ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

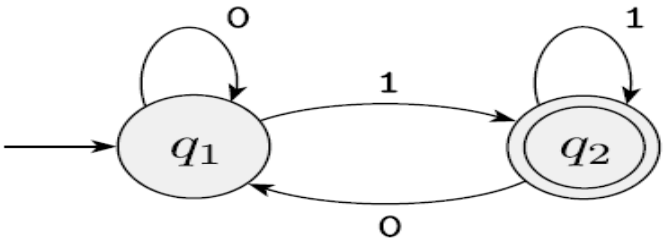
Aula 7

Expressões Regulares

Profa. Ariane Machado Lima
ariane.machado@usp.br

Aulas anteriores

AFD

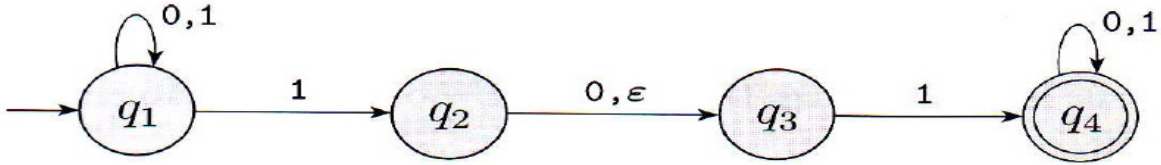


Para cada par (estado atual, próximo símbolo) está DETERMINADO qual é o próximo estado

Um ***autômato finito*** é uma 5-upla $(Q, \Sigma, \delta, q_0, F)$, onde

1. Q é um conjunto finito conhecido como os ***estados***,
2. Σ é um conjunto finito chamado o ***alfabeto***,
3. $\delta: Q \times \Sigma \longrightarrow Q$ é a ***função de transição***,[1]
4. $q_0 \in Q$ é o ***estado inicial***, e
5. $F \subseteq Q$ é o ***conjunto de estados de aceitação***.[2]

AFN



Um ***autômato finito não-determinístico*** é uma 5-upla $(Q, \Sigma, \delta, q_0, F)$, onde

1. Q é um conjunto finito de estados,
2. Σ é um alfabeto finito,
3. $\delta: Q \times \Sigma_\varepsilon \longrightarrow \mathcal{P}(Q)$ é a função de transição,
4. $q_0 \in Q$ é o estado inicial, e
5. $F \subseteq Q$ é o conjunto de estados de aceitação.

Para cada par (estado atual, próximo símbolo – incluindo ε) há um conjunto de estados possíveis

Propriedades de fechamento de ling. regulares

- Vimos como provar que a classe de linguagens regulares é fechada sob uma determinada operação
- Em particular, é fechada sob as operações:
  - União
  - Intersecção
  - Concatenação
  - Estrela
  - Complemento
- É também fechada sob outras… (ex: sufixo)
- Novamente, saber isso é importante para sabermos se podemos ou não criar um AFD/AFN para uma linguagem de interesse...

Aula de hoje

AFD <=> AFN <=> Expressões Regulares

Definição de Expressões Regulares

Expressões regulares

- Uma linguagem é um conjunto de cadeias
- Um conjunto de cadeias pode ser descrito por uma expressão
- Ex: como você escreve no campo “Pesquisar” do computador que você quer localizar todos os arquivos que começam com “ACH2043” e terminam com “.pdf”?

Expressões regulares

- Uma linguagem é um conjunto de cadeias
- Um conjunto de cadeias pode ser descrito por uma expressão
- Ex: como você escreve no campo “Pesquisar” do computador que você quer localizar todos os arquivos que começam com “ACH2043” e terminam com “.pdf”?

ACH2043*.pdf

Expressões regulares

- Uma linguagem é um conjunto de cadeias
- Um conjunto de cadeias pode ser descrito por uma expressão
- Ex: como você escreve no campo “Pesquisar” do computador que você quer localizar todos os arquivos que começam com “ACH2043” e terminam com “.pdf”?

  ACH2043*.pdf

- Isso é uma expressão que descreve um conjunto de cadeias

Expressões regulares

- Exemplo: o que essa expressão descreve?

  ({0} U {1}) ° {0}*

Expressões regulares

- Exemplo: o que essa expressão descreve?

  ({0} U {1}) ° {0}*

  Cadeias que comecem com 0 ou 1 seguido de zero ou mais 0's

Expressões regulares

- Exemplo: o que essa expressão descreve?

  ({0} U {1}) ° {0}*

  Cadeias que comecem com 0 ou 1 seguido de zero ou mais 0's

- Simplificação da notação:

  (0 U 1)0*

  Às vezes (0|1)0*

Aplicações de ERs

- Compiladores - definição de tokens para geração de analisadores léxicos. Ex: número é $\{0, 1, \ldots, 9\}^+$
- Programas utilitários (ex: grep)
- Linguagens de programação (ex: awk, perl, python)
- Ex: Python:

\w : {a, …, z, A, …, Z, 0, …, 9, _} → [a-zA-Z0-9_]

```
>>> text = "He was carefully disguised but captured quickly by police."
>>> re.findall(r"\w+ly", text)
['carefully', 'quickly']
```

https://docs.python.org/3/library/re.html

Expressões regulares

- Regras de precedência (da maior para a menor):

  Estrela

  Concatenação

  União

- Outros exemplos:
  - (0 U 1)*
  - 0Σ*
  - 0Σ* U Σ*1

Expressões regulares

DEFINIÇÃO 1.52

Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.

