Teoria da Computação

Expressões Regulares e Autômatos Finitos

Thiago Alves

Introdução

- Expressões Regulares e Autômatos Finitos são bem diferentes
- Será que são equivalentes com relação as linguagens que geram e aceitam?
- O que temos que provar?

Introdução

- Temos que mostrar que qualquer Expressão Regular pode ser convertida em um Autômato Finito que aceita a mesma linguagem que ela descreve e vice-versa
- Dessa forma, também provamos que uma linguagem é regular se e somente se é gerada por um ER

ER → AF

- Como converter Expressões Regulares em Autômatos Finitos?

ER → AF

- Como converter Expressões Regulares em Autômatos Finitos?
- Como converter a Expressão Regular **0** em um Autômato Finito?
- Como converter a Expressão Regular **1** em um Autômato Finito?
- Como converter a Expressão Regular **ε** em um Autômato Finito?

ER → AF

- Como converter a Expressão Regular **0+1** em um Autômato Finito?
- Como converter a Expressão Regular **(0+1)0** em um Autômato Finito?
- Como converter a Expressão Regular **((0+1)0)*** em um Autômato Finito?

ER → AFN

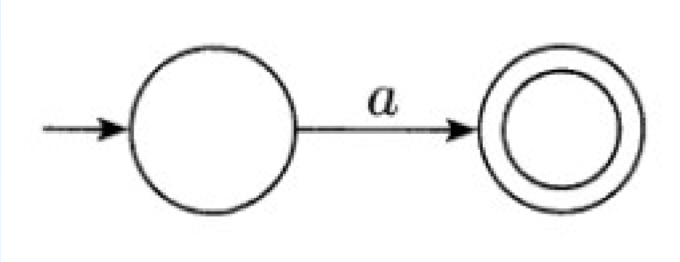
- Como converter expressões regulares em autômatos finitos?
- Qual tipo de autômato é mais adequado?
- Podemos usar a definição indutiva de expressões regulares para construir um procedimento recursivo

ER → AFN

- Se uma linguagem é gerada por uma expressão regular então a linguagem é regular
- Indução na definição de ER
  - Base: E = a
  - L(a) = {a}
  - Qual seria um AFN equivalente?

ER → AFN

- Indução na definição de ER
  - Base: E = a
  - L(a) = {a}
  - Qual seria um AFN equivalente?

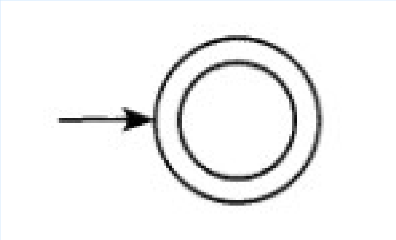


ER → AFN

- Indução na definição de ER
  - Base: $E = \varepsilon$
  - $L(\varepsilon) = \{\varepsilon\}$
  - Qual seria um AFN equivalente?

ER → AFN

- Indução na definição de ER
  - Base: $E = \varepsilon$
  - $L(\varepsilon) = \{\varepsilon\}$
  - Qual seria um AFN equivalente?

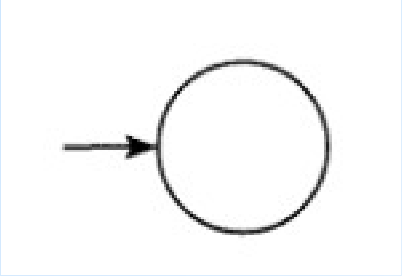


ER → AFN

- Indução na definição de ER
  - Base: $E = \emptyset$
  - $L(\emptyset) = \emptyset$
  - Qual seria um AFN equivalente?

ER → AFN

- Indução na definição de ER
  - Base: E = ∅
  - L(∅) = ∅
  - Qual seria um AFN equivalente?



ER → AFN

- Indução na definição de ER
- HI:
  - Se E_1 é uma ER então existe um AFN A_1 tal que $L(E_1) = L(A_1)$
  - Se E_2 é uma ER então existe um AFN A_2 tal que $L(E_2) = L(A_2)$

