Linguagens Formais e Autómatos / Compiladores

Autómatos Finitos (AF)

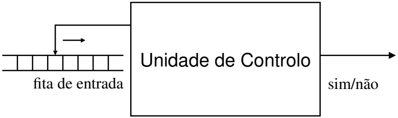
Artur Pereira <artur@ua.pt>,
Miguel Oliveira e Silva <mos@ua.pt>

DETI, Universidade de Aveiro

Sumário

AUTÓMATO FINITO

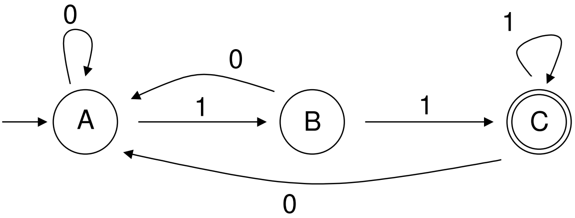
Um autómato finito é um mecanismo reconhecedor das palavras de uma linguagem regular



- A unidade de controlo é baseada na noção de estado e na 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

AUTÓMATO FINITO DETERMINISTA

Um autómato finito determinista é um autómato finito

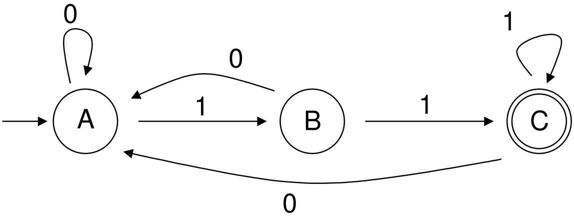


onde

- 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;
- uma dada palavra sobre o alfabeto faz o sistema avançar do estado inicial a um estado final, determinando este a aceitação ou rejeição da palavra.

Exemplo de autómato finito determinista

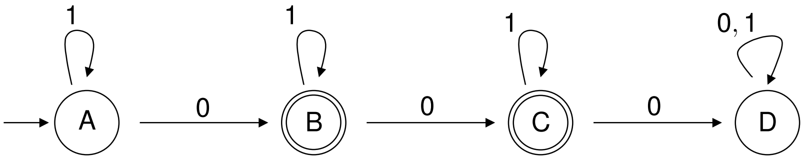
$\mathcal{Q}$ Que palavras binárias são reconhecidas pelo autómato seguinte?



$\mathcal{R}$ Todas as palavras terminadas em 11.

Exemplo de autómato finito determinista (2)

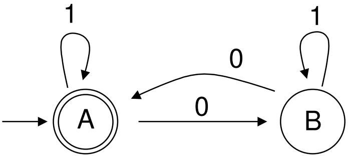
$\mathcal{Q}$ Que palavras binárias são reconhecidas pelo autómato seguinte?



$\mathcal{R}$ Todas as palavras com 1 ou 2 zeros.

Exemplo de autómato finito determinista

$\mathcal{Q}$ Que palavras binárias são reconhecidas pelo autómato seguinte?



$\mathcal{R}$ as sequências binárias com um número par de zeros.

Definição de autómato finito determinista

$\mathcal{D}$ Um **autómato finito determinista** (AFD) é um quíntuplo $M = (A, Q, q_0, \delta, F)$, em que:

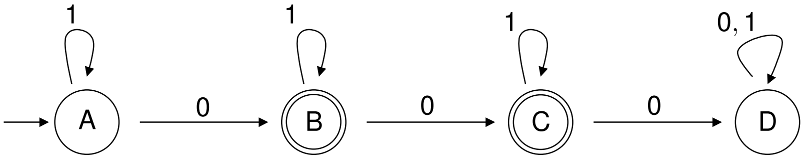
- A é o alfabeto de entrada;
- Q é um conjunto finito não vazio de estados;
- $q_0 \in Q$ é o estado inicial;
- $\delta : Q \times A \to Q$ é uma função que determina a transição entre estados; e
- $F \subseteq Q$ é o conjunto dos estados de aceitação.

$\mathcal{Q}$ Como representar a função δ ?

- Conjunto de triplos $\in Q \times A \times Q$
- Matriz de $|Q|$ linhas por $|A|$ colunas. As células contêm elementos de Q.

Exemplo de autómato finito determinista

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



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

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

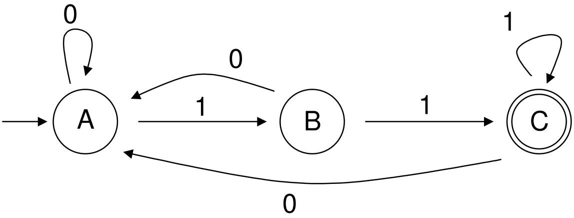
- $\delta = \{$
  $(A, 0, B), (A, 1, A),$
  $(B, 0, C), (B, 1, B),$
  $(C, 0, D), (C, 1, C),$
  $(D, 0, D), (D, 1, D)\}$

- $\delta =$

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

EXEMPLO DE AUTÓMATO FINITO DETERMINISTA (2)

$\mathcal{Q}$ 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

- 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 $i = 1, \cdots, n, \quad s_i = \delta(s_{i-1}, u_i)$;
  3. $s_n \in F$.

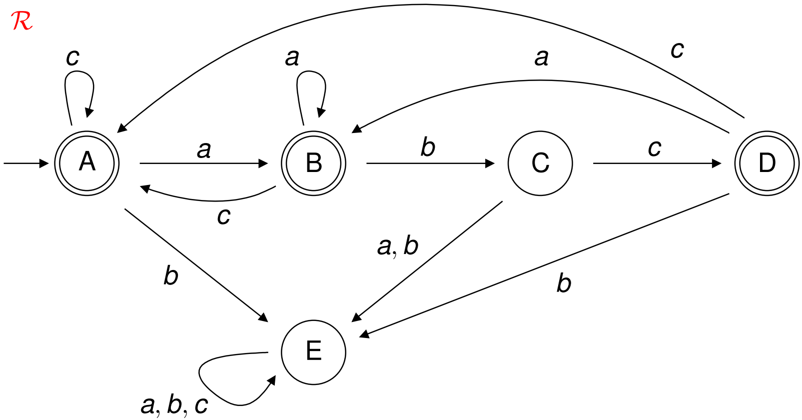
  Caso contrário diz-se que M **rejeita** a sequência de entrada.

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

PROJETO DE AUTÓMATO FINITO DETERMINISTA

$\mathcal{Q}$ Projete um AFD que reconheça as sequências definidas sobre o alfabeto $A = \{a, b, c\}$ que satisfazem o requisito de qualquer b ter um a imediatamente à sua esquerda e um c imediatamente à sua direita.