Expressões regulares (ER)

- Abreviações:

  $R^+ = RR^*$ (concatenação de 1 ou mais R's)

  $R? = \{\varepsilon\} \cup R$ (0 ou 1 ocorrência de R)

- Se R é uma ER, dizemos que L(R) é a linguagem descrita por R

Exemplos de ERs (Σ = {0,1})

- 0*10* =
- Σ*1Σ* =
- (ΣΣΣ)* =
- 0Σ*0 U 1Σ*1 U 0 U 1 =

- 1Ø =
- Ø* =

Exemplos de ERs (Σ = {0,1})

- 0*10* = {w | w contém um ÚNICO 1}
- Σ*1Σ* =
- (ΣΣΣ)* =
- 0Σ*0 U 1Σ*1 U 0 U 1 =

- 1Ø =
- Ø* =

Exemplos de ERs (Σ = {0,1})

- 0*10* = {w | w contém um ÚNICO 1}
- Σ*1Σ* = {w | w contém PELO MENOS UM 1}
- (ΣΣΣ)* =
- 0Σ*0 U 1Σ*1 U 0 U 1 =

- 1Ø =
- Ø* =

Exemplos de ERs (Σ = {0,1})

- 0*10* = {w | w contém um ÚNICO 1}
- Σ*1Σ* = {w | w contém PELO MENOS UM 1}
- (ΣΣΣ)* = {w | o comprimento de w é múltiplo de 3}
- 0Σ*0 U 1Σ*1 U 0 U 1 =

- 1Ø =
- Ø* =

Exemplos de ERs (Σ = {0,1})

- 0*10* = {w | w contém um ÚNICO 1}
- Σ*1Σ* = {w | w contém PELO MENOS UM 1}
- (ΣΣΣ)* = {w | o comprimento de w é múltiplo de 3}
- 0Σ*0 U 1Σ*1 U 0 U 1 = {w | w começa e termina com o mesmo símbolo}
- 1Ø =
- Ø* =

Exemplos de ERs (Σ = {0,1})

- 0*10* = {w | w contém um ÚNICO 1}
- Σ*1Σ* = {w | w contém PELO MENOS UM 1}
- (ΣΣΣ)* = {w | o comprimento de w é múltiplo de 3}
- 0Σ*0 U 1Σ*1 U 0 U 1 = {w | w começa e termina com o mesmo símbolo}
- 1Ø = Ø (concatenação com Ø produz Ø)
- Ø* =

Exemplos de ERs (Σ = {0,1})

- 0*10* = {w | w contém um ÚNICO 1}
- Σ*1Σ* = {w | w contém PELO MENOS UM 1}
- (ΣΣΣ)* = {w | o comprimento de w é múltiplo de 3}
- 0Σ*0 U 1Σ*1 U 0 U 1 = {w | w começa e termina com o mesmo símbolo}
- 1*Ø = Ø (concatenação com Ø produz Ø)
- Ø* = { ε } (operador * concatena qualquer número de cadeias para obter uma cadeia no resultado)

Igualdades válidas

- $R\emptyset = \emptyset$
- $\emptyset^* = \{ \varepsilon \}$
- $R \cup \emptyset = R$
- $R\varepsilon = R$

Exp. regulares e autômatos finitos

- Qual a relação entre linguagens descritas por expressões regulares e linguagens reconhecidas por autômatos finitos?

Exp. regulares e autômatos finitos

- Qual a relação entre linguagens descritas por expressões regulares e linguagens reconhecidas por autômatos finitos?
  - Pertencem ao mesmo conjunto! (Linguagens regulares)
  - Expressões regulares são equivalentes a autômatos finitos

Exp. regulares e autômatos finitos

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Exp. regulares e autômatos finitos

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

- Para isso, o que eu preciso mostrar?

Exp. regulares e autômatos finitos

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

- Para isso, o que eu preciso mostrar?
  - Que eu consigo converter um AFD ou AFN em uma expressão regular (ER) equivalente (=>)
  - E que eu consigo converter uma ER em um AFD/AFN (<=)

Equivalência de ERs e AFs

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Prova:

Parte 1: (<=)

LEMA 1.55

Se uma linguagem é descrita por uma expressão regular, então ela é regular.

Prova parte 1: vamos construir um AFN que reconheça L(R), e portanto L(R) será regular (ou seja, vamos mostrar como converter uma ER em um AFN)

Parte 2: (=>)

LEMA 1.60

Se uma linguagem é regular, então ela é descrita por uma expressão regular.

Prova parte 2: se L é regular então um existe um AFD que a descreve. Então vamos mostrar que é sempre possível (e mostraremos como) converter um AFD em uma ER equivalente.

Equivalência de ERs e AFs

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Prova:

Parte 1: (<=)

LEMA 1.55

Se uma linguagem é descrita por uma expressão regular, então ela é regular.

Prova parte 1: vamos construir um AFN que reconheça L(R), e portanto L(R) será regular (ou seja, vamos mostrar como converter uma ER em um AFN)

Parte 2: (=>)

LEMA 1.60

Se uma linguagem é regular, então ela é descrita por uma expressão regular.

Prova parte 2: se L é regular então um existe um AFD que a descreve. Então vamos mostrar que é sempre possível (e mostraremos como) converter um AFD em uma ER equivalente.

Equivalência de ERs e AFs – Parte 1

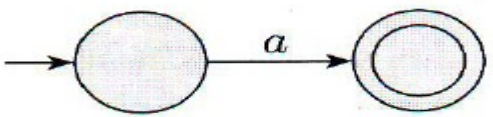
Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.

Equivalência de ERs e AFs – Parte 1

Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.



Equivalência de ERs e AFs – Parte 1

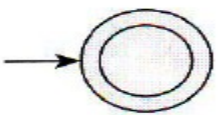
Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.

Equivalência de ERs e AFs – Parte 1

Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.



Equivalência de ERs e AFs – Parte 1

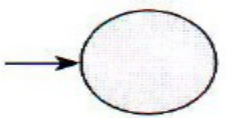
Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.