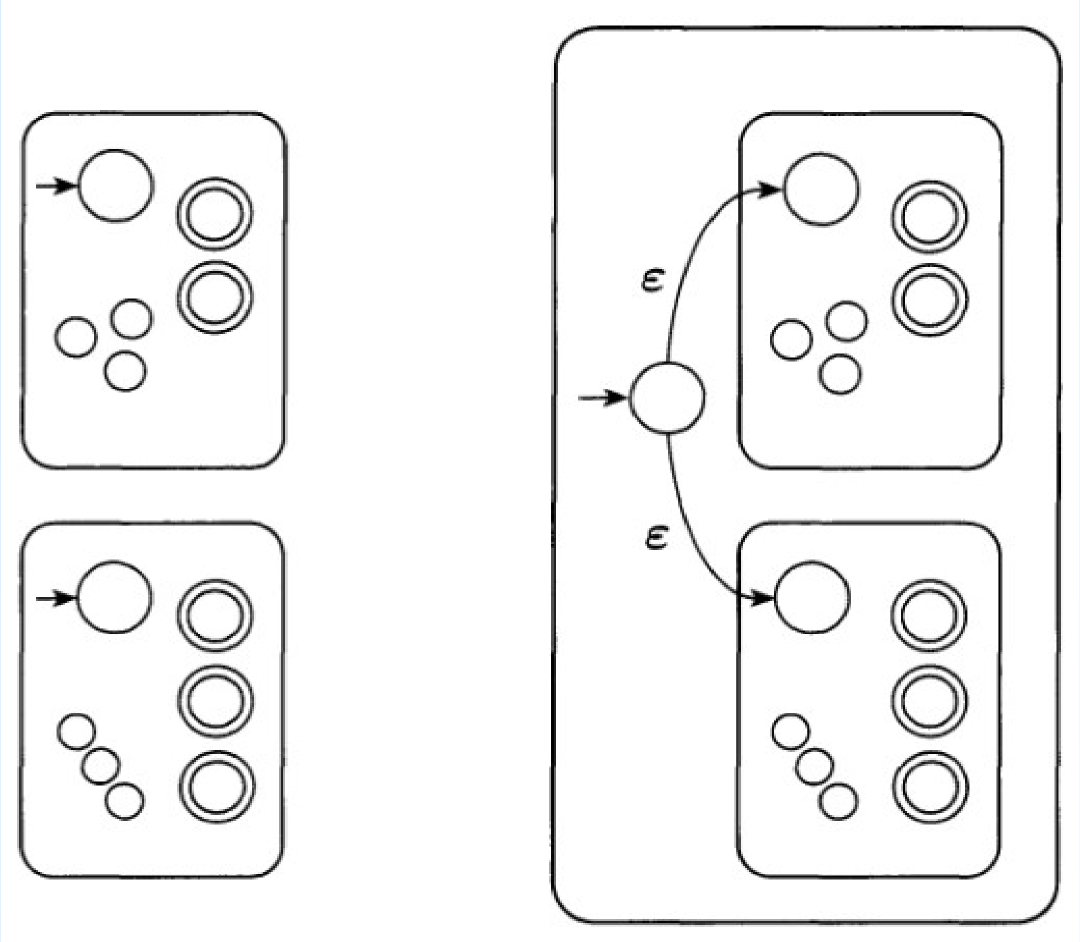
ER → AFN

- Indução na definição de ER
- PI:
  - Seja $E = (E_1 + E_2)$
  - $L(E) = L(E_1) \cup L(E_2)$
  - E_1 é ER. Pela HI, existe A_1 tal que $L(E_1) = L(A_1)$
  - E_2 é ER. Pela HI, existe A_2 tal que $L(E_2) = L(A_2)$

ER → AFN

- $L(E) = L(A_1) \cup L(A_2)$
- Como fazer um AFN A tal que $L(A) = L(A_1) \cup L(A_2)$?

ER → AFN



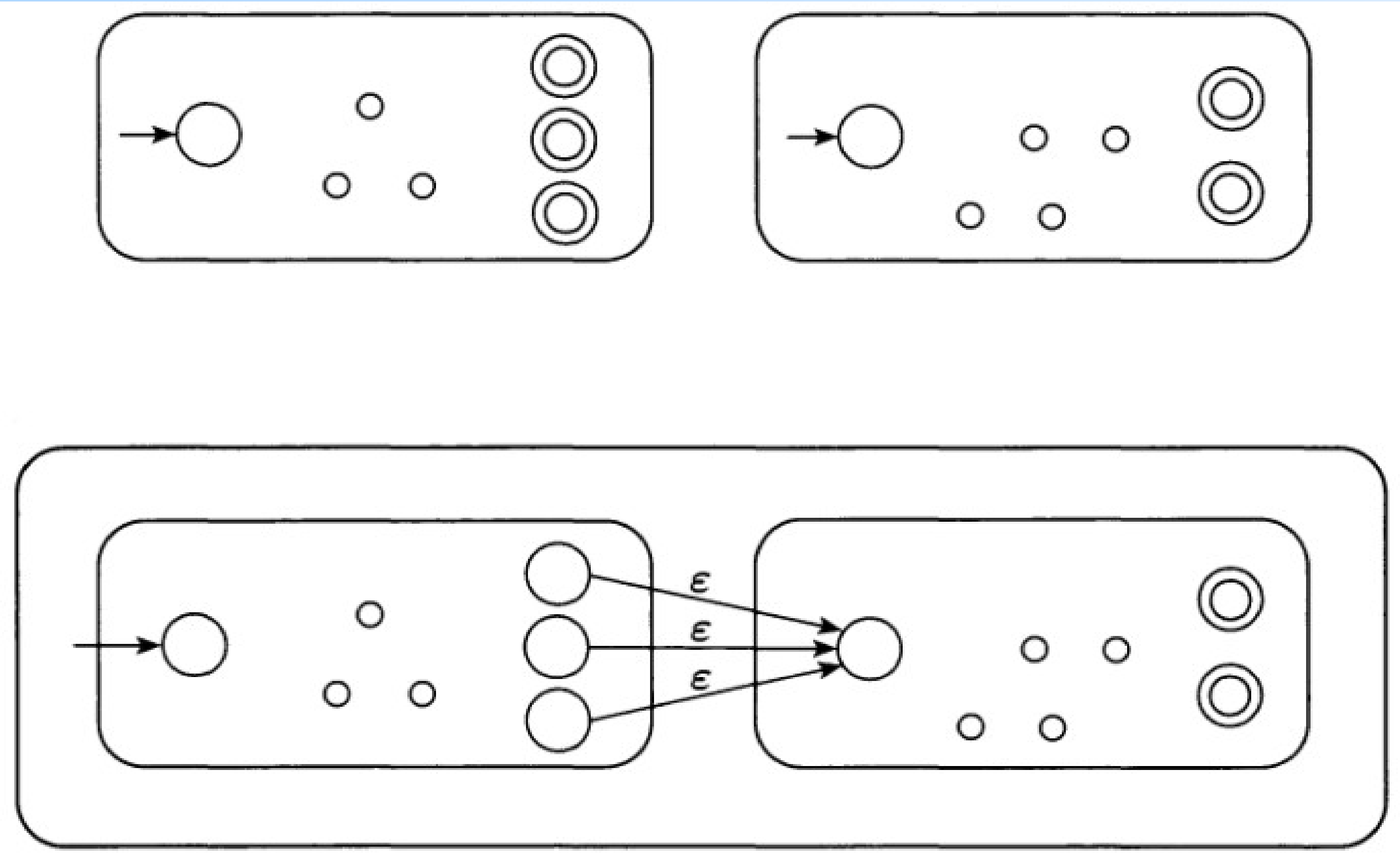
ER → AFN

- PI:
  - Seja $E = (E_1E_2)$
  - $L(E) = L(E_1)L(E_2)$
  - E_1 é ER. Pela HI, existe A_1 tal que $L(E_1) = L(A_1)$
  - E_2 é ER. Pela HI, existe A_2 tal que $L(E_2) = L(A_2)$

ER → AFN

- $L(E) = L(A_1)L(A_2)$
- Como fazer um AFN A tal que $L(A) = L(A_1)L(A_2)$?

