Linguagens Formais e Autómatos / Compiladores

Autómatos Finitos (AF)

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

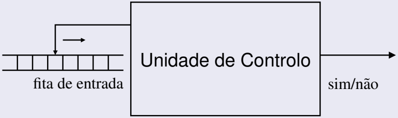
DETI, Universidade de Aveiro

Ano letivo de 2020-2021

Sumário

Autómato finito

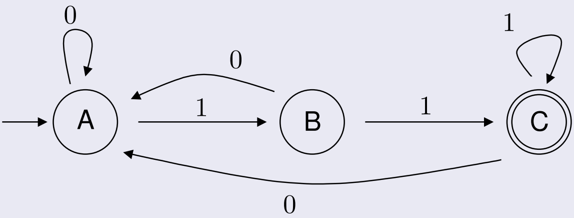
Um autómato finito é um mecanismo reconhecedor das palavras de uma linguagem regular



- 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 (reconhecida)
- Os autómatos finitos podem ser **deterministas**, **não deterministas** ou **generalizados**

Autómato finito determinista

Um **autómato finito determinista** é um autómato finito

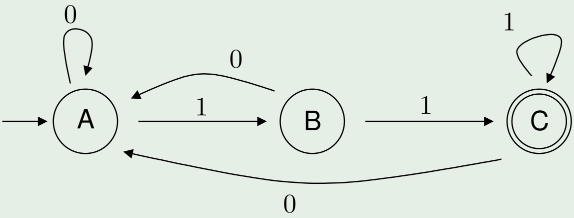


onde

- 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;
- os caminhos que começam no estado inicial e terminam num estado de aceitação representam as palavras aceites (reconhecidas) pelo autómato.

Autómato finito determinista: exemplo (1)

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



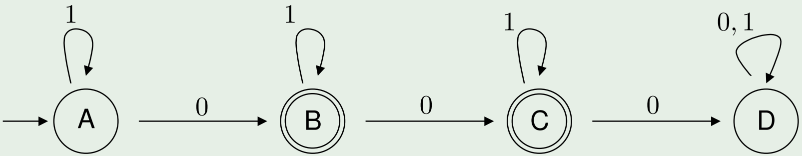
$\mathcal{R}$ Todas as palavras terminadas em 11.

$\mathcal{E}$ Obtenha uma expressão regular que represente a mesma linguagem.

ER = 0 (01 | 10)* 11

Autómato finito determinista: exemplo (2)

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



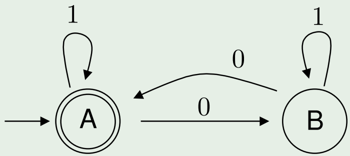
$\mathcal{R}$ Todas as palavras com apenas 1 ou 2 zeros.

$\mathcal{E}$ Obtenha uma expressão regular que represente a mesma linguagem.

ER = (0 | 1)*00*

Autómato finito determinista: exemplo (3)

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



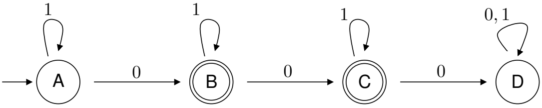
$\mathcal{R}$ as sequências binárias com um número par de zeros.

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.

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



Definição de autómato finito determinista

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

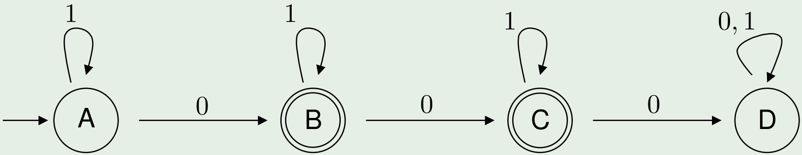
- A é o alfabeto de entrada;
- Q é um conjunto finito não vazio de estados;
- $q_0 \in Q$ é o estado inicial;
- $\delta : Q \times A \to Q$ é uma função que determina a transição entre estados; e
- $F \subseteq Q$ é o conjunto dos estados de aceitação.

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

- 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 $(q, a, q) \in Q \times A \times Q$

Autómato finito determinista: exemplo (4)

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



$\mathcal{R}$

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

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

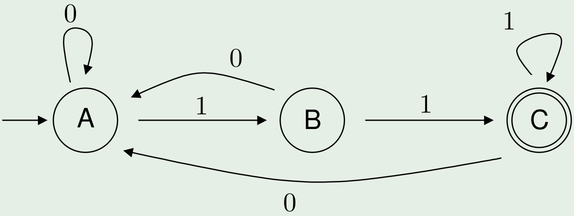
- $\delta = \{$
  $(A, 0, B), (A, 1, A),$
  $(B, 0, C), (B, 1, B),$
  $(C, 0, D), (C, 1, C),$
  $(D, 0, D), (D, 1, D)\}$

- $\delta =$

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

Autómato 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

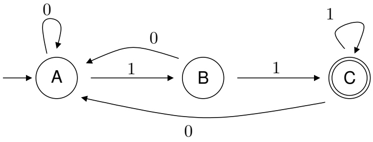
Linguagem reconhecida por um AFD (1)

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

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

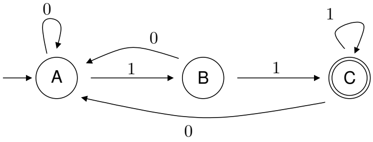
- A palavra $\omega_1 = \texttt{0101}$ faz o caminho $A \xrightarrow{0} A \xrightarrow{1} B \xrightarrow{0} A \xrightarrow{1} B$
  - como B não é de aceitação, ω_1 não pertence à linguagem
- A palavra $\omega_2 = \texttt{0011}$ faz o caminho $A \xrightarrow{0} A \xrightarrow{0} A \xrightarrow{1} B \xrightarrow{1} C$
  - como C é de aceitação, ω_2 pertence à linguagem



Linguagem reconhecida por um AFD (2)

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

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



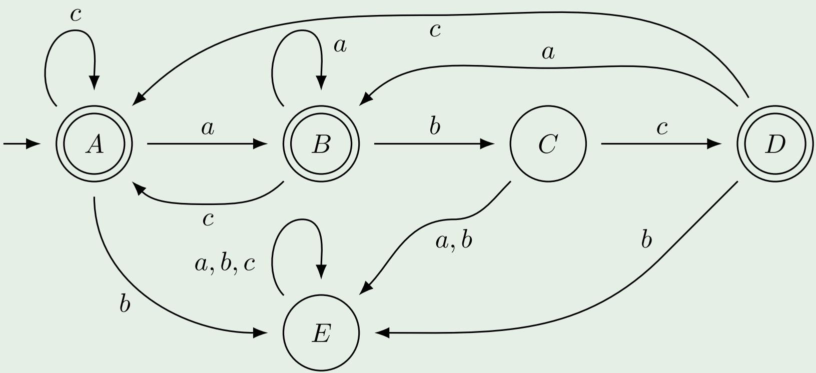
Autómato finito determinista: exemplo (6)

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

$$L = \{\omega \in A^* \ : \ (\omega_i = \mathtt{b}) \ \Rightarrow \ ((\omega_{i-1} = \mathtt{a}) \ \wedge \ (\omega_{i+1} = \mathtt{c}))\}$$

Projecte um autómato que reconheça L.