Equivalência de ERs e AFs – Parte 1

Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.



Equivalência de ERs e AFs – Parte 1

Digamos que R é uma ***expressão regular*** se R for

1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.

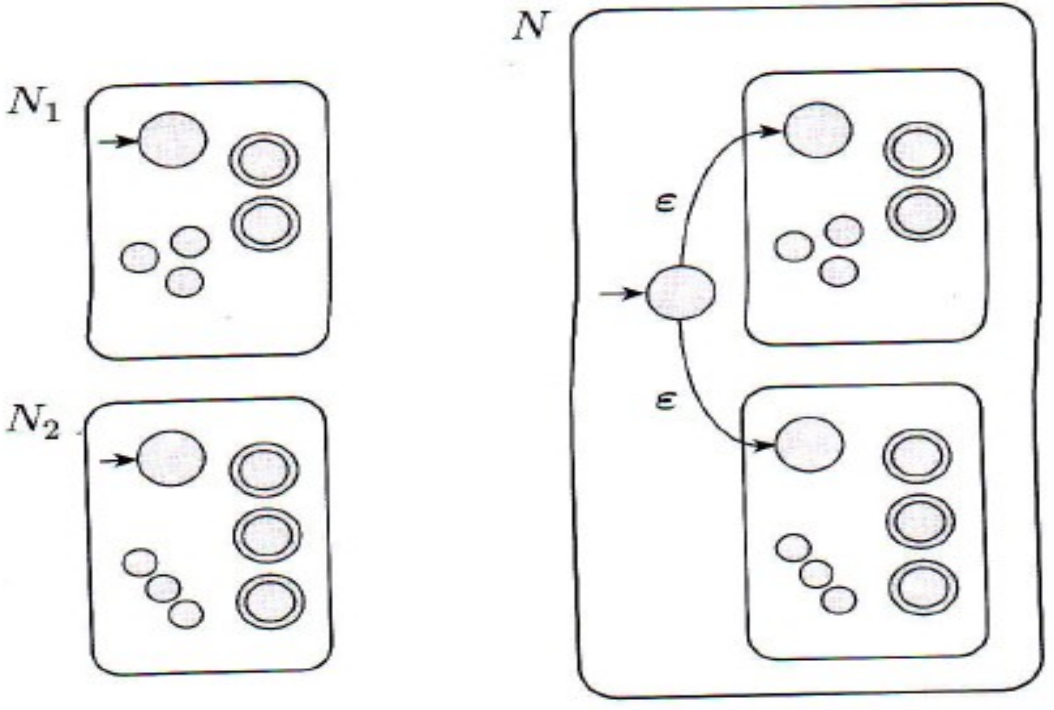
Equivalência de ERs e AFs – Parte 1

Digamos que R é uma ***expressão regular*** se R for

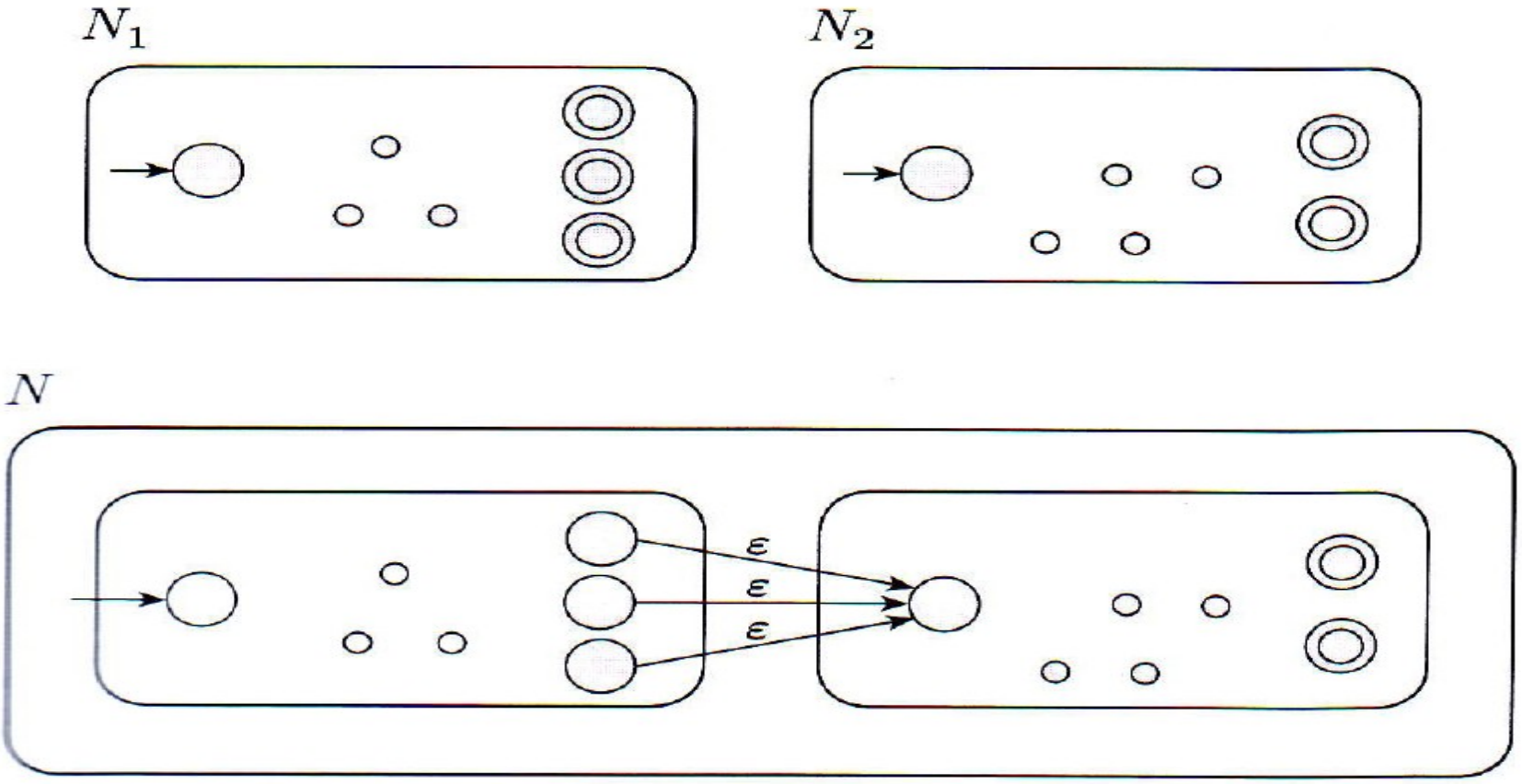
1. a para algum a no alfabeto Σ,
2. ε,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, onde R_1 e R_2 são expressões regulares,
5. $(R_1 \circ R_2)$, onde R_1 e R_2 são expressões regulares, ou
6. (R_1^*), onde R_1 é uma expressão regular.