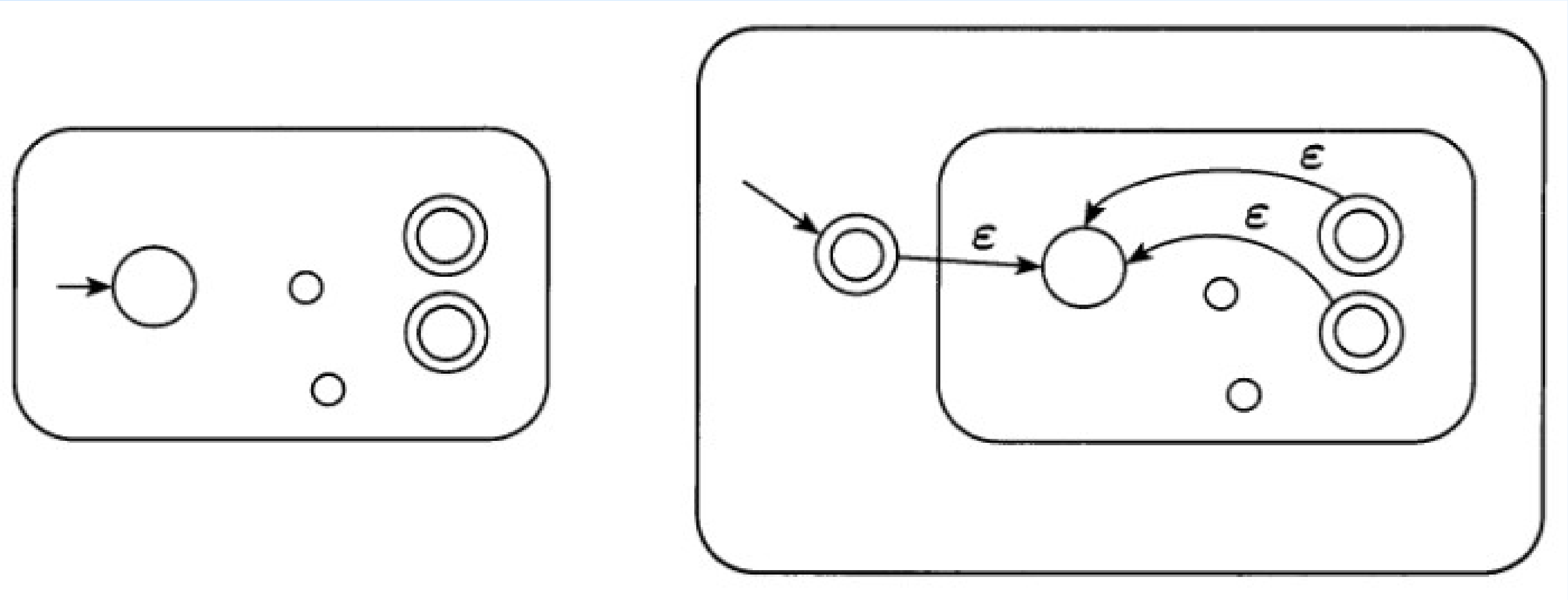
ER → AFN



ER → AFN

- PI:
  - Seja E = (E_1*)
  - L(E) = L(E_1)*
  - E_1 é ER. Pela HI, existe A_1 tal que L(E_1) = L(A_1)
  - L(E) = L(A_1)*
  - Como fazer um AFN A tal que L(A) = L(A_1)*?