$\mathcal{R}$



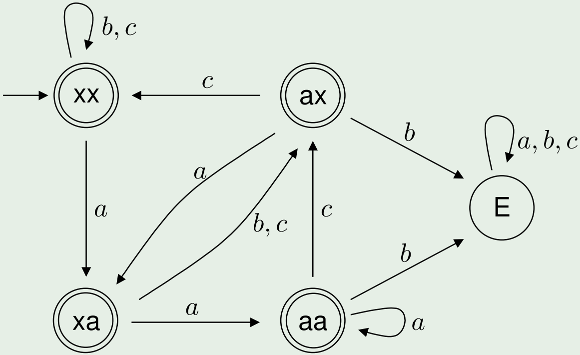
Autómato finito determinista: exemplo (7)

$\mathcal{Q}$ 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.

$\mathcal{R}$



Autómato finito determinista: exemplo (8)

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

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

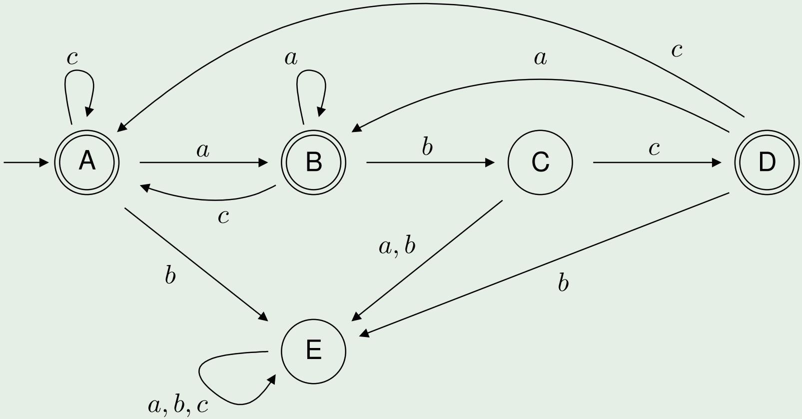
Projecte um autómato que reconheça L.

$\mathcal{R}$

???

Redução de autómato finito determinista (1)

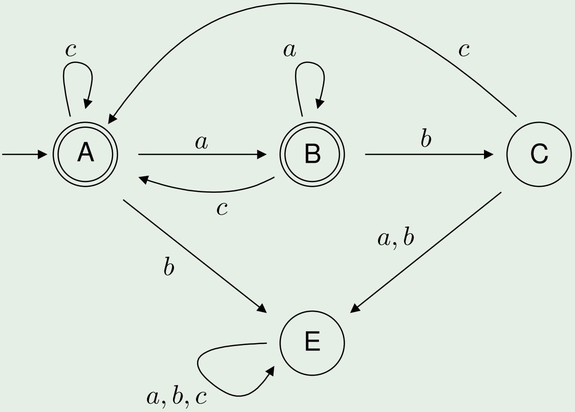
$\mathcal{Q}$ Considere o autómato seguinte (o do exemplo 6) e compare os estados A e D. Que pode concluir ?



- São equivalentes. Por conseguinte, podem ser fundidos

Redução de autómato finito determinista (2)

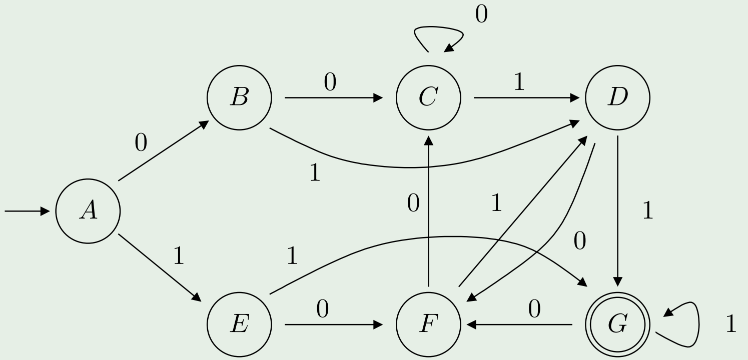
- O que resulta em



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

Algoritmo de Redução de AFD (1)

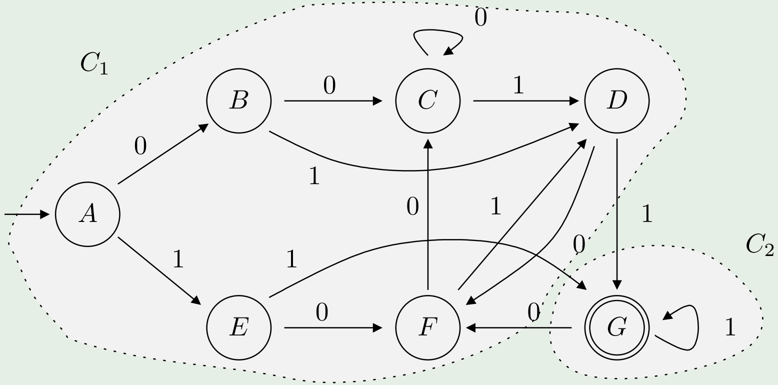
- Como proceder para reduzir um AFD?



- 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)

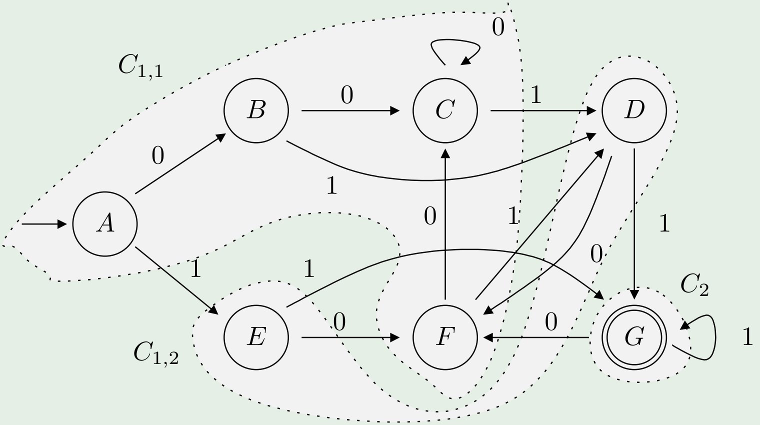
- Obtêm-se $C_1 = \{A, B, C, D, E, F\}$ e $C_2 = \{G\}$.



- 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)

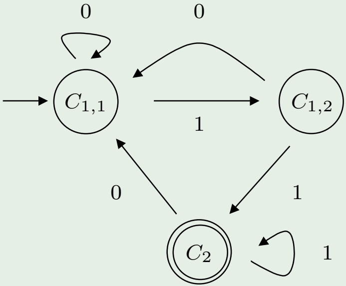
- Dividindo C_1 em $C_{1,1} = \{A, B, C, F\}$ e $C_{1,2} = \{D, E\}$ obtem-se



- 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 autómato.

Algoritmo de Redução de AFD (4)

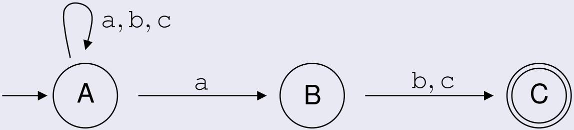
- Autómato reduzido



- Nos apontamentos encontra uma versão não gráfica do algoritmo.

Autómato finito não determinista

Um autómato finito não determinista é um autómato finito



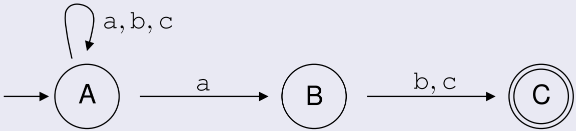
onde

- as transições estão associadas a símbolos individuais do alfabeto **ou à palavra vazia (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;
- os caminhos que começam no estado inicial e terminam num estado de aceitação representam as palavras aceites (reconhecidas) pelo autómato.

- As transições múltiplas ou com ε 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` ?