Provas de fechamento sob operações de união, concatenação e estrela (que vimos na aula passada)

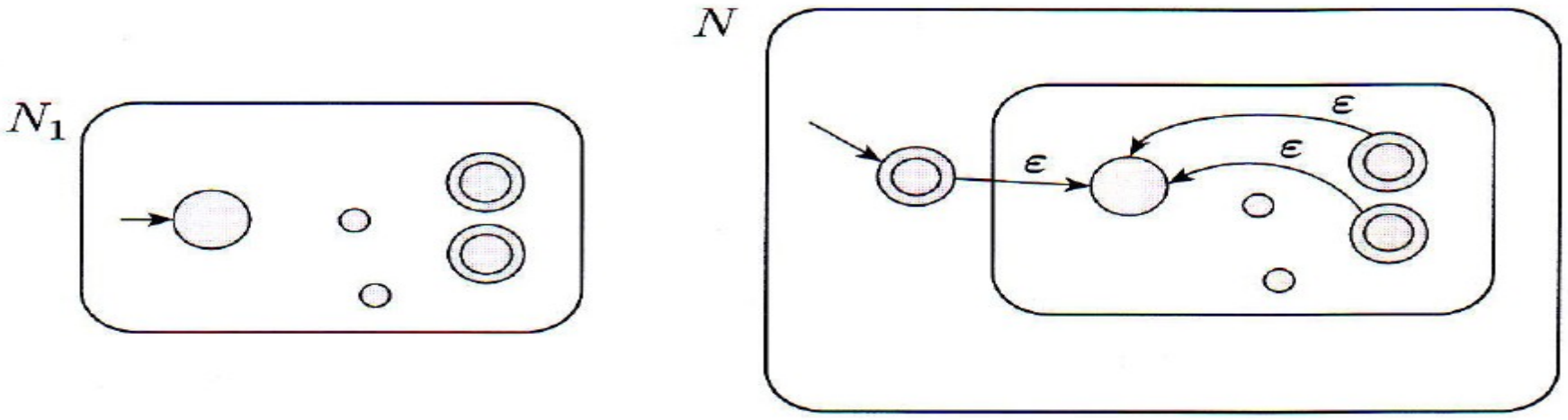
AFN união de dois AFNs



AFN concatenação de 2 AFNs

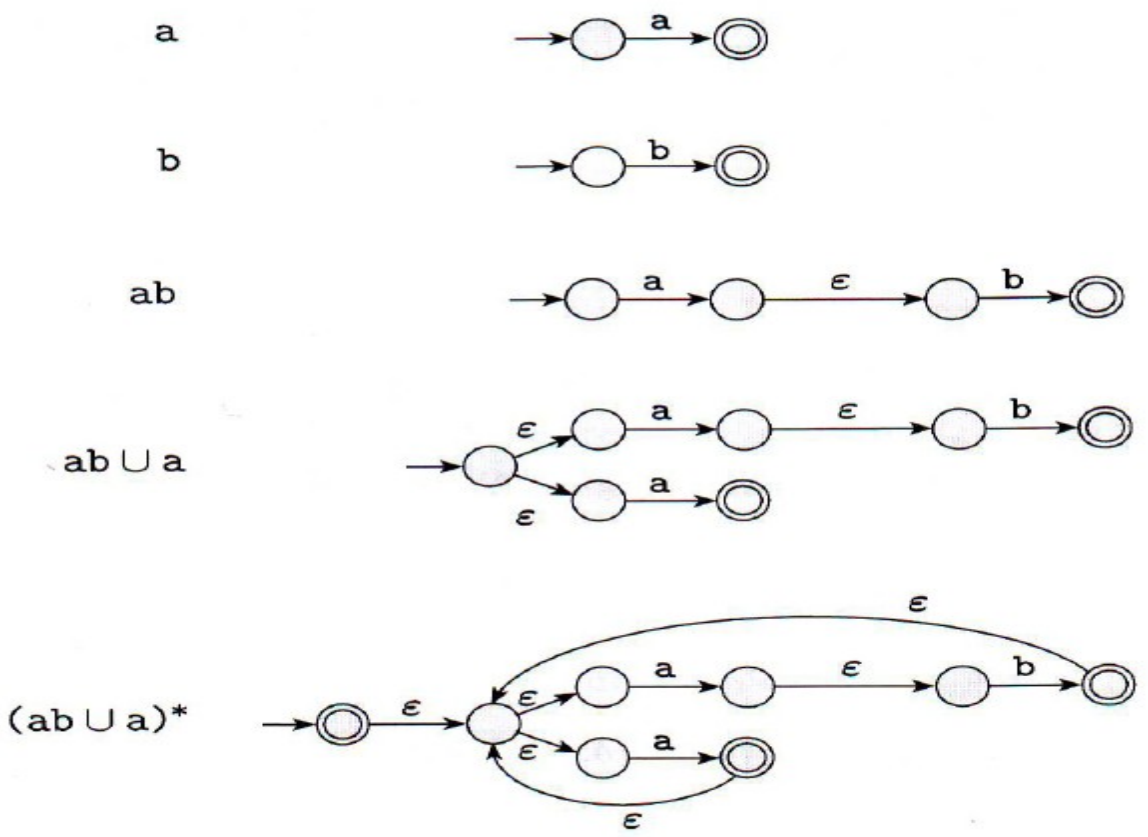


AFN estrela de outro AFN



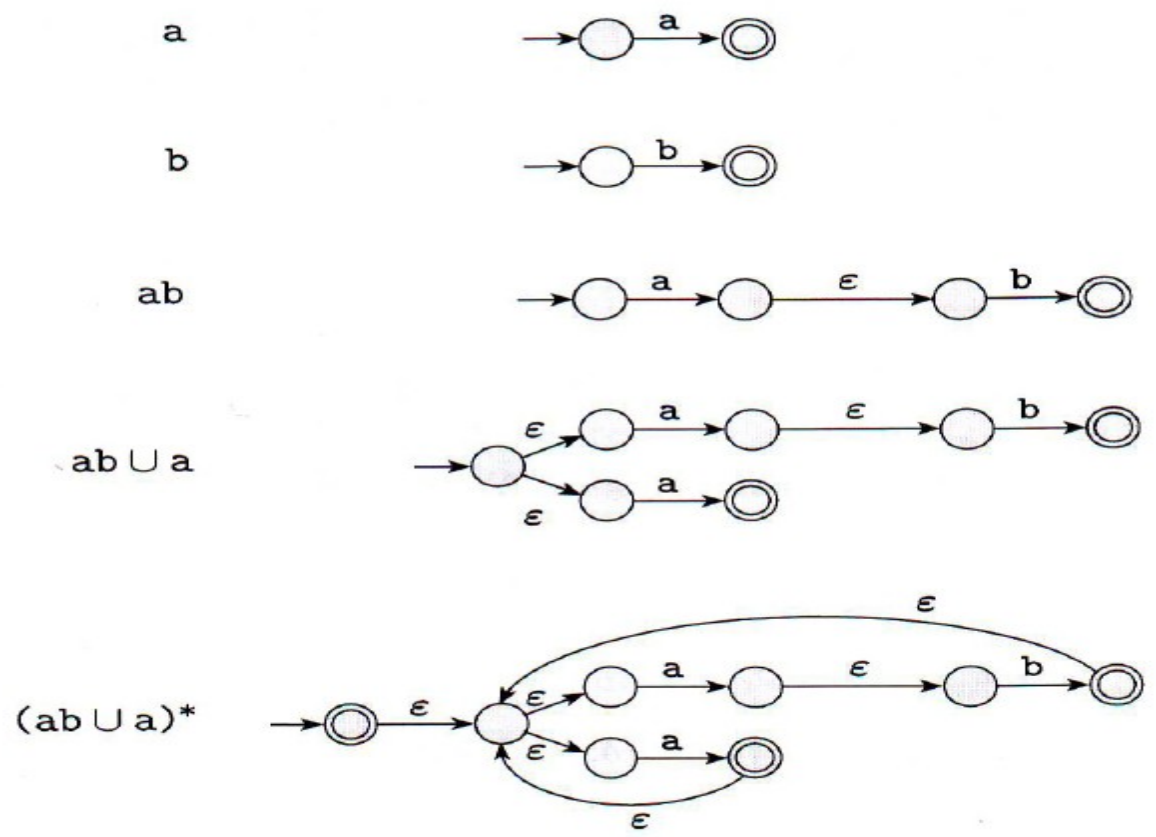
Equivalência de ERs e AFs – Parte 1

- Observação: essa prova fornece um mecanismo/algoritmo para construção de AFNs a partir de ERs.
- Ex: (ab U a)*
-
-



Equivalência de ERs e AFs – Parte 1

- Observação: essa prova fornece um mecanismo/algoritmo para construção de AFNs a partir de ERs.
- Ex: (ab U a)*
- O AFN resultante não necessariamente possui o número mínimo de estados
- Como seria o AFN com apenas 2 estados para essa expressão?



Qual a relação entre expressões regulares e linguagens regulares

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

- Para isso, o que eu preciso mostrar?
  - Que eu consigo converter um AFD ou AFN em uma expressão regular (ER) equivalente
  - E que eu consigo converter uma ER em um AFD/AFN