ER → AFN

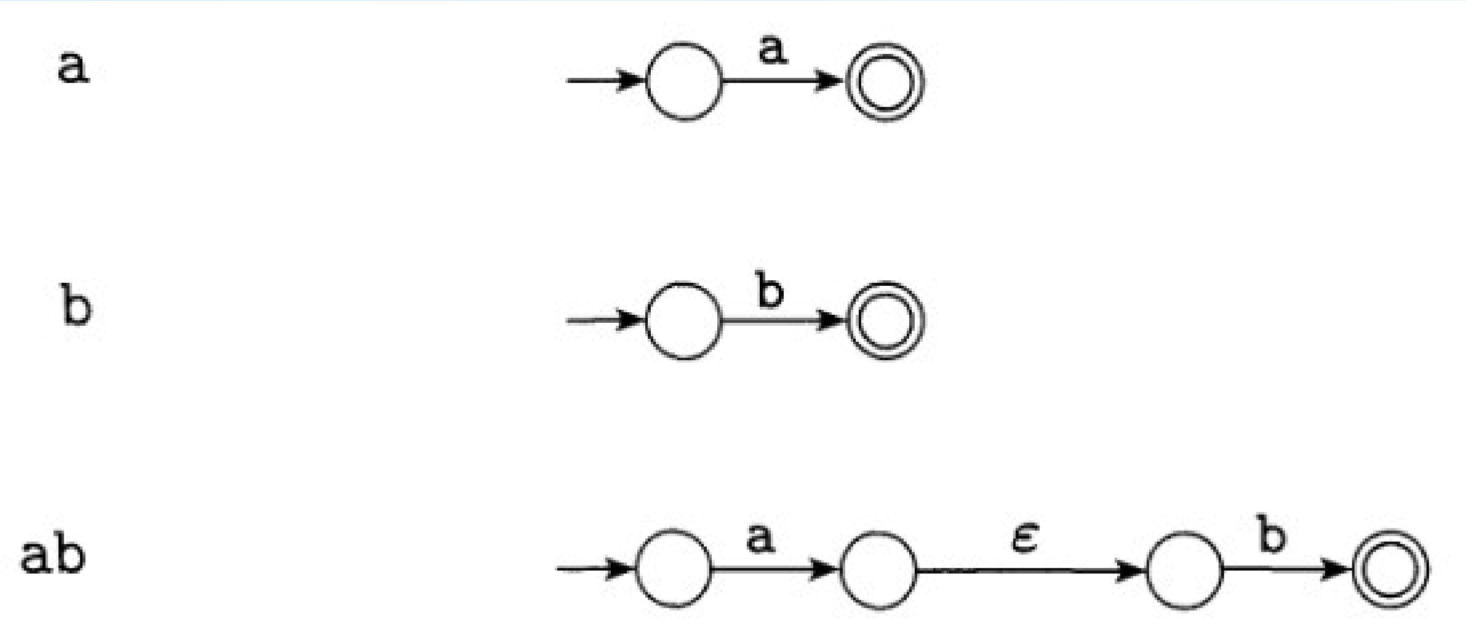


Exemplos

- Vamos converter (**ab** + **a**)* para um AFN

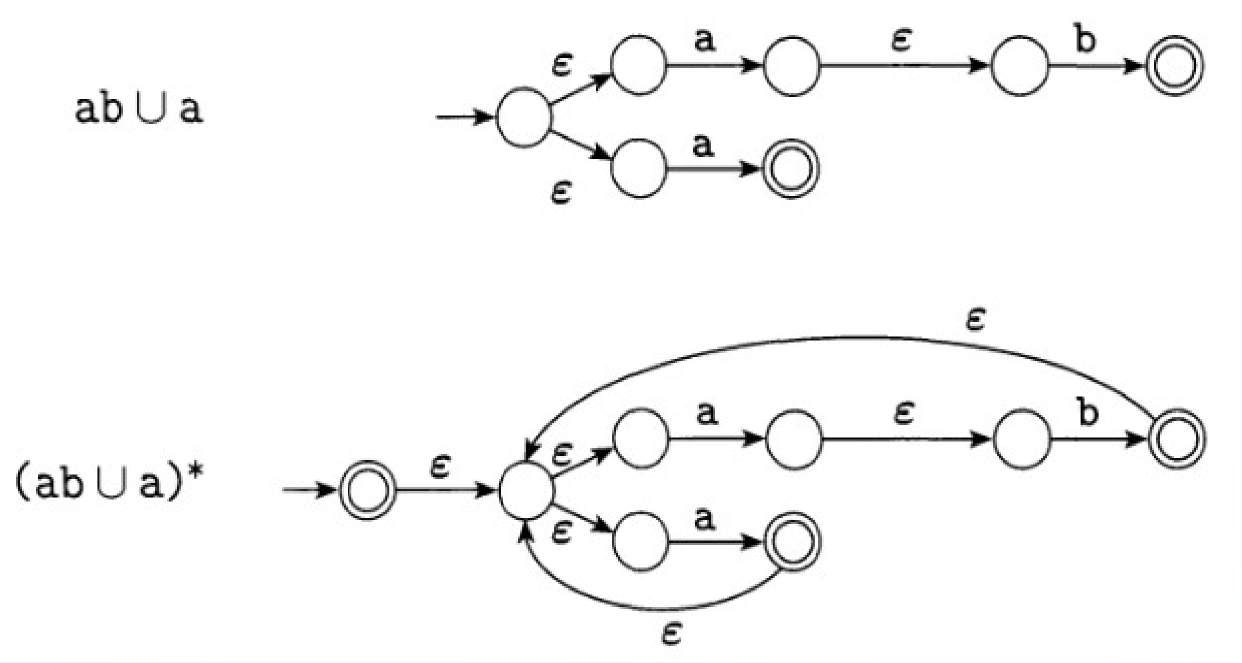
Exemplos

- Vamos converter (**ab** + **a**)* para um AFN



Exemplos

- Vamos converter (**ab** + **a**)* para um AFN

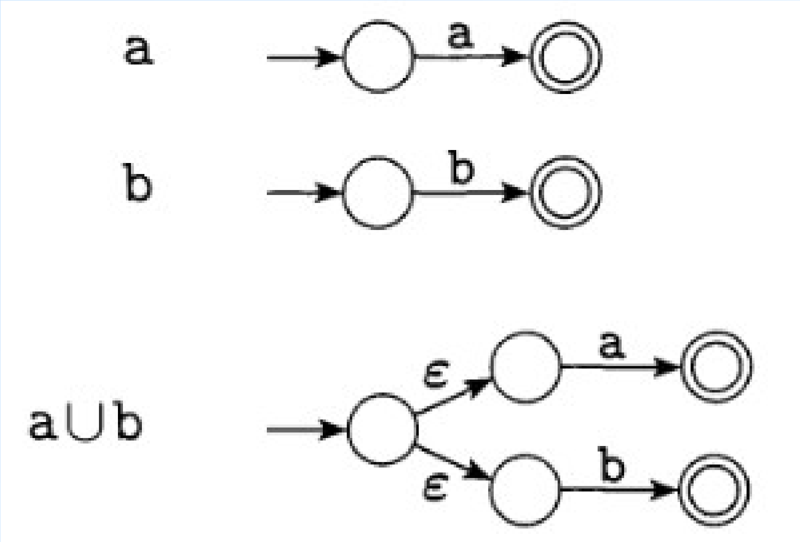


Exemplos

- Vamos converter (**a** + **b**)***aba** para um AFN

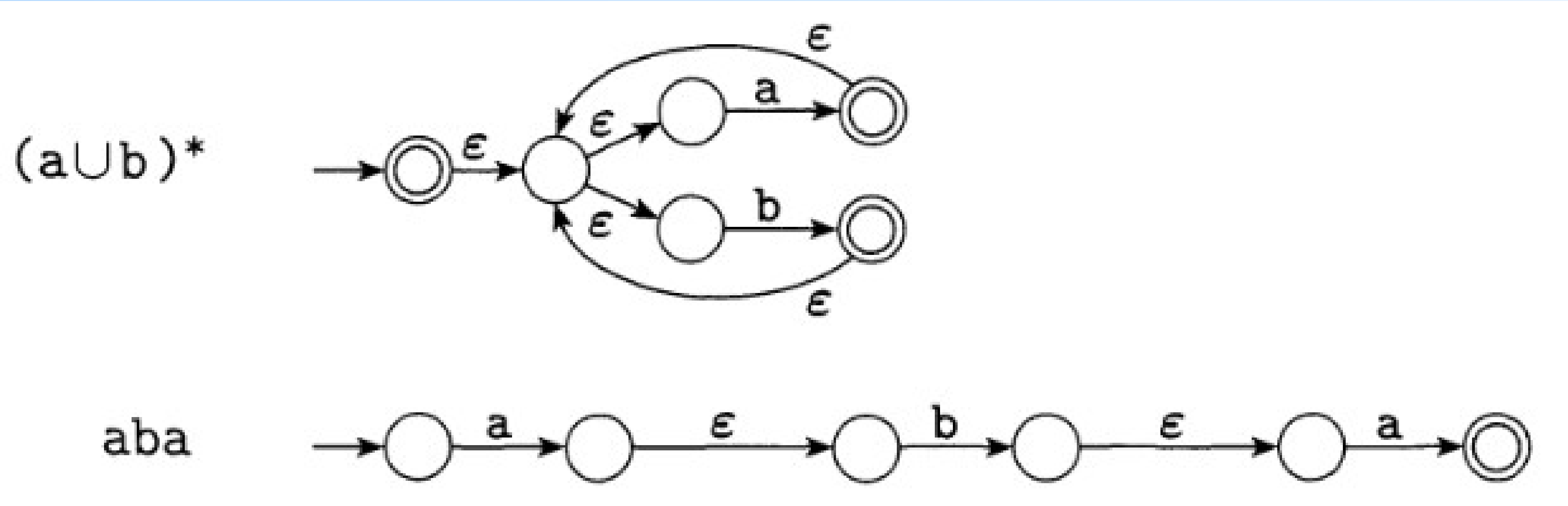
Exemplos

- Vamos converter (**a** + **b**)*aba para um AFN



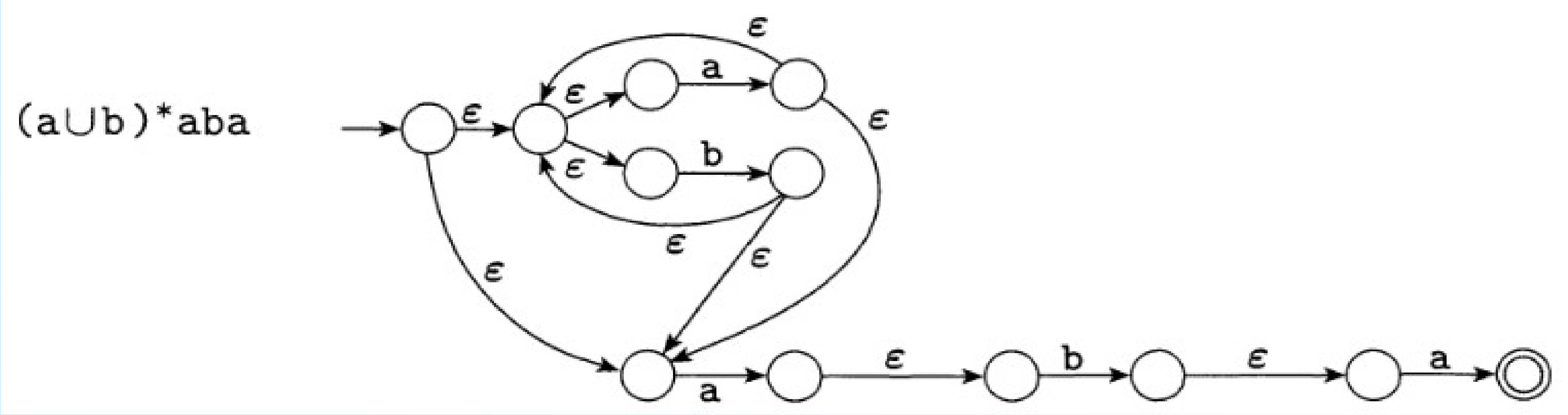
Exemplos

- Vamos converter (**a** + **b**)***aba** para um AFN