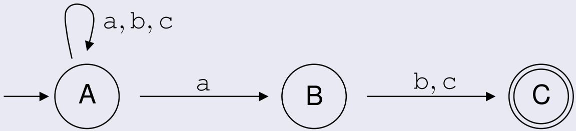
- Há 3 caminhos alternativos

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

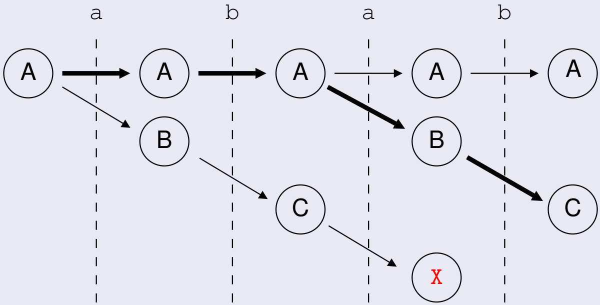
- Como há um caminho que conduz a um estado de aceitação a palavra é reconhecida pelo autómato

AFND: caminhos alternativos

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

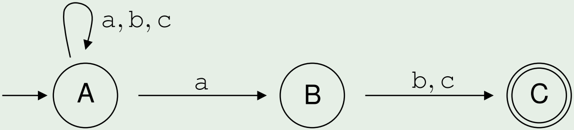


- Que se podem representar de forma arbórea



AFND: exemplo

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



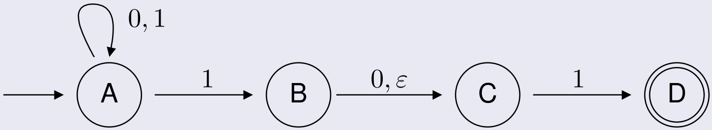
$\mathcal{R}$ Todas as palavras que terminarem em ab ou ac

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

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

AFND com transições-ε

- Considere o AFND seguinte que contém uma transição-ε.



- A palavra `101` é reconhecida pelo autómato através do caminho

  $$A \xrightarrow{1} B \xrightarrow{0} C \xrightarrow{1} D$$

- A palavra `11` é reconhecida pelo autómato através do caminho

  $$A \xrightarrow{1} B \xrightarrow{\varepsilon} C \xrightarrow{1} D$$

  porque $11 = 1\varepsilon1$

- Este autómato reconhece todas as palavras terminadas em `11` ou `101`

  $$L = \{\omega_1\omega_2 \,:\, \omega_1 \in A^* \,\wedge\, \omega_2 \in \{\texttt{11}, \texttt{101}\}\}.$$

AFND: definição

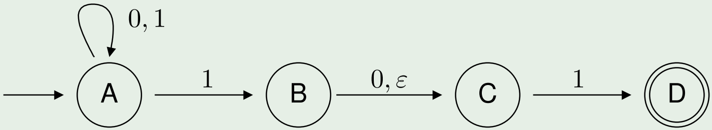
$\mathcal{D}$ Um **autómato finito não determinista** (AFND) é um quíntuplo $M = (A, Q, q_0, \delta, F)$, em que:

- 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 (2)

$\mathcal{Q}$ Represente textualmente o AFND



$\mathcal{R}$

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

- O par $(A, 1, A), (A, 1, B)$ faz com que δ 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 $s_0, s_1, \cdots, s_n$, 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 indicar que a palavra u permite ir do estado q_i ao estado q_j.
- Usando esta notação tem-se $L(M) = \{u \,:\, q_0 \xrightarrow{u} q_f \,\wedge\, q_f \in F\}$.

Equivalência entre AFD e AFND

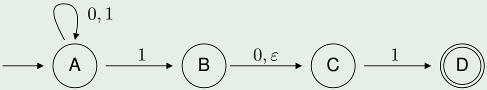
- A classe das linguagens cobertas por um AFD é a mesma que a classe das linguagens cobertas por um AFND
- Isto significa que:
  - Se M é um AFD, então $\exists_{M' \in \text{AFND}} : L(M') = L(M)$.
  - Se M é um AFND, então $\exists_{M' \in \text{AFD}} : L(M') = L(M)$.

- 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 $\quad \delta : Q \times A \to Q$.
  - Nos AFND $\quad \delta \subset Q \times A_\varepsilon \times Q$
  - Mas, se $\delta : Q \times A \to Q$ então $\delta \subseteq Q \times A \times Q \subset Q \times A_\varepsilon \times Q$
  - Logo, um AFD é um AFND

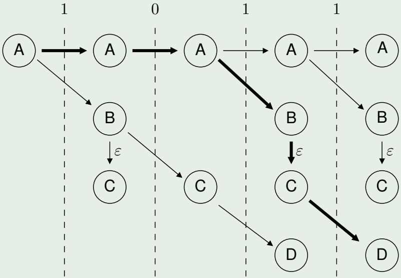
Equivalente AFD de um AFND (1)

- 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)

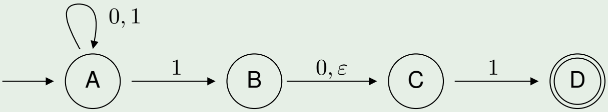
- Dado um AFND $M = (A, Q, q_0, \delta, F)$, considere o AFD $M' = (A, Q', q'_0, \delta', F')$ onde:
  - $Q' = \wp(Q)$
  - $q'_0 = \varepsilon\text{-closure}(q_0)$
  - $F' = \{f' \in \wp(Q) \ : \ f' \cap F \neq \emptyset\}$
  - $\delta' = \wp(Q) \times A \to \wp(Q)$,

    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.

- ε-closure(q) é o conjunto de estados constituído por q mais todos os direta ou indiretamente alcançáveis a partir de q apenas por transições-ε
- 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

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



$\mathcal{R}$

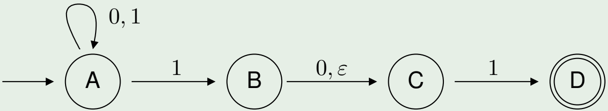
- $Q' = \{X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, x_{10}, X_{11}, X_{12}, X_{13}, X_{14}, X_{15}\}$, com

$$\begin{array}{llll} X_0 = \{\} & X_1 = \{A\} & X_2 = \{B\} & X_3 = \{A, B\} \\ X_4 = \{C\} & X_5 = \{A, C\} & X_6 = \{B, C\} & X_7 = \{A, B, C\} \\ X_8 = \{D\} & X_9 = \{A, D\} & X_{10} = \{B, D\} & X_{11} = \{A, B, D\} \\ X_{12} = \{C, D\} & X_{13} = \{A, C, D\} & X_{14} = \{B, C, D\} & X_{15} = \{A, B, C, D\} \end{array}$$

- $q'_0 = \varepsilon$-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

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



$\mathcal{R}$

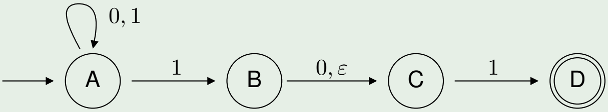
- $\delta' =$

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

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

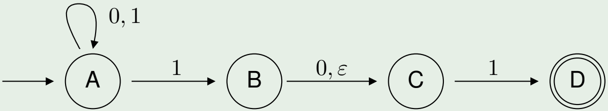


$\mathcal{R}$

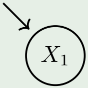
- Consegue-se o mesmo resultado através de um processo construtivo.