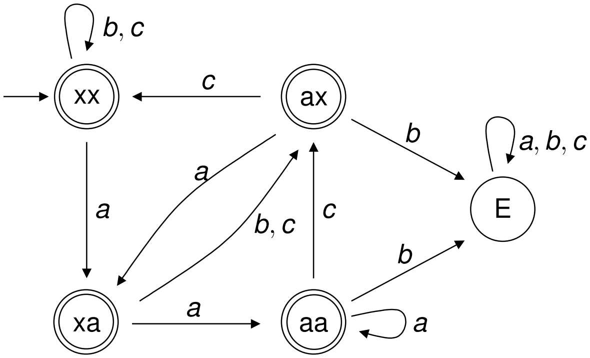
$\mathcal{R}$



Projeto de autómato finito determinista (2)

$\mathcal{Q}$ Sobre o alfabeto $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ considere a linguagem
$L = \{\omega \in A^* \; : \; (\omega_i = \mathtt{a}) \;\Rightarrow\; (\omega_{i+2} \neq \mathtt{b})\}$
Projecte um autómato que reconheça L.

$\mathcal{R}$

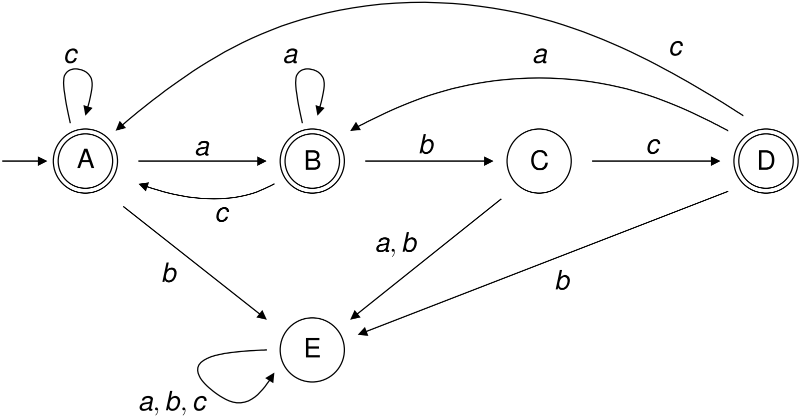


Projeto de autómato finito determinista (2)

$\mathcal{Q}$ Sobre o alfabeto $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ considere a linguagem
$L = \{\omega \in A^* \ :\ (\omega_i = \mathtt{a}) \ \Rightarrow\ (\omega_{i+2} = \mathtt{b})\}$
Projecte um autómato que reconheça L.

Redução de autómato finito determinista

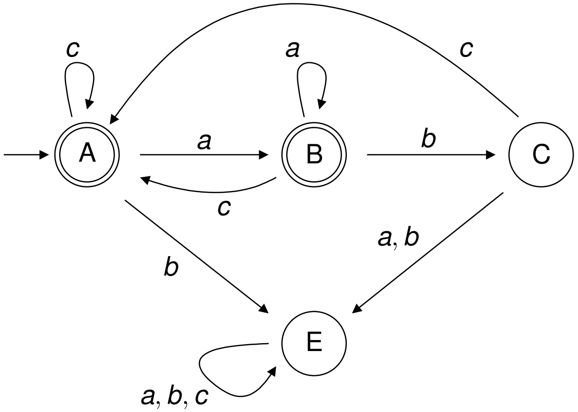
$\mathcal{Q}$ Considere o autómato seguinte e compare os estados A e D. Que pode concluir ?



$\mathcal{R}$ São equivalentes. Podem ser fundidos...

Redução de autómato finito determinista (2)

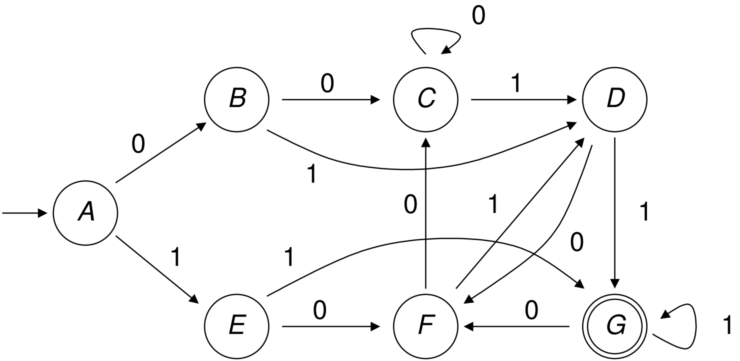
$\mathcal{R}$ ... O que resulta em



Este, pode provar-se que não tem estados equivalentes.

ALGORITMO DE REDUÇÃO DE AFD

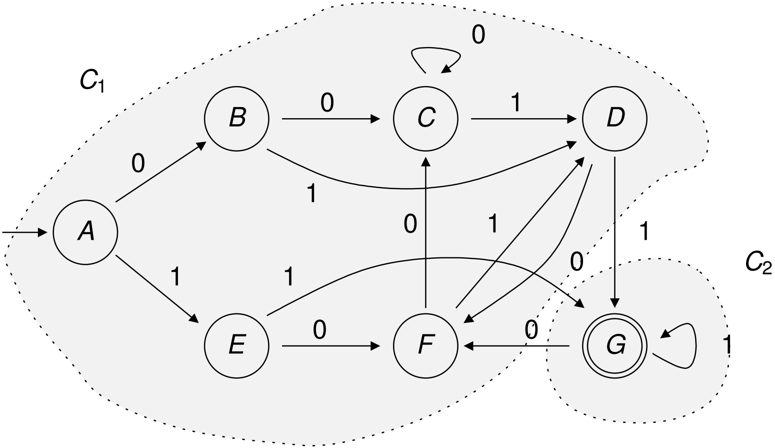
$\mathcal{Q}$ Pretende-se reduzir o AFD



$\mathcal{R}$ Primeiro, dividem-se os estados em aceitação e não-aceitação.