Qual a relação entre expressões regulares e linguagens regulares

TEOREMA 1.54

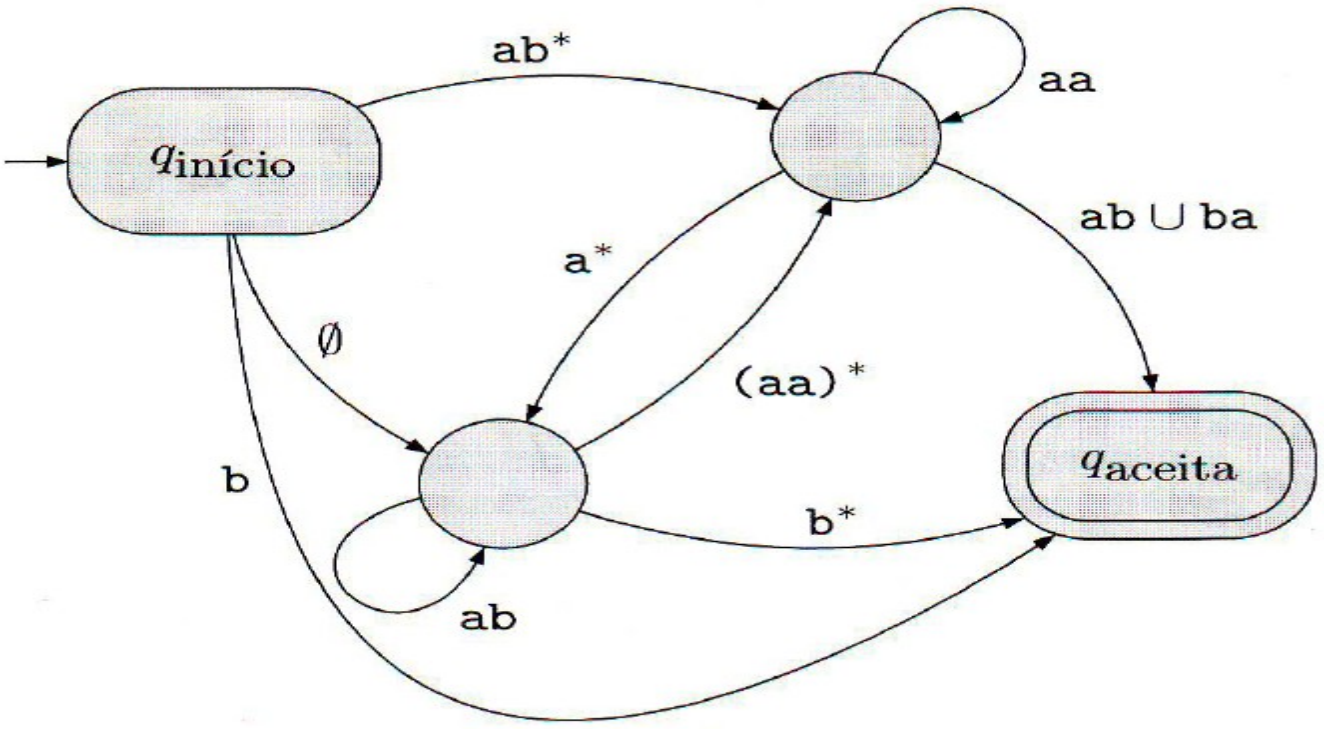
Uma linguagem é regular se e somente se alguma expressão regular a descreve.

- Para isso, o que eu preciso mostrar?
  - Que eu consigo converter um AFD ou AFN em uma expressão regular (ER) equivalente
  - E que eu consigo converter uma ER em um AFD/AFN

Para essa conversão vamos precisar de um terceiro tipo de autômato...

Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

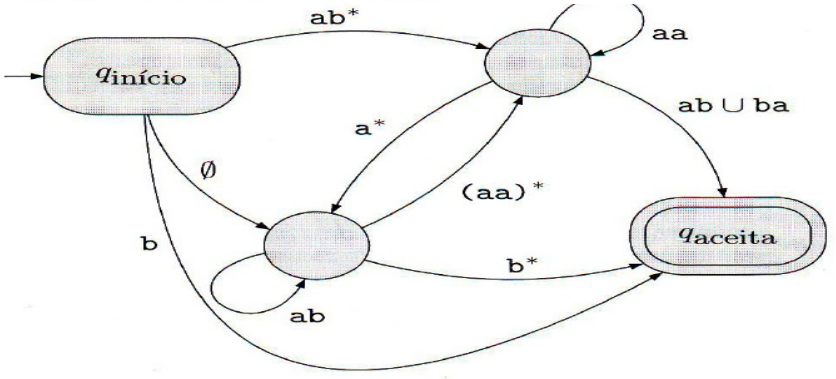
Rótulos das arestas são expressões regulares



Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

Por conveniência, requeremos que os AFNGs tenham sempre um formato especial que atenda às seguintes condições: (facilitará a definição da função de transição)

- O estado inicial tem setas de transição saindo para todos os outros estados, mas nenhuma seta chegando de qualquer outro estado.
- Existe apenas um estado de aceitação, e ele tem setas chegando de todos os outros estados, mas nenhuma seta saindo para qualquer outro estado. Além disso, o estado de aceitação não é o mesmo que o estado inicial.
- Com exceção dos estados inicial e de aceitação, uma seta sai de cada estado para todos os outros e também de cada estado para ele mesmo.

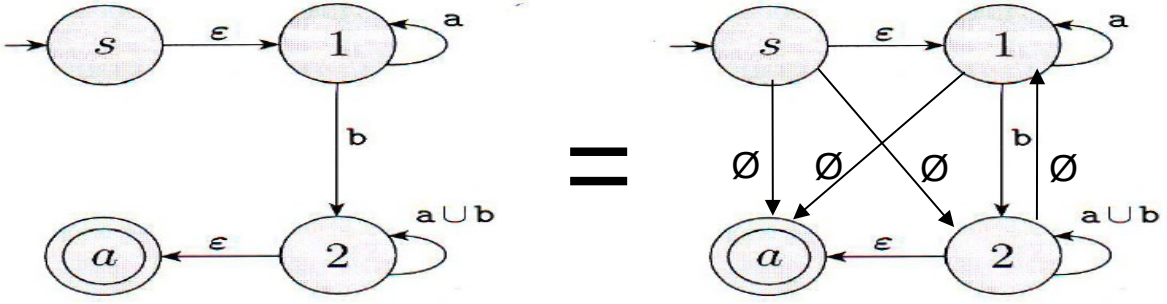


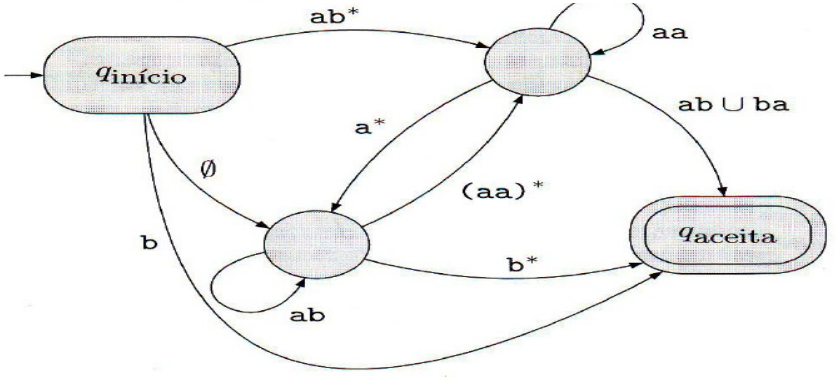
Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