Exemplos

- Vamos converter (**a** + **b**)*__aba__ para um AFN



AF → ER

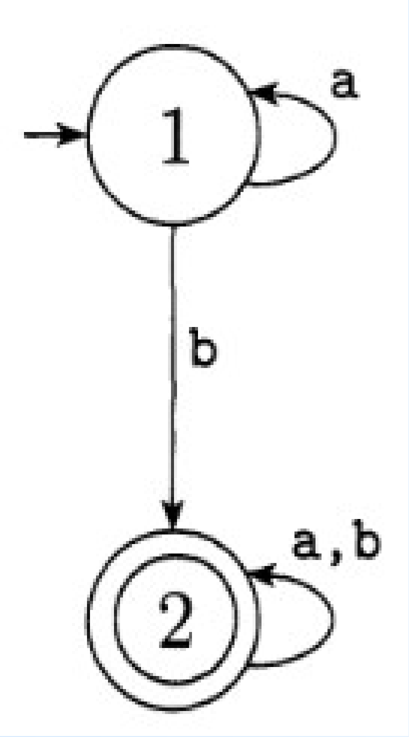
- Temos que mostrar que qualquer autômato finito A pode ser convertido em uma expressão regular E tal que L(A) = L(E)
- Qual tipo de autômato finito vamos escolher?

AF → ER

- Temos que mostrar que qualquer autômato finito A pode ser convertido em uma expressão regular E tal que L(A) = L(E)
- Qual tipo de autômato finito vamos escolher?
  - Determinístico é mais simples

AFD → ER

- Como converter autômatos finitos determinísticos em expressões regulares?