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$

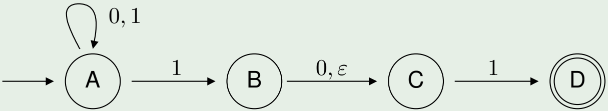


$X_1 = \{A\}$

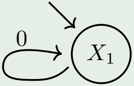
- Comece-se com o estado inicial ($X_1 = \{A\}$)

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$

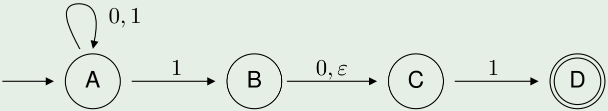


$X_1 = \{A\}$

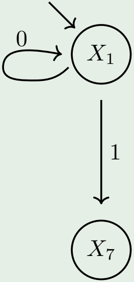
- $\delta'(X_1, 0) = \varepsilon$-closure$(A) = \{A\}$

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$

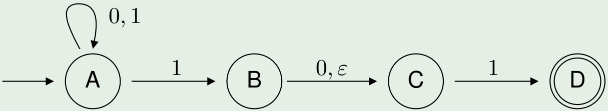


$X_1 = \{A\}$
$X_7 = \{A, B, C\}$

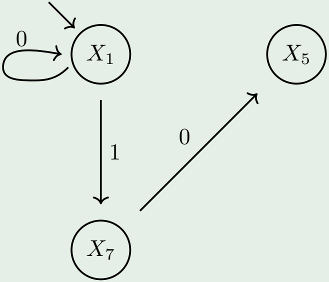
- $\delta'(X_1, 1) = \varepsilon\text{-closure}(A) \cup \varepsilon\text{-closure}(B) = \{A\} \cup \{B, C\} = \{A, B, C\}$

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$

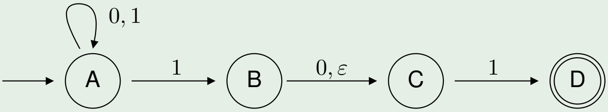


$X_1 = \{A\}$
$X_7 = \{A, B, C\}$
$X_5 = \{A, C\}$

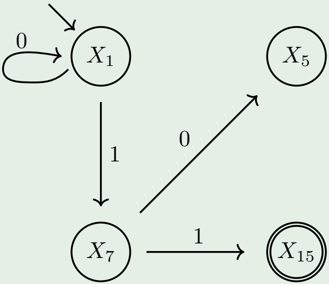
- $\delta'(X_7, 0) = \varepsilon$-closure$(A) \cup \varepsilon$-closure$(C) = \{A\} \cup \{C\} = \{A, C\}$

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$

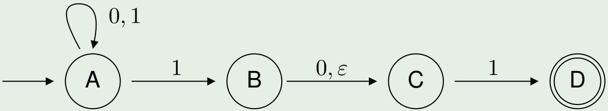


$X_1 = \{A\}$
$X_7 = \{A, B, C\}$
$X_5 = \{A, C\}$
$X_{15} = \{A, B, C, D\}$

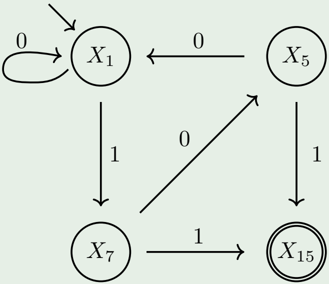
- $\delta'(X_7, 1) = \varepsilon$-closure$(A) \cup \varepsilon$-closure$(B) \cup \varepsilon$-closure$(D) = \{A\} \cup \{B, C\} \cup \{D\} = \{A, B, C, D\}$
- É de aceitação porque $\{A, B, C, D\} \cap \{D\} \neq \emptyset$

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$

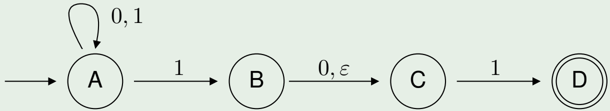


$X_1 = \{A\}$
$X_7 = \{A, B, C\}$
$X_5 = \{A, C\}$
$X_{15} = \{A, B, C, D\}$

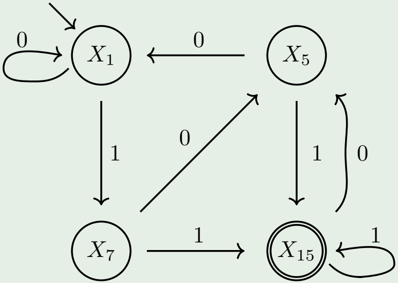
- $\delta'(X_5, 0) = \varepsilon$-closure$(A) = \{A\}$
- $\delta'(X_5, 1) = \varepsilon$-closure$(A) \cup \varepsilon$-closure$(B) \cup \varepsilon$-closure$(D) = \{A\} \cup \{B, C\} \cup \{D\} = \{A, B, C, D\}$

Equivalente AFD de um AFND: exemplo

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



$\mathcal{R}$



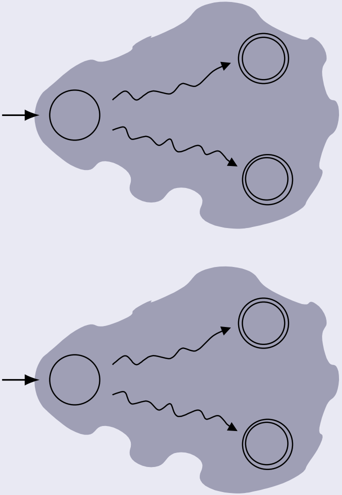
$X_1 = \{A\}$
$X_7 = \{A, B, C\}$
$X_5 = \{A, C\}$
$X_{15} = \{A, B, C, D\}$

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

Operações sobre AFD e AFND

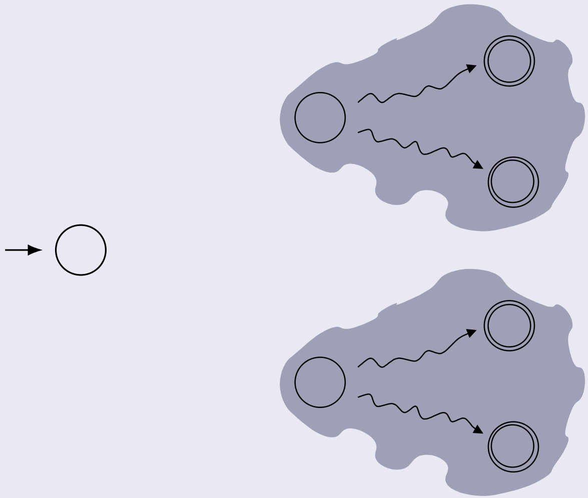
- Os automátos finitos (AF) são fechados sobre as operações de:
  - Reunião
  - Concatenação
  - Fecho
  - Intersecção
  - Complementação

Reunião de AF



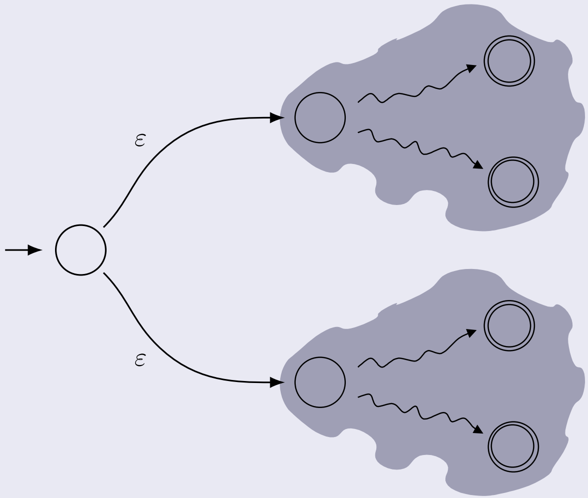
- Como criar um AF que represente a reunião destes dois AF?