ALGORITMO DE REDUÇÃO DE AFD (2)

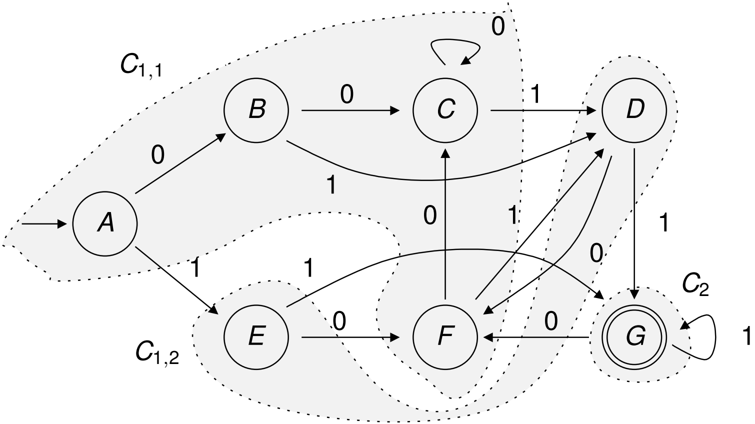
$\mathcal{R}$ Obtêm-se 2 classes, $C_1 = \{A, B, C, D, E, F\}$ e $C_2 = \{G\}$.



A classe C_1 tem que ser dividida.

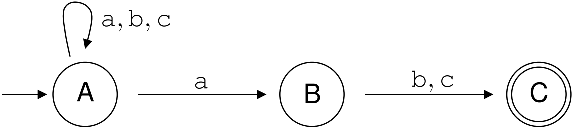
ALGORITMO DE REDUÇÃO DE AFD (3)

- Dividindo C_1 em $\{A, B, C, F\}$ e $\{D, E\}$ obtem-se



AUTÓMATO FINITO NÃO DETERMINISTA

Um autómato finito não determinista é um autómato finito



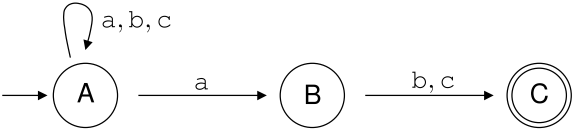
onde

- as transições estão associadas a símbolos individuais do alfabeto ou a ε;
- de cada estado saem *zero ou mais* transições por cada símbolo do alfabeto ou ε;
- há um estado inicial;
- há 0 ou mais estados de aceitação, que determinam as palavras aceites;
- uma dada palavra sobre o alfabeto faz o sistema avançar do estado inicial a zero ou mais estados finais, determinando estes a aceitação ou rejeição da palavra.

- As transições múltiplas permitem alternativas de reconhecimento.
- As transições ausentes representam quedas num estado de *morte* (estado não representado).

AFND: CAMINHOS ALTERNATIVOS

$\mathcal{Q}$ `abab` $\in L$?

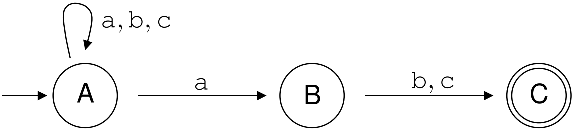


$\mathcal{R}$ Há 3 caminhos alternativos:

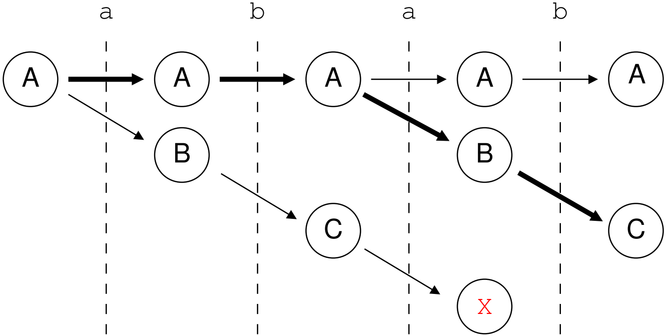
1. $A \xrightarrow{a} B \xrightarrow{b} C \xrightarrow{a}$ X
2. $A \xrightarrow{a} A \xrightarrow{b} A \xrightarrow{a} B \xrightarrow{b} C$
3. $A \xrightarrow{a} A \xrightarrow{b} A \xrightarrow{a} A \xrightarrow{b} A$

AFND: CAMINHOS ALTERNATIVOS

$\mathcal{Q}$ abab $\in L$?

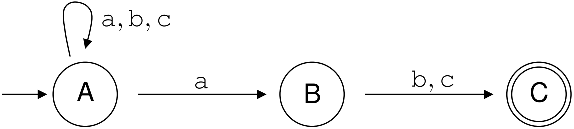


$\mathcal{R}$ Que se podem representar de forma arbórea



Exemplo de AFND

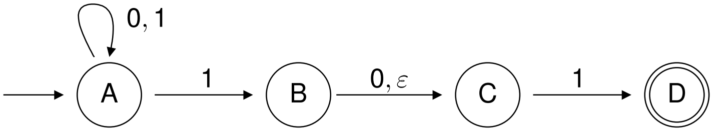
$\mathcal{Q}$ Que palavras são reconhecidas pelo autómato seguinte?



$\mathcal{R}$ $L = \{\omega \mathtt{a} x \; : \; \omega \in A^* \; \wedge \; x \in \{\mathtt{b}, \mathtt{c}\}\}$.

AFND: EXEMPLO COM TRANSIÇÕES-ε

$\mathcal{Q}$ `1011` $\in L$?



$\mathcal{Q}$ `1011` pertence à linguagem descrita pelo AFND ?

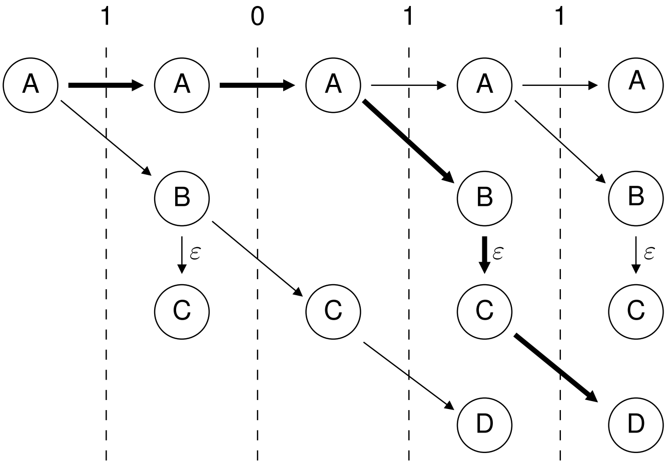
$\mathcal{R}$ Há 4 caminhos possíveis.

1. $A \xrightarrow{1} A \xrightarrow{0} A \xrightarrow{1} A \xrightarrow{1} A$
2. $A \xrightarrow{1} A \xrightarrow{0} A \xrightarrow{1} A \xrightarrow{1} B$
3. $A \xrightarrow{1} A \xrightarrow{0} A \xrightarrow{1} A \xrightarrow{1} B \xrightarrow{\varepsilon} C$
4. $A \xrightarrow{1} A \xrightarrow{0} A \xrightarrow{1} B \xrightarrow{\varepsilon} C \xrightarrow{1} D$

Há um caminho que conduz a D, logo pertence.