Por conveniência, requeremos que os AFNGs tenham sempre um formato especial que atenda às seguintes condições: (facilitará a definição da função de transição)

- O estado inicial tem setas de transição saindo para todos os outros estados, mas nenhuma seta chegando de qualquer outro estado.
- Existe apenas um estado de aceitação, e ele tem setas chegando de todos os outros estados, mas nenhuma seta saindo para qualquer outro estado. Além disso, o estado de aceitação não é o mesmo que o estado inicial.
- Com exceção dos estados inicial e de aceitação, uma seta sai de cada estado para todos os outros e também de cada estado para ele mesmo.

Por simplicidade podemos omitir arestas com Ø, mas elas estão lá...





Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

DEFINIÇÃO 1.64

Um ***autômato finito não-determinístico generalizado*** é uma 5-upla, $(Q, \Sigma, \delta, q_{\text{início}}, q_{\text{aceita}})$, onde

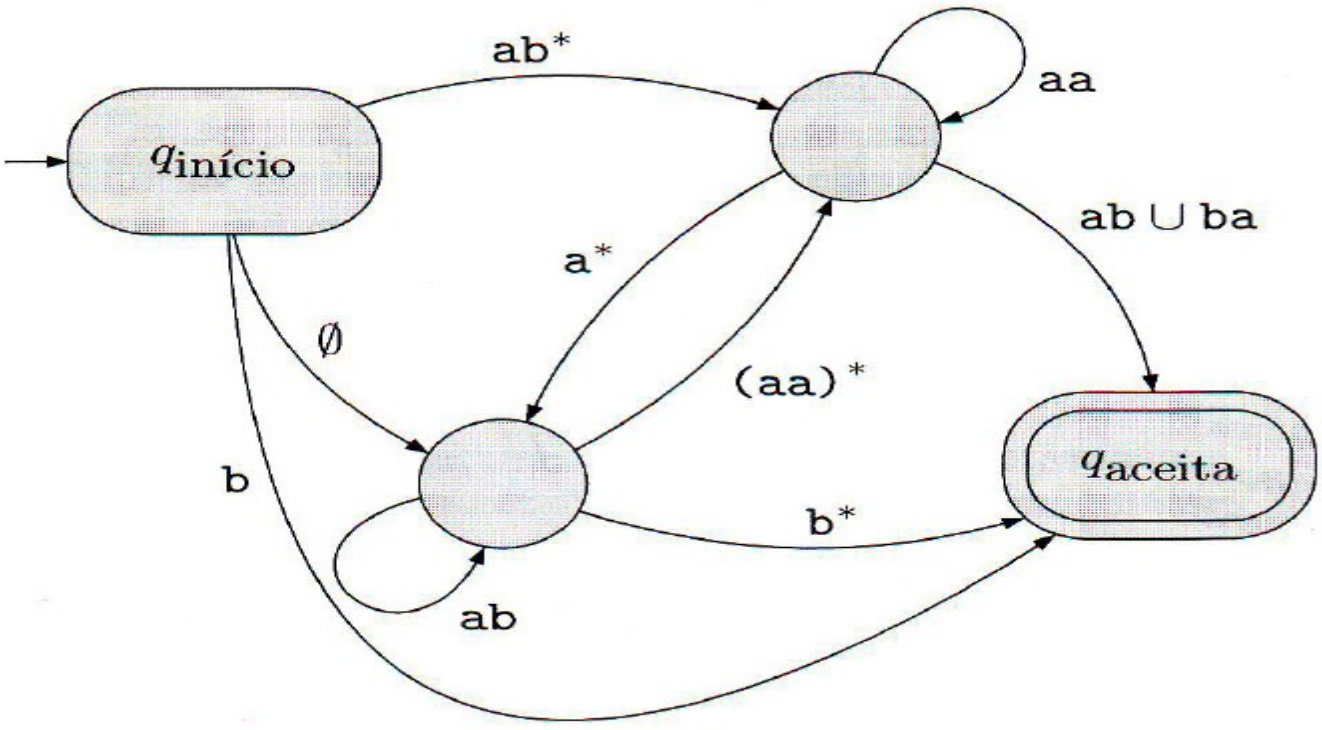
1. Q é o conjunto finito de estados,
2. Σ é o alfabeto de entrada,
3. $\delta: (Q - \{q_{\text{aceita}}\}) \times (Q - \{q_{\text{início}}\}) \longrightarrow \mathcal{R}$ é a função de transição,
4. $q_{\text{início}}$ é o estado inicial, e
5. q_{aceita} é o estado de aceitação.

