrescentar uma transição de  $A_i$  para  $C_j$  que contemple essas palavras, ou seja, com a etiqueta  $(e_{a,i})(e_{b,j})$
- Esta transição fica em paralelo com uma que já exista

## Conversão de um AFG numa ER

### Algoritmo de eliminação de um estado

- Caso em que o estado a eliminar ( $B$ ) **tem** transições de si para si

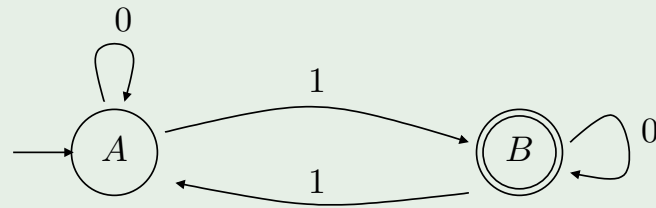


- Pode acontecer que haja  $A_i = C_j$
- Para ir de  $A_i$  para  $C_j$  através de  $B$ , para  $i = 1, 2, \dots, n$  e  $j = 1, 2, \dots, m$ , é preciso uma palavra que encaixe na expressão regular  $(e_{a,i})(e_c)^*(e_{b,j})$
- Então, se se retirar  $B$ , é preciso acrescentar uma transição de  $A_i$  para  $C_j$  que contemple essas palavras, ou seja com etiqueta  $(e_{a,i})(e_c)^*(e_{b,j})$
- Esta transição fica em paralelo com uma que já exista

## Conversão de um AFG numa ER

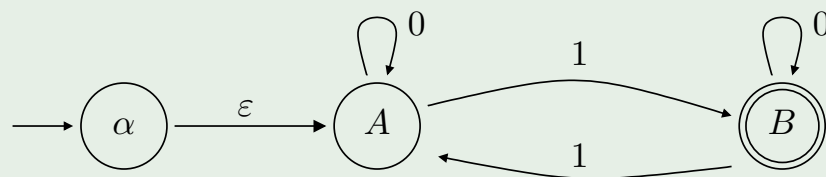
### Exercício

Q Obtenha uma ER equivalente ao AF seguinte



R Aplique-se passo a passo o algoritmo de conversão

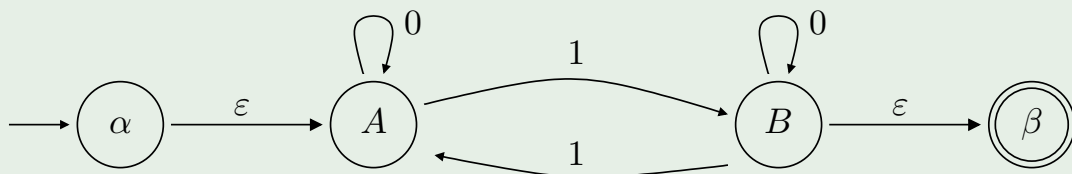
- Porque o estado inicial possui uma transição a entrar, deve substituir-se o estado inicial, de acordo com o passo 1 do algoritmo



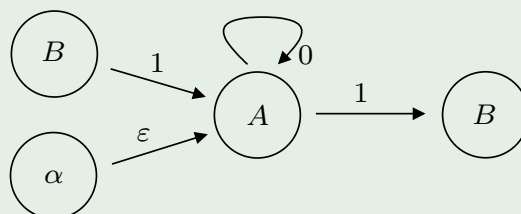
## Exemplo de conversão de um AFG numa ER

### Exercício

- Porque o estado de aceitação possui uma transição a sair, deve-se aplicar o passo 2 do algoritmo de conversão



- Elimine-se o estado A. Para isso é preciso ver os segmentos de caminhos que passam por A.