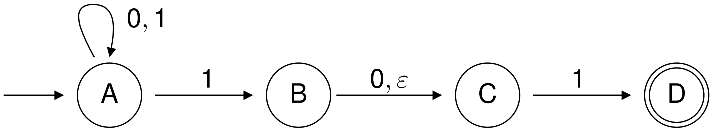
AFND: EXEMPLO (2)

$\mathcal{R}$ na forma arbórea ...



EXEMPLO DE AFND

$\mathcal{Q}$ Que palavras são reconhecidas pelo autómato seguinte?



$\mathcal{R}$ $L = \{\omega \in (\mathtt{0}, \mathtt{1})^* \ : \ \omega$ termina em 11 ou 101$\}$.

AFND: DEFINIÇÃO

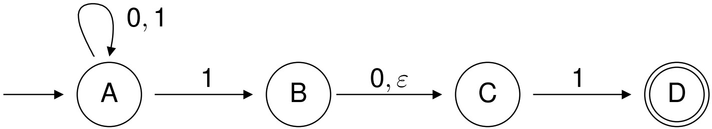
$\mathcal{D}$ Um **autómato finito não determinista** (AFND) é um quíntuplo $M = (A, Q, q_0, \delta, F)$, em que:

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

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

AFND: Exemplo (3)

$\mathcal{Q}$ Represente analiticamente o AFND



$\mathcal{R}$

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

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 $s_0, s_1, \cdots, s_n$, que satisfaça as seguintes condições:
  1. $s_0 = q_0$;
  2. qualquer que seja o $i = 1, \cdots, n$, $(s_{i-1}, u_i, s_i) \in \delta$;
  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 ε.

- Usar-se-á a notação $q_i \xrightarrow{u} q_j$ para representar a existência de uma palavra u que conduza do estado q_i ao estado q_j.
- Usando esta notação tem-se $L(M) = \{u \; : \; q_0 \xrightarrow{u} q_f \; \wedge \; q_f \in F\}$.

Equivalência entre AFD e AFND

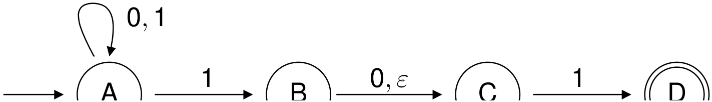
- A classe das linguagens cobertas por um AFD é a mesma que a classe das linguagens cobertas por um AFND
- Se M é um AFD, então $\exists_{M' \in AFND} \; : \; L(M') = L(M)$.
- Se M é um AFND, então $\exists_{M' \in AFD} \; : \; L(M') = L(M)$.

- Como determinar um AFND equivalente a um AFD dado ?
- Como determinar um AFD equivalente a um AFND dado ?

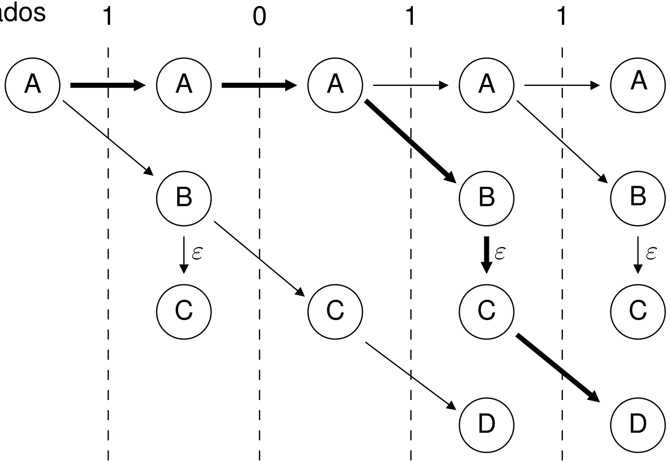
- Pelas definições de AFD e AFND, um AFD é um AFND.
  - Q, q_0 e F têm a mesma definição.
  - Nos AFD $\delta : Q \times A \to Q$.
  - Nos AFND $\delta \subset Q \times A_\varepsilon \times Q$
  - Mas, se $\delta : Q \times A \to Q$ então $\delta \subset Q \times A \times Q \subset Q \times A_\varepsilon \times Q$
  - Logo, um AFD é um AFND

Equivalência entre AFD e AFND

$\mathcal{Q}$ Como determinar um AFD equivalente ao AFND seguinte ?



$\mathcal{R}$ A árvore de reconhecimento aponta para sub-conjuntos de estados

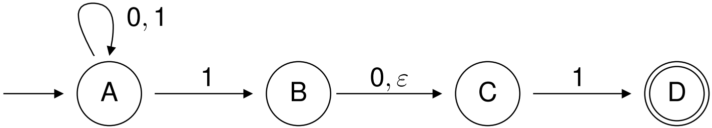


EQUIVALÊNCIA ENTRE AFD E AFND

- 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' é o subconjunto de Q constituído pelo estado inicial de M mais todos os alcançáveis a partir dele por ocorrências de εs
  - $F = \{f' \in \wp(Q) \ : \ f' \cap F \neq \emptyset\}$
  - $\delta' = \wp(Q) \times A \rightarrow \wp(Q)$, com $\delta'(q', a) = \bigcup_{q \in q'} \{\delta(q, a)\}$ fechado em ε.
- M e M' reconhecem a mesma linguagem.

Equivalência entre AFD e AFND

$\mathcal{Q}$ Determinar um AFD equivalente ao AFND seguinte ?

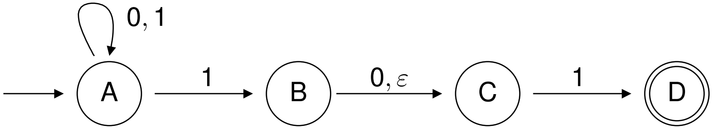


- $\delta' =$

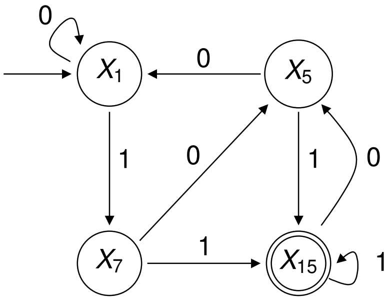
estado	0	1		estado	0	1
X_0	X_0	X_0		X_1	X_1	X_7
X_2	X_4	X_0		X_3	X_5	X_7
X_4	X_0	X_8		X_5	X_1	X_{15}
X_6	X_4	X_8		X_7	X_5	X_{15}
X_8	X_0	X_0		X_9	X_1	X_7
X_{10}	X_4	X_0		X_{11}	X_5	X_7
X_{12}	X_0	X_8		X_{13}	X_1	X_{15}
X_{14}	X_4	X_8		X_{15}	X_5	X_{15}

Equivalência entre AFD e AFND

$\mathcal{Q}$ Determinar um AFD equivalente ao AFND seguinte ?