Reunião de AF



- acrescenta-se um novo estado que passa a ser o inicial

Reunião de AF



- e acrescentam-se transições-ε deste novo estado para os estados iniciais originais

Reunião 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

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

$$F = F_1 \cup F_2$$

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

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)

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

$$L_1 = \{\omega\texttt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\texttt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$

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

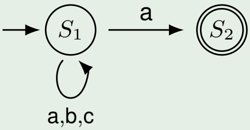
Reunião de AF: exemplo (1)

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- Constroi-se um AF para a linguagem L_1

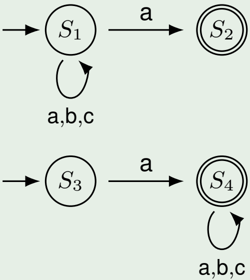
Reunião de AF: exemplo (1)

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- Constroi-se um AF para a linguagem L_2

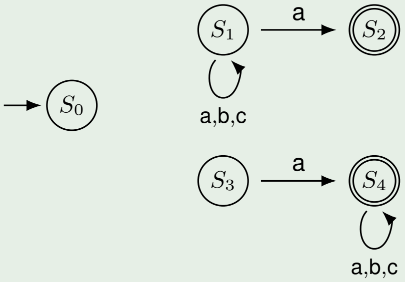
Reunião de AF: exemplo (1)

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- Acrescenta-se um novo estado (S_0), que passa a ser o inicial

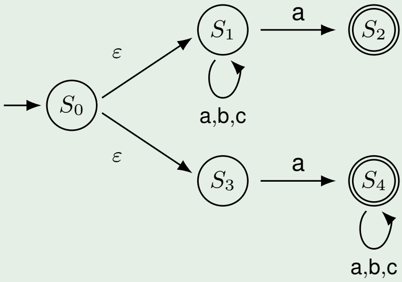
Reunião de AF: exemplo (1)

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

$$L_1 = \{\omega\texttt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\texttt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- E acrescentam-se transições-ε de S_0 (novo estado inicial) para S_1 e S_2 (os estados iniciais originais)

Reunião de AF: exemplo (1)

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$

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

$Q_1 = \{S_1, S_2\}, \quad q_1 = S_1, \quad F_1 = \{S_2\}$

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

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

$Q_2 = \{S_3, S_4\}, \quad q_2 = S_3, \quad F_2 = \{S_4\}$

$\delta_2 = \{(S_3, a, S_4), (S_4, a, S_4), (S_4, b, S_4), (S_4, c, S_4)$

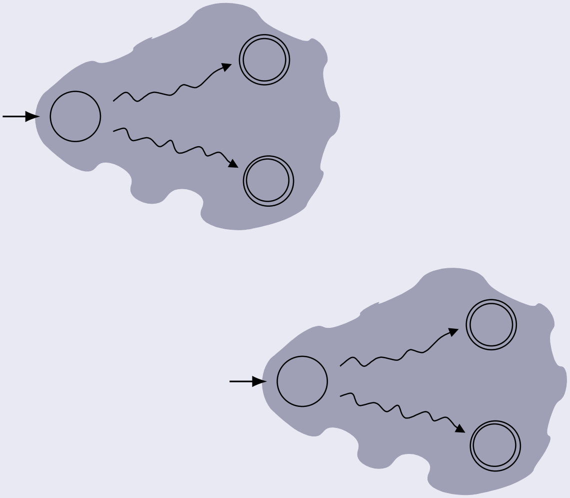
$M = M_1 \cup M_2 = (A, Q, q_0, \delta, F)$ com

$Q = \{S_0, S_1, S_2, S_3, S_4\}, \quad q_0 = S_0, \quad F = \{S_2, S_4\},$

$\delta = \{(S_0, \varepsilon, S_1), (S_0, \varepsilon, S_3), (S_1, a, S_1), (S_1, b, S_1), (S_1, c, S_1),$
$(S_1, a, S_2), (S_3, a, S_4), (S_4, a, S_4), (S_4, b, S_4), (S_4, c, S_4 + 9)\}$

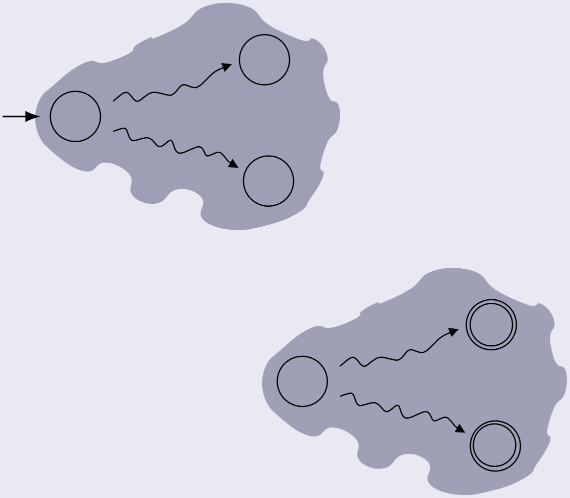
- Alternativamente, pode ser escrito de forma textual

Concatenação de AF



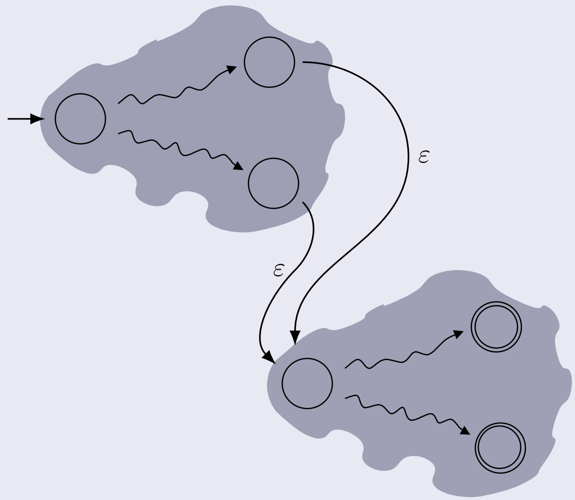
- Como criar um AF que represente a concatenação destes dois AF?

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

Concatenação de AF



- e acrescentam-se transições-ε 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 $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, sejam L_1 e L_2 as duas linguagens seguintes:

$$L_1 = \{\omega \mathtt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\mathtt{a} \omega \mid \omega \in A^*\}$$

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?

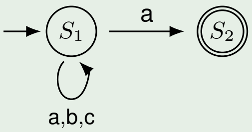
Concatenação de AF: exemplo

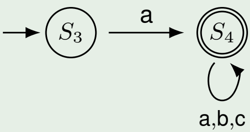
$\mathcal{Q}$ Sobre o alfabeto $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, sejam L_1 e L_2 as duas linguagens seguintes:

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$





- Constroi-se AF para as linguagens L_1 e L_2

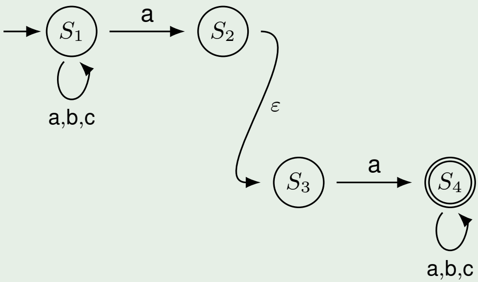
Concatenação de AF: exemplo

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad\qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

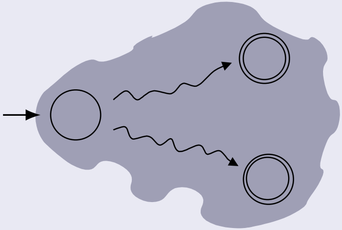
Determine um AF que reconheça $L = L_1 \cdot L_2$.