AFD → ER

- Os rótulos das transições nos caminhos entre o estado inicial e os estados finais representam as strings aceitas pelo autômato finito determinístico
  - Vamos remover estados e trocar as transições por expressões regulares

AFD → ER

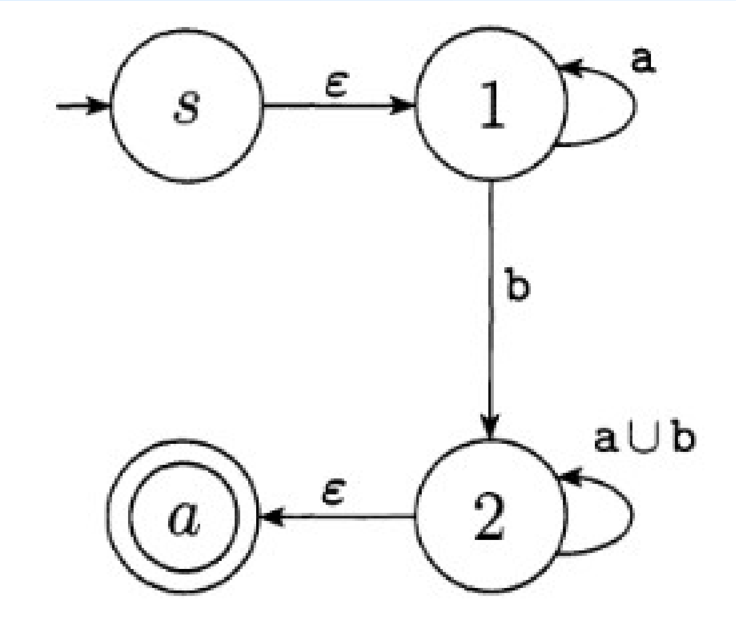
- 1) Criar um novo inicial para ter apenas transições saindo dele e tirar os laços dele
- 2) Criar um novo estado final para ser único e ter apenas transições chegando nele
- 3) Incluir transições com ∅ onde não tiver transição

AFD → ER

- Transições com múltiplos rótulos podem ser convertidas para expressões regulares usando a operação + de união

AFD → ER

- Não mostramos as transições com ∅ para não poluir a figura



AFD → ER

- Tirar um estado e adaptar as transições com expressões regulares até sobrar apenas o inicial e o final
  - A transição entre eles é a expressão regular equivalente ao autômato finito determinístico original

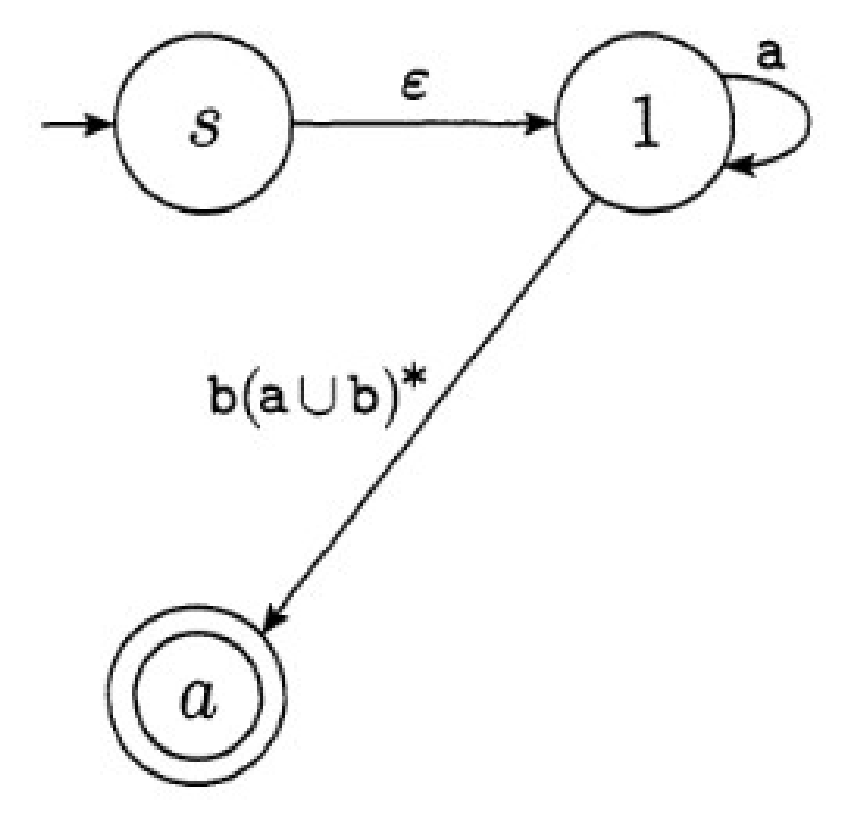
AFD → ER

- Tirar um estado e adaptar as transições com expressões regulares até sobrar apenas o inicial e o final
  - Como adaptar as transições com expressões regulares?

AFD → ER

- Como adaptar as transições com expressões regulares?
  - Concatenar a expressão regular chegando no estado com a do laço e com com a saindo do estado
  - Fazer a união com a expressão regular da transição já existente entre os estados