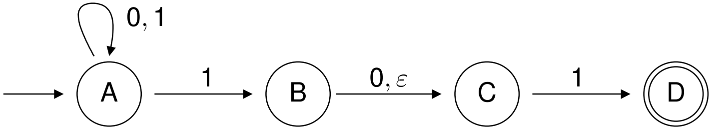


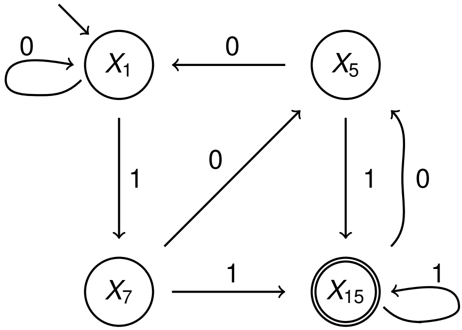
- $M' =$



Equivalência entre AFD e AFND

$\mathcal{Q}$ Determinar um AFD equivalente ao AFND seguinte ?





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

Operações sobre AFD e AFND

- Os automátos finitos são fechados sobre as operações de:
  - Reunião
  - Concatenação
  - Fecho
  - Interceção
  - Complementação

Operações sobre AFD e AFND: reunião

Exemplo

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

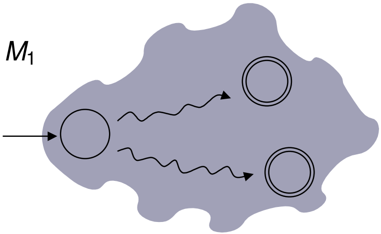
$$L_1 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

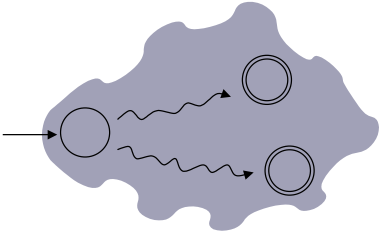
Determine um AFD ou AFND que reconheça $L = L_1 \cup L_2$.

Operações sobre AFD e AFND: reunião

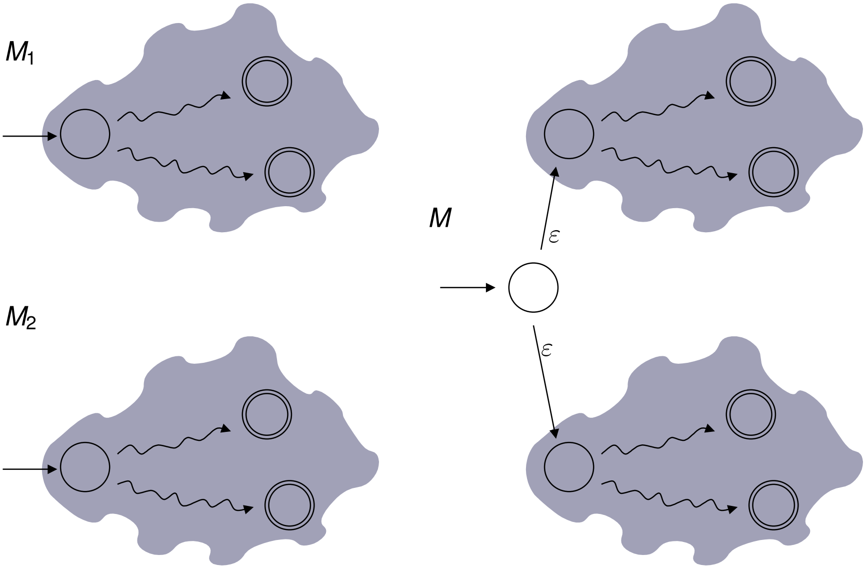
M_1



M_2



Operações sobre AFD e AFND: reunião



Operações sobre AFD e AFND: reunião

$\mathcal{D}$ 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

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

$$F = F_1 \cup F_2$$

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

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

Operações sobre AFD e AFND: reunião

Exemplo

$\mathcal{Q}$ Sobre o alfabeto $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, sejam $L_1 = \{\mathtt{a}\omega \mid \omega \in A^*\}$ e $L_2 = \{\omega\mathtt{a} \mid \omega \in A^*\}$. Determine um AFD ou AFND que reconheça $L = L_1 \cup L_2$.

$\mathcal{R}$ $M_1 = (A, Q_1, q_1, \delta_1, F_1)$ com

$$Q_1 = \{S_1, S_2\}, \quad q_1 = S_1, \quad F_1 = \{S_2\}$$
$$\delta_1 = \{(S_1, a, S_2), (S_2, a, S_2), (S_2, b, S_2), (S_2, c, S_2)$$

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

$$Q_2 = \{S_3, S_4\}, \quad q_2 = S_3, \quad F_2 = \{S_4\}$$
$$\delta_1 = \{(S_3, a, S_3), (S_3, b, S_3), (S_3, c, S_3), (S_3, a, S_4)$$

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

$$Q = \{S_0, S_1, S_2, S_3, S_4\}, \quad q_0 = S_0, \quad F = \{S_2, S_4\},$$
$$\delta = \{(S_0, \varepsilon, S_1), (S_0, \varepsilon, S_3), (S_1, a, S_2), (S_2, a, S_2), (S_2, b, S_2), (S_2, c, S_2), (S_3, a, S_3), (S_3, b, S_3), (S_3, c, S_3), (S_3, a, S_4)\}$$

Operações sobre AFD e AFND: concatenação

Exemplo

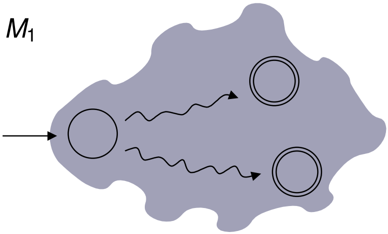
$\mathcal{Q}$ Sobre o alfabeto $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, sejam L_1 e L_2 as duas linguagens seguintes:

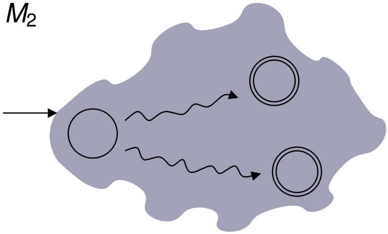
$$L_1 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