$\mathcal{R}$



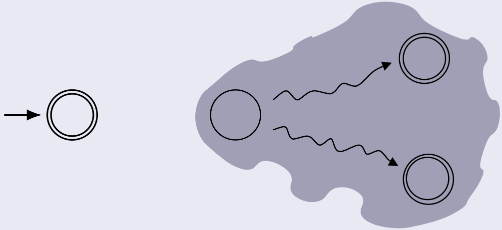
- S_2 deixa de ser de aceitação; S_3 deixa de ser de entrada
- acrescenta-se uma transição-ε de S_2 para S_3

Fecho de AF



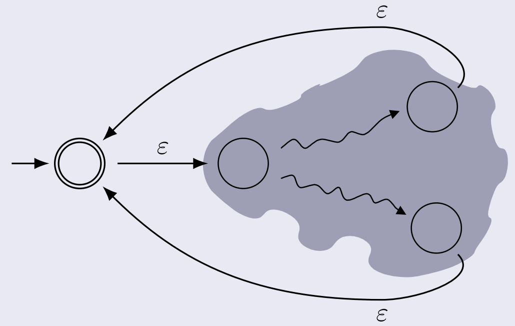
- Como criar um AF que represente a concatenação destes dois AF?

Fecho de AF



- acrescenta-se um novo estado que passa a ser o inicial
- o novo estado inicial é de aceitação

Fecho de AF



- e acrescentam-se transições-ε dos estados de aceitação do AF para o (novo) estado inicial
- os 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

$\mathcal{D}$ 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

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

$$F = \{q_0\}$$

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

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-ε se dirigem a q_1

Fecho de AF: 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 que reconhece a linguagem ${L_1}^*$.

$\mathcal{R}$

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

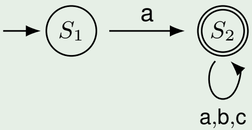
Fecho de AF: exemplo

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

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

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

$\mathcal{R}$



- Constroi-se um AF para L_1

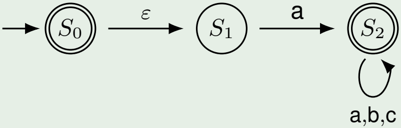
Fecho de AF: exemplo

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

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

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

$\mathcal{R}$



- 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-ε

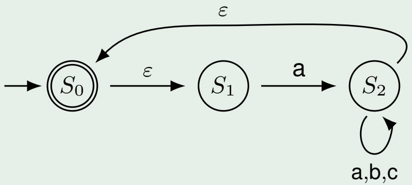
Fecho de AF: exemplo

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

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

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

$\mathcal{R}$



- liga-se o estado S_2 (aceitação anterior) ao S_0 (novo inicial)
- S_2 deixa (pode deixar) de ser de aceitação

Intersecção de AF: exemplo

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$

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

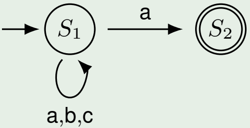
Intersecção de AF: exemplo

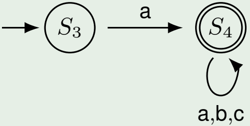
$\mathcal{Q}$ Sobre o alfabeto $A = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, sejam L_1 e L_2 as duas linguagens seguintes:

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$





- Constroi-se AF para as linguagens L_1 e L_2

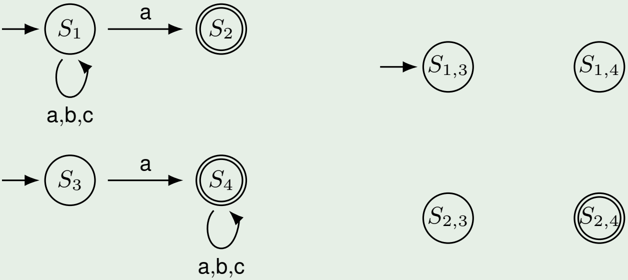
Intersecção de AF: exemplo

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

$$L_1 = \{\omega\mathtt{a} \,|\, \omega \in A^*\} \qquad\qquad L_2 = \{\mathtt{a}\omega \,|\, \omega \in A^*\}$$

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

$\mathcal{R}$



- Definem-se os estados que resultam do produto cartesiano $\{S_1, S_2\} \times \{S_3, S_4\}$
- Mas, alguns podem não ser alcançáveis

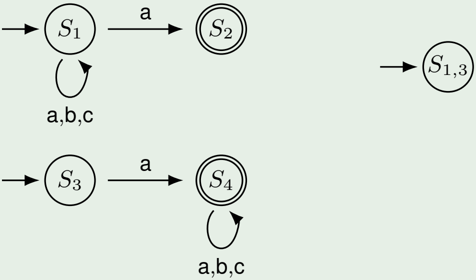
Intersecção de AF: exemplo

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- Pelo que comecemos apenas pelo $S_{1,3}$

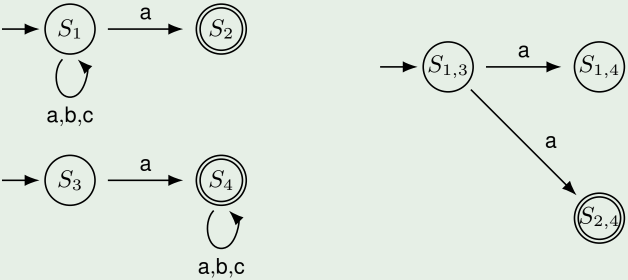
Intersecção de AF: exemplo

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- de $S_1 \xrightarrow{\mathtt{a}} S_2$ e $S_3 \xrightarrow{\mathtt{a}} S_4$ aparece $S_{1,3} \xrightarrow{\mathtt{a}} S_{2,4}$
- de $S_1 \xrightarrow{\mathtt{a}} S_1$ e $S_3 \xrightarrow{\mathtt{a}} S_4$ aparece $S_{1,3} \xrightarrow{\mathtt{a}} S_{1,4}$

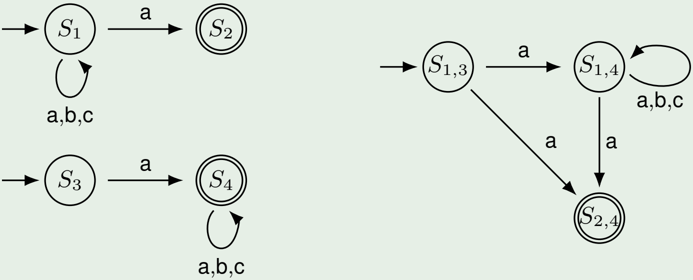
Intersecção de AF: exemplo

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

$$L_1 = \{\omega\mathtt{a} \mid \omega \in A^*\} \qquad L_2 = \{\mathtt{a}\omega \mid \omega \in A^*\}$$

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

$\mathcal{R}$



- de $S_1 \xrightarrow{\mathrm{x}} S_1$ e $S_4 \xrightarrow{\mathrm{x}} S_4$ aparece $S_{1,4} \xrightarrow{\mathrm{x}} S_{1,4}$, para $x \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$
- de $S_1 \xrightarrow{\mathrm{a}} S_2$ e $S_4 \xrightarrow{\mathrm{a}} S_4$ aparece $S_{1,4} \xrightarrow{\mathrm{a}} S_{2,4}$,