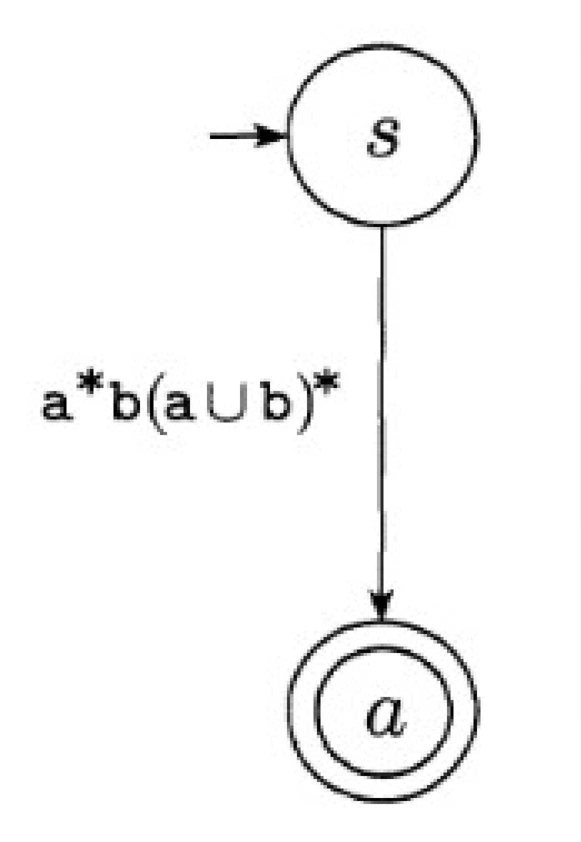
AFD → ER



AFD → ER

- Definição da nova transição
  - q_r é o estado removido
  - q_i, q_j são os outros pares de estados
  - $\delta'(q_i, q_j) = \delta(q_i, q_r)(\delta(q_r, q_r))^*\delta(q_r, q_j) + \delta(q_i, q_j)$

AFD → ER



Algoritmo

CONVERT(G):

1. Let k be the number of states of G.
2. If $k = 2$, then G must consist of a start state, an accept state, and a single arrow connecting them and labeled with a regular expression R.
   Return the expression R.
3. If $k > 2$, we select any state $q_{\text{rip}} \in Q$ different from q_{start} and q_{accept} and let G' be the GNFA $(Q', \Sigma, \delta', q_{\text{start}}, q_{\text{accept}})$, where

   $$Q' = Q - \{q_{\text{rip}}\},$$

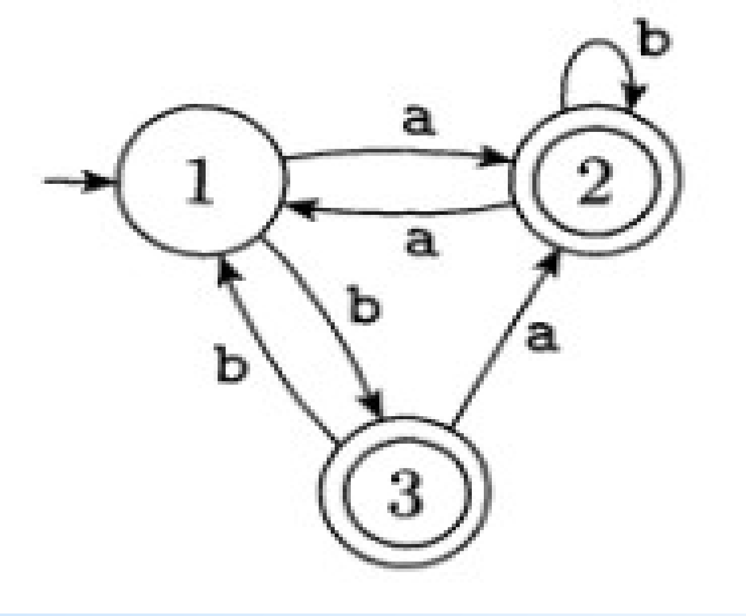
   and for any $q_i \in Q' - \{q_{\text{accept}}\}$ and any $q_j \in Q' - \{q_{\text{start}}\}$ let

   $$\delta'(q_i, q_j) = (R_1)(R_2)^*(R_3) \cup (R_4),$$

   for $R_1 = \delta(q_i, q_{\text{rip}})$, $R_2 = \delta(q_{\text{rip}}, q_{\text{rip}})$, $R_3 = \delta(q_{\text{rip}}, q_j)$, and $R_4 = \delta(q_i, q_j)$.
4. Compute CONVERT(G') and return this value.