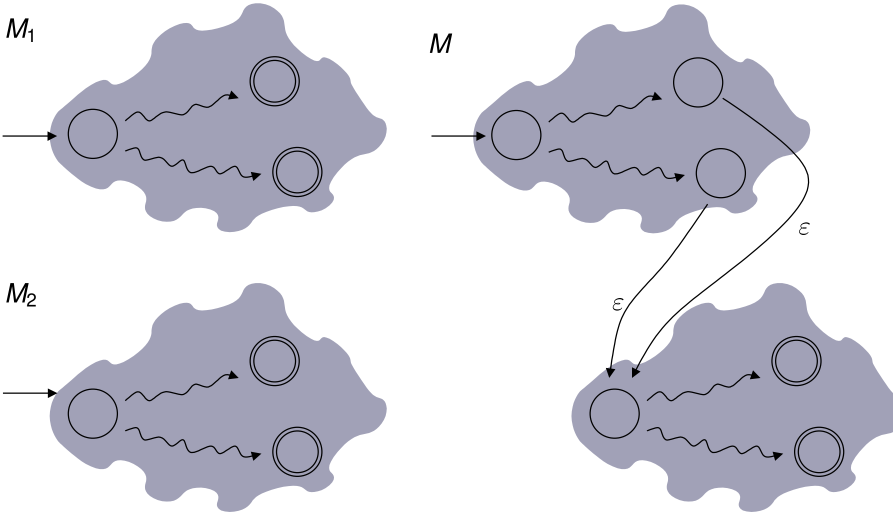
Determine um AFND que reconheça $L = L_1 \cdot L_2$.

Operações sobre AFD e AFND: concatenação





Operações sobre AFD e AFND: concatenação



Operações sobre AFD e AFND: concatenação

$\mathcal{D}$ 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, $L(M) = L(M_1) \cdot L(M_2)$.

Operações sobre AFD e AFND: concatenação

Exemplo

- Determine o AFND que reconhece a linguagem constituída pelos literais da linguagem de programação C que representam constantes numéricas reais.

Operações sobre AFD e AFND: fecho

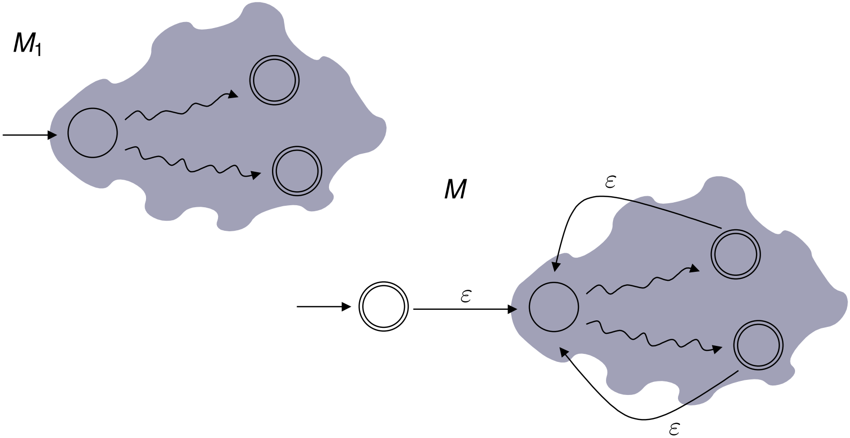
Exemplo

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

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

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

Operações sobre AFD e AFND: fecho



Note que em geral não se pode fundir o novo estado inicial com o antigo.

Operações sobre AFD e AFND: fecho

$\mathcal{D}$ 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

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

$$F = F_1 \cup \{q_0\}$$

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

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

Operações sobre AFD e AFND: intersecção

Exemplo

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

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

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

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

Operações sobre AFD e AFND: intersecção

$\mathcal{D}$ 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

$$Q = Q_1 \times Q_2$$
$$q_0 = (q_1, q_2)$$
$$F = F_1 \times F_2$$
$$\delta \subseteq (Q_1 \times Q_2) \times A_\varepsilon \times (Q_1 \times Q_2)$$

sendo δ 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., $L(M) = L(M_1) \cap L(M_2)$.

Operações sobre AFD e AFND: intersecção

Exercício

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

$$L_1 = \{\upsilon\omega \mid \upsilon \in \{\mathtt{a}, \mathtt{b}\} \wedge \omega \in A^*\}$$

$$L_2 = \{\omega \in A^* \mid \#(\mathtt{a}, \omega)\%2 = 0\}$$

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

Operações sobre AFD e AFND: complementação

Exemplo

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

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

Determine o AFND ou AFD que reconhece a linguagem $\overline{L_1}$.

$\mathcal{R}$ Determina-se o AFD de L_1 e complementa-se o conjunto de aceitação.

Conversão de uma ER num AFND

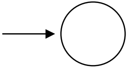
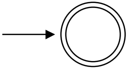
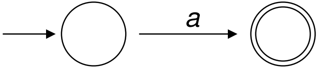
Dada uma expressão regular qualquer ela é:

- ou um elemento primitivo;
- ou uma expressão do tipo e^*, sendo e uma expressão regular qualquer;
- ou uma expressão do tipo $e_1.e_2$, sendo e_1 e e_2 duas expressões regulares quaisquer;
- ou uma expressão do tipo $e_1|e_2$, sendo e_1 e e_2 duas expressões regulares quaisquer;

Se se identificar os autómatos equivalentes das expressões primitivas, tem-se o problema da conversão de uma expressão regular para um autómato finito resolvido, visto que se sabe como fazer a reunião, a concatenação e o fecho de autómatos.

Conversão de uma ER num AFND

Autómatos dos elementos primitivos

expressão regular	autómato finito
()	
ε	
a	

- Na realidade, o autómato referente a ε pode ser obtido aplicando o fecho ao autómato de ().

Conversão de uma ER num AFND

Algoritmo de conversão

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

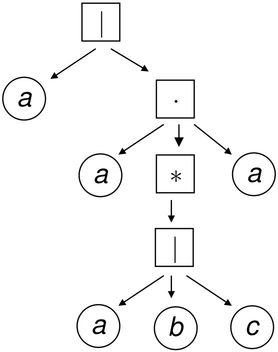
- Na realidade, o algoritmo corresponde a um processo de decomposição arbórea a partir da raiz seguido de um processo de construção arbórea a partir das folhas.