Intersecção 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

$$Q = Q_1 \times Q_2$$

$$q_0 = (q_1, q_2)$$

$$F = F_1 \times F_2$$

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

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

Complementação de AF

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

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

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

$\mathcal{R}$

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

Complementação de AF: exemplo

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

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

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

$\mathcal{R}$

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

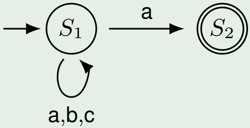
Complementação de AF: exemplo

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

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

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

$\mathcal{R}$



- Considere-se um AFND para a linguagem L_1

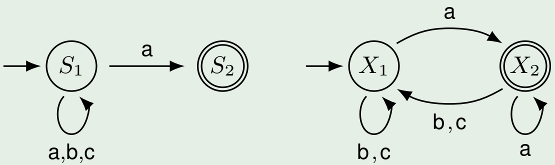
Complementação de AF: exemplo

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

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

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

$\mathcal{R}$



- Obtenha-se um determinista equivalente

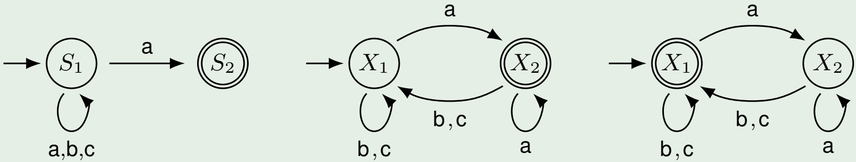
Complementação de AF: exemplo

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

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

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

$\mathcal{R}$



- Complemente-se os estados de aceitação

Operações sobre AF: exercício

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

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

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

Determine AF que reconheça a linguagem

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

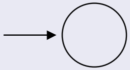
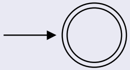
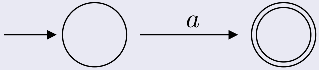
Equivalência entre ER e AF

- 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
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 AF
Algoritmo de conversão

- Se a expressão regular é do tipo primitivo, o autómato correspondente pode ser obtido da tabela anterior
- 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
- 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
- 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

Conversão de uma ER num AF
Exemplo

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

$\mathcal{R}$

1. Decomposição

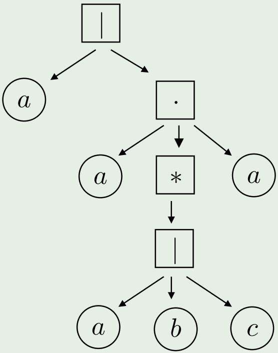
$$\boxed{\boxed{a} \mid \boxed{\boxed{a}\,\boxed{\boxed{(\,\boxed{a} \mid \boxed{b} \mid \boxed{c}\,)}^{*}}\,\boxed{a}}}$$

Conversão de uma ER num AF
Exemplo

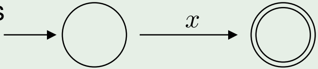
$\mathcal{Q}$ Construa um autómato equivalente à expressão regular $e = a|a(a|b|c)^*a$

$\mathcal{R}$

1. Decomposição



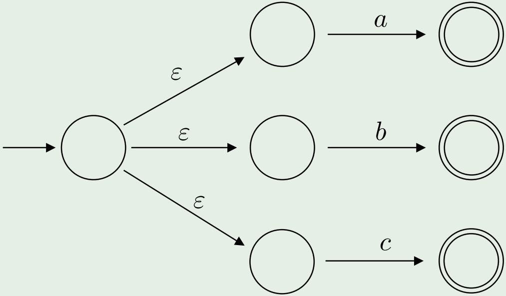
2. Autómatos primitivos



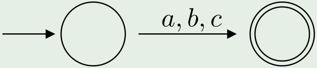
com $x = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

Conversão de uma ER num AF
Exemplo

3. Reunião para obter $(a|b|c)$

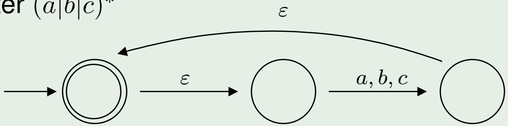


4. Simplificando

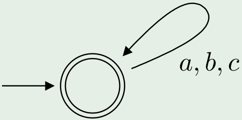


Conversão de uma ER num AF
Exemplo

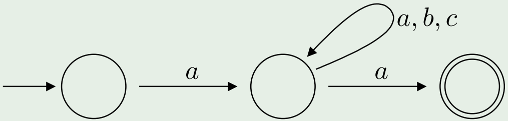
5. Fecho para obter $(a|b|c)^*$



6. Simplificando

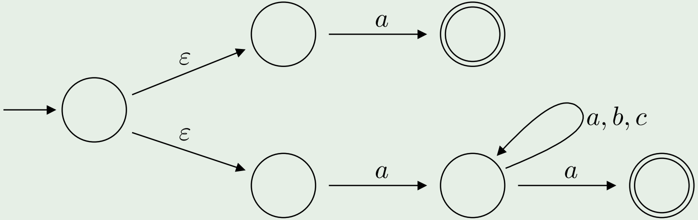


7. Concatenação (já com simplificação) para obter $a(a|b|c)^*a$

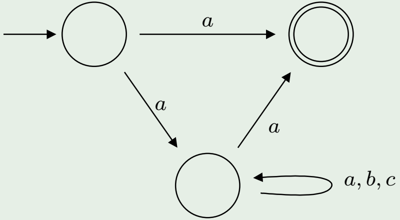


Conversão de uma ER num AF
Exemplo

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



9. Simplificando



Autómato finito generalizado (AFG)
Definição

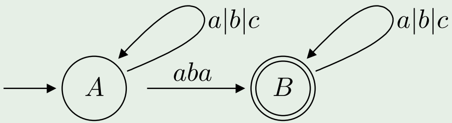
$\mathcal{D}$ Um **autómato finito generalizado** (AFG) é um quíntuplo $M = (A, Q, q_0, \delta, F)$, em que:

- 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
- $F \subseteq Q$ é o conjunto dos estados de aceitação

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

Autómato finito generalizado (AFG)
Exemplo

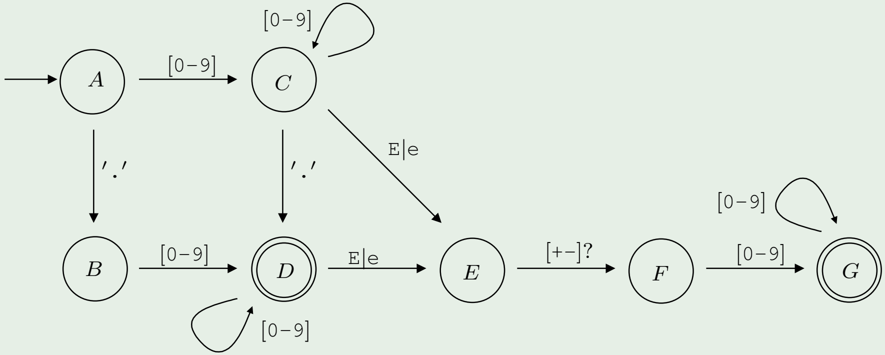
- O AFG seguinte representa o conjunto das palavras, definidas sobre o alfabeto $A = \{a, b, c\}$, que contêm a sub-palavra aba



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

Autómato finito generalizado (AFG)
Exemplo

- O AFG seguinte representa as constantes reais em C

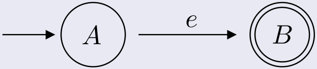


- Note que se usou `'.'` e não `.`, porque o último é uma expressão regular que representa qualquer letra do alfabeto