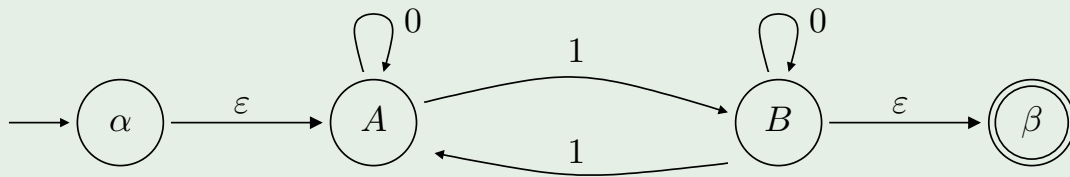


- Note que B aparece à esquerda e à direita

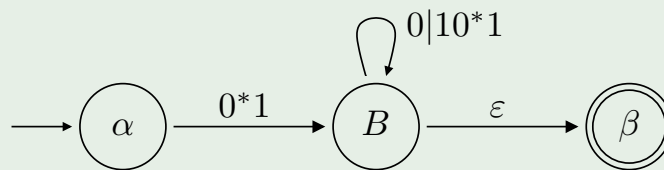
## Exemplo de conversão de um AFG numa ER

### Exercício

- Porque o estado de aceitação possui uma transição a sair, deve-se aplicar o passo 2 do algoritmo de conversão



- Eliminando o estado  $A$  obtém-se



- Finalmente, eliminando o estado  $B$  obtém-se a ER  $0^*1(0|10^*1)^*$

## Equivalência entre GR e AF

- A classe das linguagens cobertas por gramáticas regulares (ER) é a mesma que a classe das linguagens cobertas por autómatos finitos (AF)
- Logo:
  - Se  $G$  é uma ER, então  $\exists M \in AF : L(M) = L(G)$
  - Se  $M$  é um AF, então  $\exists G \in ER : L(G) = L(M)$
- Isto introduz duas operações:
  - Como converter um AF numa GR equivalente
  - Como converter uma GR num AF equivalente

# Conversão de um AF numa GR

## Procedimento de conversão

$\mathcal{A}$  Seja  $M = (A, Q, q_0, \delta, F)$  um autómato finito qualquer.

A GR  $G = (T, N, P, S)$ , onde

$$T = A$$

$$N = Q$$

$$S = q_0$$

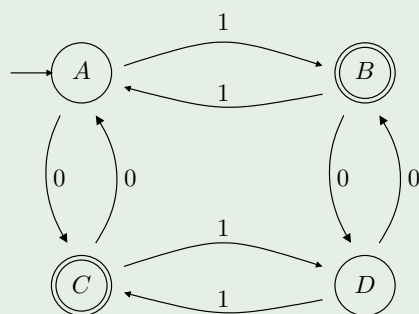
$$P = \{p \rightarrow a q : p, q \in Q \wedge a \in T \wedge (p, a, q) \in \delta\} \\ \cup \{p \rightarrow \varepsilon : p \in F\}$$

representa a mesma linguagem que  $M$ , isto é,  $L(G) = L(M)$ .

# Conversão de um AF numa GR

## Exemplo

$\mathcal{Q}$  Determine uma GR equivalente ao AF



$\mathcal{R}$

$$A \rightarrow 0 C \mid 1 B$$

$$B \rightarrow 0 D \mid 1 A \mid \varepsilon$$

$$C \rightarrow 0 A \mid 1 D \mid \varepsilon$$

$$D \rightarrow 0 B \mid 1 C$$

# Conversão de uma GR num AFG

## Procedimento de conversão

Seja  $G = (T, N, P, S)$  uma gramática regular qualquer.

O AF  $M = (A, Q, q_0, \delta, F)$ , onde

$$A = T$$

$$Q = N \cup \{q_f\}, \quad \text{com } q_f \notin N$$

$$q_0 = S$$

$$F = \{q_f\}$$

$$\delta = \{(q_i, e, q_j) : q_i, q_j \in N \wedge e \in T^* \wedge q_i \rightarrow e q_j \in P\} \\ \cup \{(q, e, q_f) : q \in N \wedge e \in T^* \wedge q \rightarrow e \in P\}$$

representa a mesma linguagem que  $G$ , isto é,  $L(M) = L(G)$ .

# Conversão de uma GR num AFG

## Exemplo

Q Determine um AFG equivalente à GR

$$S \rightarrow a S \mid b S \mid c S \mid a b a X$$

$$X \rightarrow a X \mid b X \mid c X \mid \varepsilon$$

R

Sendo  $M = (A, Q, q_0, \delta, F)$  o AFG equivalente, tem-se

$$A = \{a, b, c\}$$

$$Q = \{S, X, q_f\}$$

$$q_0 = S$$

$$\delta = \{(S, a, S), (S, b, S), (S, c, S), (S, aba, X), \\ (X, a, X), (X, b, X), (X, c, X), (X, \varepsilon, q_f)\}$$

$$F = \{q_f\}$$