Exemplo de conversão de uma ER num AFND

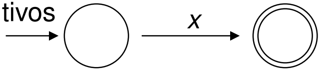
$\mathcal{Q}$ Construa um autómato equivalente à expressão regular $e = a|a(a|b|c)^*a$.

$\mathcal{R}$

1. Decomposição:



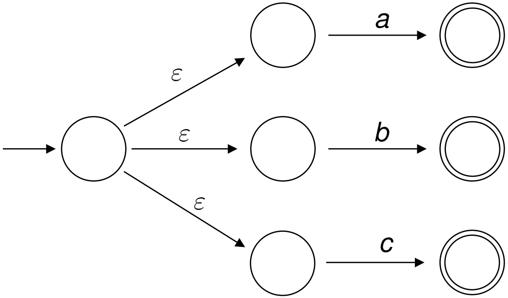
2. Autómatos primitivos



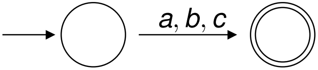
com $x = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

Exemplo de conversão de uma ER num AFND

3. Reunião para obter $(a|b|c)$

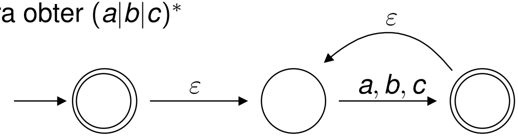


4. Simplificando

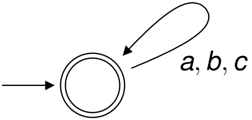


Exemplo de conversão de uma ER num AFND

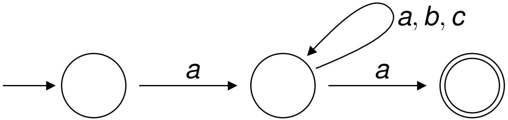
5. Fecho para obter $(a|b|c)^*$



6. Simplificando

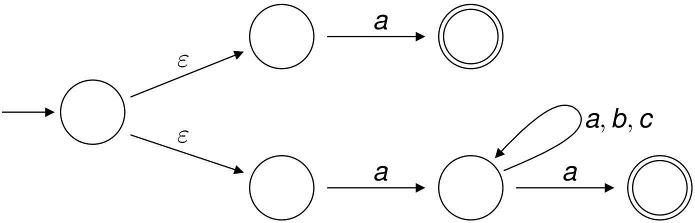


7. Concatenação (já com simplificação) para obter $a(a|b|c)^*a$

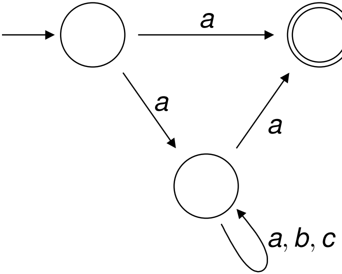


Exemplo de conversão de uma ER num AFND

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



- Simplificando



AUTÓMATO FINITO GENERALIZADO (AFG)

$\mathcal{D}$ Um **autómato finito generalizado** (AFG) é um quíntuplo $M = (A, Q, q_0, \delta, F)$, em que:

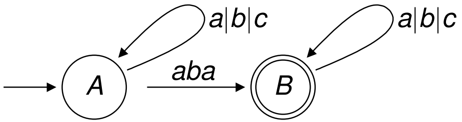
- A é o alfabeto de entrada;
- Q é um conjunto finito não vazio de estados;
- $q_0 \in Q$ é o estado inicial;
- $\delta \subseteq (Q \times E \times Q)$ é a relação de transição entre estados, sendo E o conjunto das expressões regulares definidas sobre A; e
- $F \subseteq Q$ é o conjunto dos estados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

- A diferença em relação ao AFND está na definição da relação δ. Neste caso as etiquetas são *expressões regulares*.

- Com base nesta definição os automátos finitos deterministas e não deterministas são autómatos finitos generalizados.

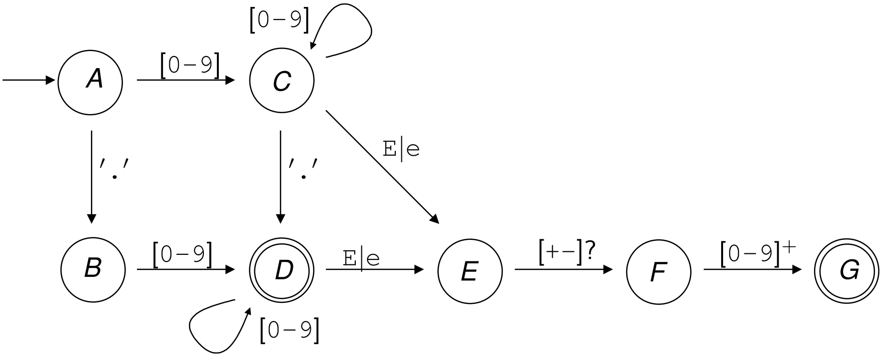
EXEMPLOS DE AFG

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



Exemplos de AFG

- O AFG seguinte representa as constantes reais em C.



Conversão de um AFG numa ER

AFG reduzido

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

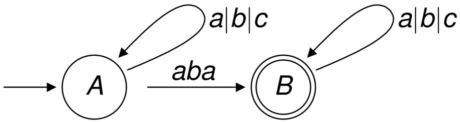
$$\longrightarrow (A) \xrightarrow{e} ((B))$$

designa-se por **autómato 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 a expressão e é uma expressão regular equivalente ao autómato.

CONVERSÃO DE UM AFG NUMA ER

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



$\mathcal{Q}$ Como converter este autómato à forma reduzida?

Conversão de um AFG numa ER

Algoritmo de conversão

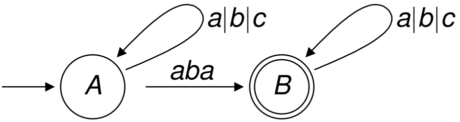
1. transformação de um AFG noutro cujo estado inicial **não tenha transições a chegar**.
    - Se necessário, acrescenta-se um novo estado inicial com uma transição em ε para o antigo.
2. transformação de um AFG noutro com **um único estado de aceitação**, **sem transições de saída**.
    - Se necessário,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

Conversão de um AFG numa ER

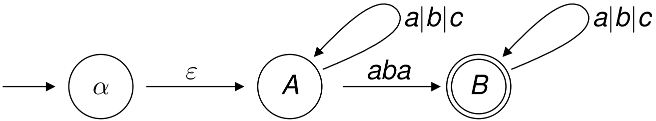
Algoritmo de conversão

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 ε para o antigo.