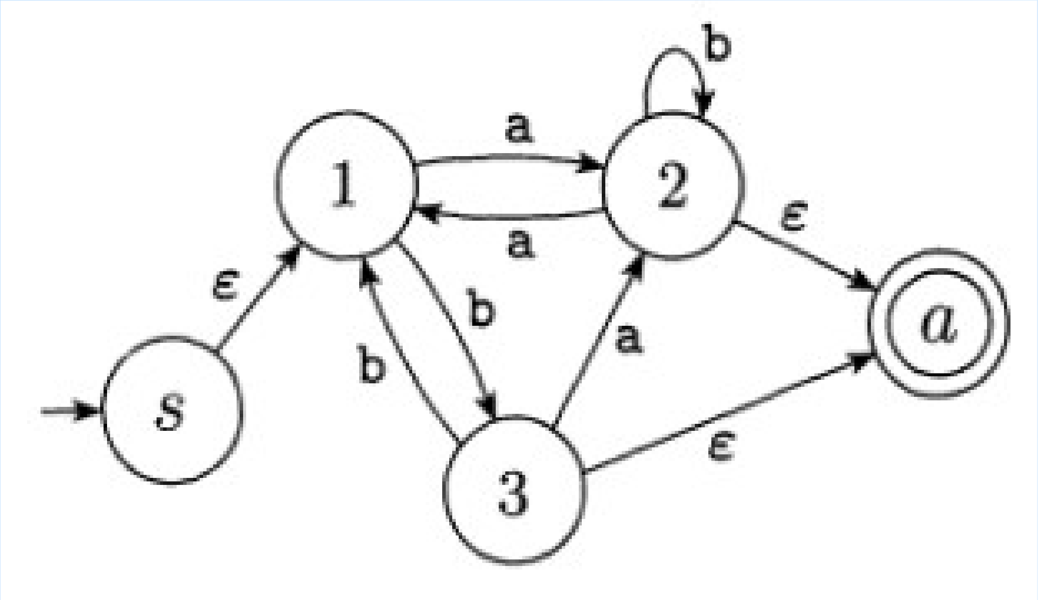
Exemplo

- Converta o AFD abaixo em uma ER
  - Criar novo estado inicial e final



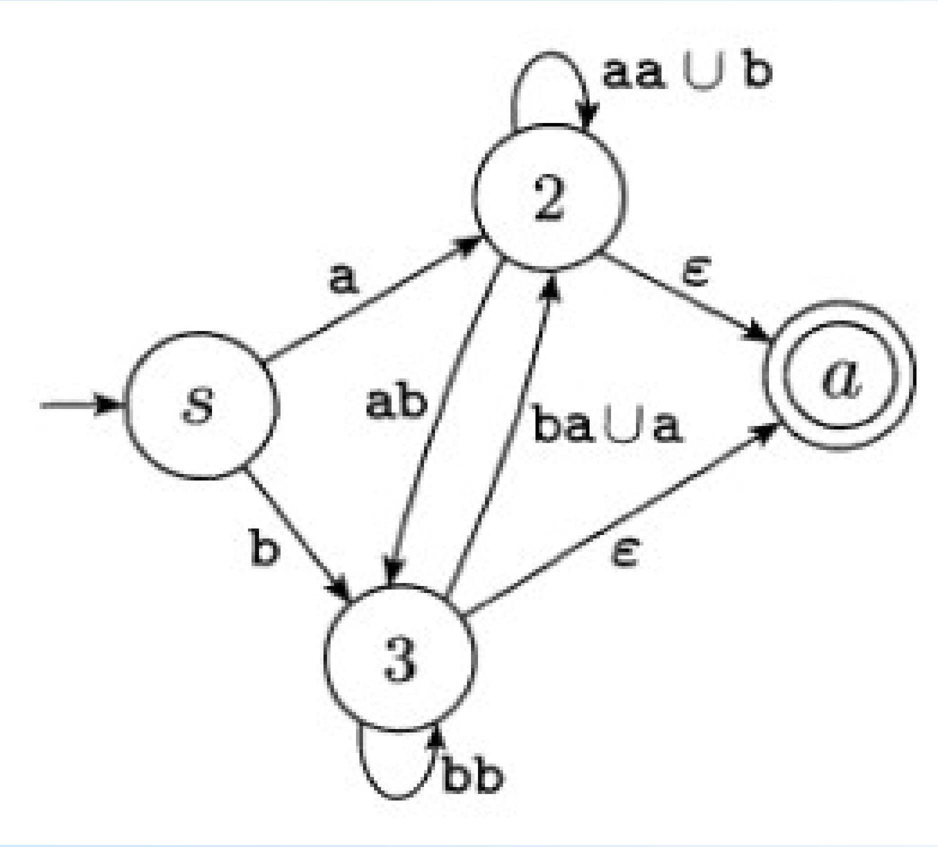
Exemplo

- Remover o estado 1



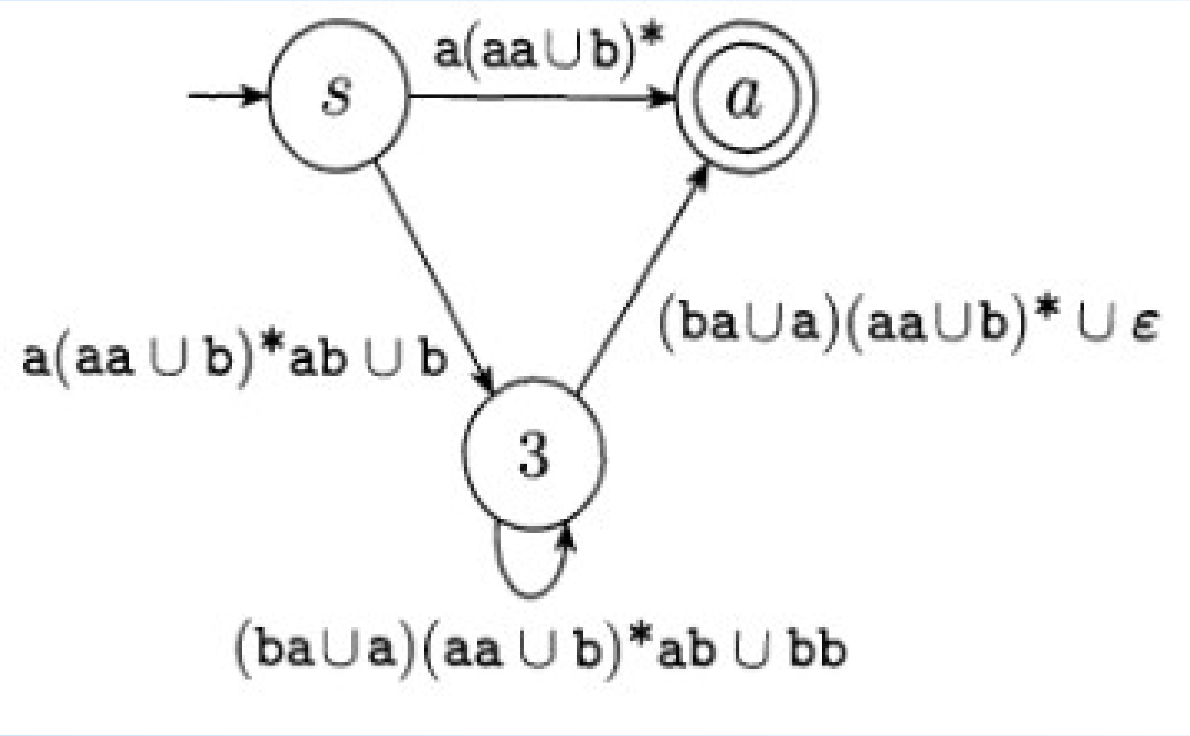
Exemplo

- Remover o estado 2



Exemplo

- Remover o estado 3



Exemplo

$$(a(aa\cup b)^*ab\cup b)((ba\cup a)(aa\cup b)^*ab\cup bb)^*((ba\cup a)(aa\cup b)^*\cup\varepsilon)\cup a(aa\cup b)^*$$