Conversão de um AFG numa ER

Abordagem

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



designa-se por **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, e é uma expressão regular equivalente ao autómato
- O processo de conversão resume-se assim à conversão de AFG à forma reduzida

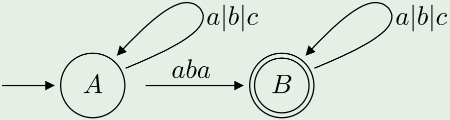
Conversão de um AFG numa ER
Algoritmo de conversão

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 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 estados intermédios
   - Os estados são eliminados um a um, em processos de transformação que mantêm a equivalência

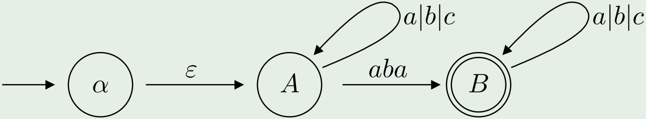
Conversão de um AFG numa ER
Ilustração com um exemplo

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 ε para o antigo

antes



depois

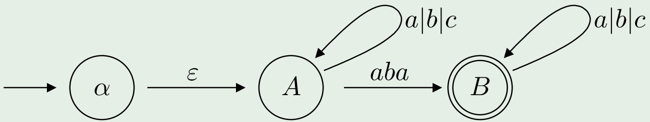


Conversão de um AFG numa ER

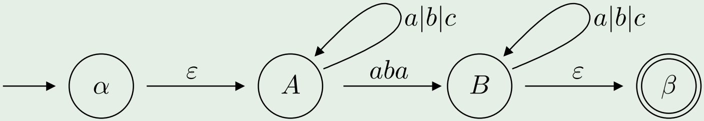
Ilustração com um exemplo

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

antes



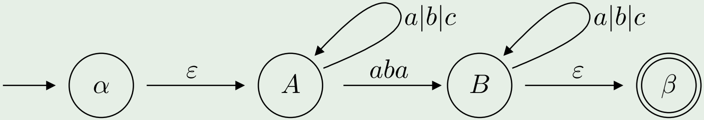
depois



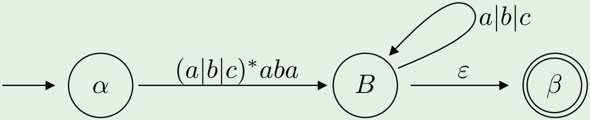
Conversão de um AFG numa ER
Ilustração com um exemplo

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



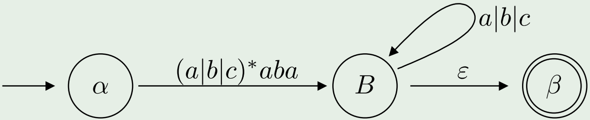
depois da eliminação de A



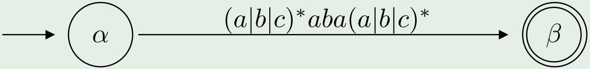
Conversão de um AFG numa ER
Ilustração com um exemplo

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



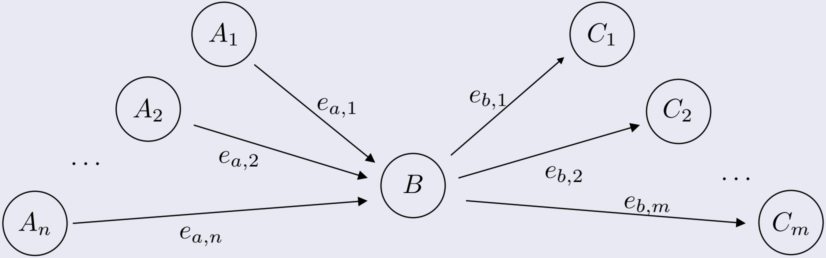
depois da eliminação de B



- Sendo $\mathtt{(a|b|c)^*aba(a|b|c)^*}$ a expressão regular pretendida

Conversão de um AFG numa ER
Algoritmo de eliminação de um estado

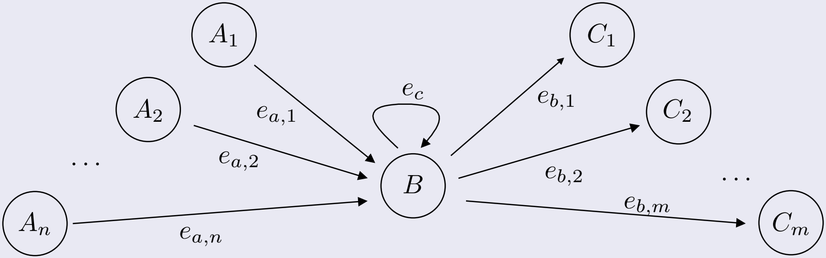
- Caso em que o estado a eliminar (B) **não tem** transições de si para si



- Pode acontecer que haja $A_i = C_j$
- Para ir de A_i para C_j através de B, para $i = 1, 2, \cdots, n$ e $j = 1, 2, \cdots, m$, 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

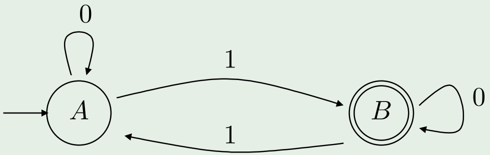
- Caso em que o estado a eliminar (B) **tem** transições de si para si



- Pode acontecer que haja $A_i = C_j$
- Para ir de A_i para C_j através de B, para $i = 1, 2, \cdots, n$ e $j = 1, 2, \cdots, m$, 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 $(e_{a,i})(e_c)^*(b,j)$
- Esta transição fica em paralelo com uma que já exista

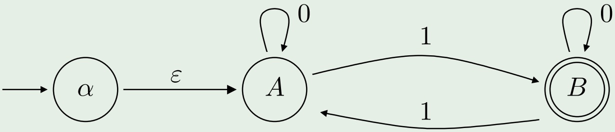
Conversão de um AFG numa ER
Exercício

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



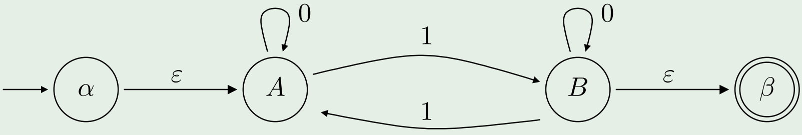
$\mathcal{R}$ Aplique-se passo a passo o algoritmo de conversão

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

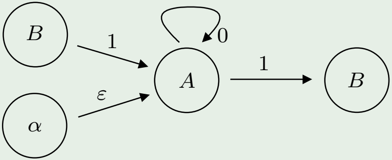


Exemplo de conversão de um AFG numa ER
Exercício

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



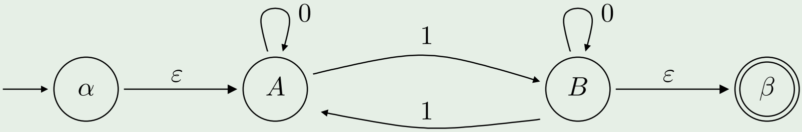
- Elimine-se o estado A. Para isso é preciso ver os segmentos de caminhos que passam por A.



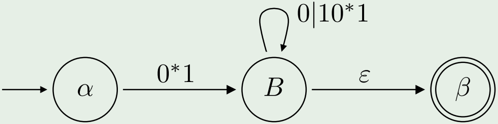
- Note que B aparece à esquerda e à direita

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 algorimo de conversão



- Eliminando o estado A obtém-se



- Finalmente, eliminando o estado B obtém-se a ER $\texttt{0}^*\texttt{1}(\texttt{0}|\texttt{1}\texttt{0}^*\texttt{1})^*$