antes



depois

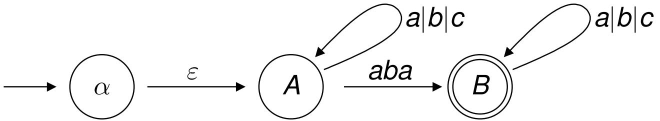


Conversão de um AFG numa ER

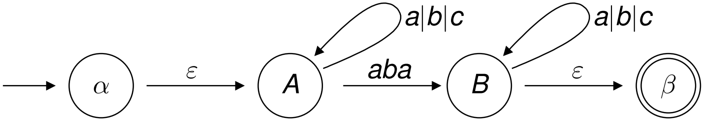
Algoritmo de conversão

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 ε dos anteriores estados de aceitação, que deixam de o ser.

antes



depois

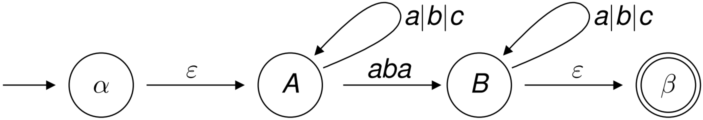


Conversão de um AFG numa ER

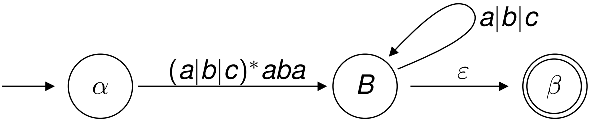
Algoritmo de conversão

3. Eliminação dos restantes estados.
    - Os estados são eliminados um a um, em processos de transformação que mantêm a equivalência: estado A.

antes



depois

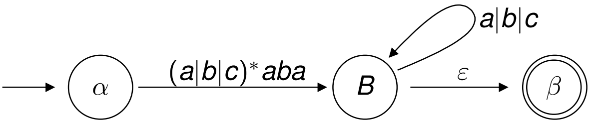


Conversão de um AFG numa ER

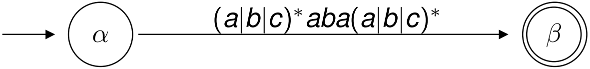
Algoritmo de conversão

3. Eliminação dos restantes estados.
   - Os estados são eliminados um a um, em processos de transformação que mantêm a equivalência: estado B.

antes



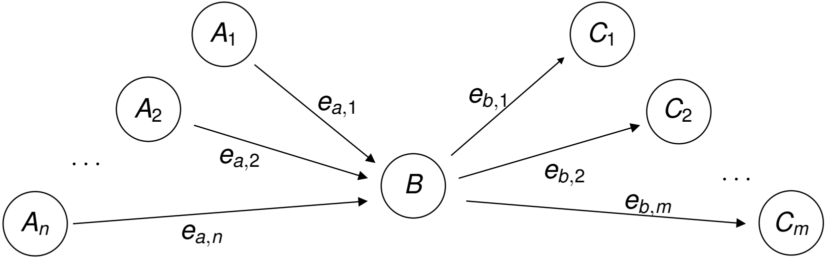
depois



Conversão de um AFG numa ER

Algoritmo de eliminação de um estado

- Considere que pretende eliminar o estado B

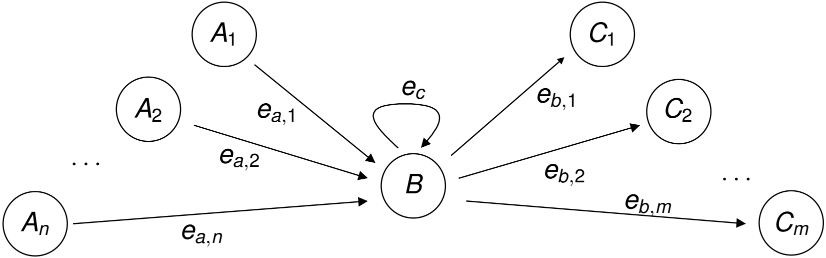


- Para ir de A_i para C_j através de B é preciso uma palavra que encaixe na expressão regular $(e_{a,i})(e_{b,j})$.
- Então, se se retira B, é preciso acrescentar uma transição de A_i para C_j com etiqueta $(e_{a,i})(e_{b,j})$.
- Esta transição pode ficar em paralelo com um que já exista.

Conversão de um AFG numa ER

Algoritmo de eliminação de um estado (2)

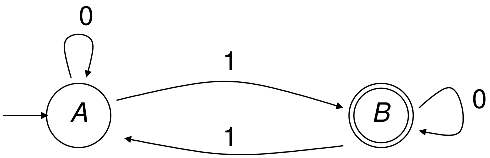
- Considere que pretende eliminar o estado B



- Para ir de A_i para C_j através de B é preciso uma palavra que encaixe na expressão regular $(e_{a,i})(e_c)^*(e_{b,j})$.
- Então, se se retira B, é preciso acrescentar uma transição de A_i para C_j com etiqueta $(e_{a,i})(e_c)^*(b,j)$.
- Esta transição pode ficar em paralelo com um que já exista.

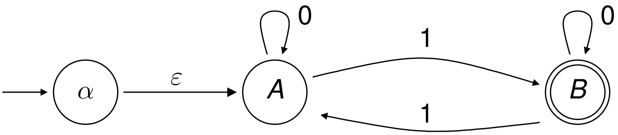
EXEMPLO DE CONVERSÃO DE UM AFG NUMA ER

$\mathcal{Q}$ Obtenha uma ER equivalente ao AF seguinte



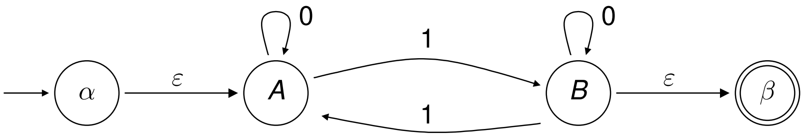
$\mathcal{R}$ Aplique-se passo a passo o algoritmo anterior.

1. Porque o estado inicial possui uma transição a chegar, deve-se acrescentar um novo estado inicial:

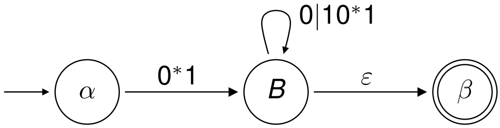


EXEMPLO DE CONVERSÃO DE UM AFG NUMA ER

2. Porque o estado aceitação possui uma transição a sair, deve-se acrescentar um novo estado de aceitação



3. Eliminando o estado *A* obtém-se



4. Eliminando o estado *B* obtém-se $0^*\,1\,(0\mid 10^*1)^*$.