Conjunto de todas as expressões regulares possíveis sobre o alfabeto

Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

Será que essas cadeias são aceitas pelo AFNG abaixo:

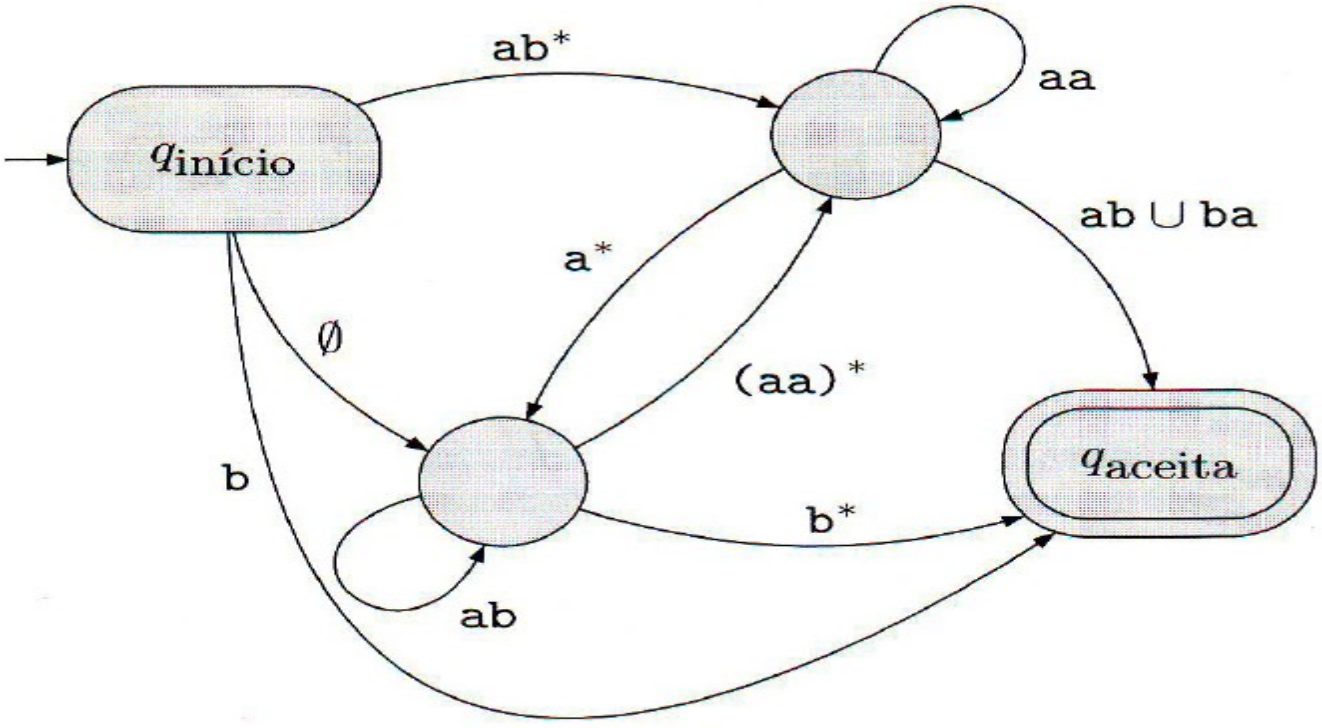
- ab
- aab
- b
- abbabaaaaba
- bb



Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

Será que essas cadeias são aceitas pelo AFNG abaixo:

- ab
- aab
- b
- abbabaaaaba
- bb



Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

Um AFNG aceita uma cadeia w em Σ^* se $w = w_1 w_2 \cdots w_k$, onde cada w_i está em Σ^*, e existe uma seqüência de estados $q_0, q_1, \ldots, q_k$ tal que

Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

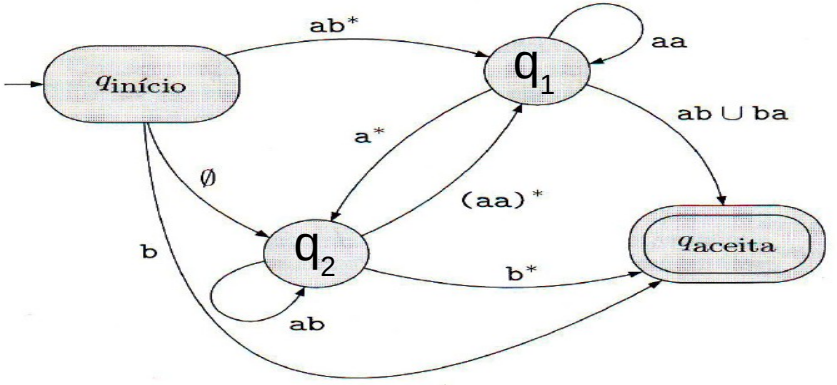
Um AFNG aceita uma cadeia w em Σ^* se $w = w_1 w_2 \cdots w_k$, onde cada w_i está em Σ^*, e existe uma seqüência de estados $q_0, q_1, \ldots, q_k$ tal que

1. $q_0 = q_{\text{início}}$ é o estado inicial,
2. $q_k = q_{\text{aceita}}$ é o estado de aceitação, e
3. para cada i, temos $w_i \in L(R_i)$, onde $R_i = \delta(q_{i-1}, q_i)$; em outras palavras, R_i é a expressão sobre a seta de q_{i-1} a q_i.

Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

Um AFNG aceita uma cadeia w em Σ^* se $w = w_1 w_2 \cdots w_k$, onde cada w_i está em Σ^*, e existe uma seqüência de estados $q_0, q_1, \ldots, q_k$ tal que

1. $q_0 = q_{\text{início}}$ é o estado inicial,
2. $q_k = q_{\text{aceita}}$ é o estado de aceitação, e
3. para cada i, temos $w_i \in L(R_i)$, onde $R_i = \delta(q_{i-1}, q_i)$; em outras palavras, R_i é a expressão sobre a seta de q_{i-1} a q_i.



Ex: abbbaaababbb ?

$q_{início} \rightarrow q_1 \rightarrow q_1 \rightarrow q_2 \rightarrow q_2 \rightarrow q_2 \rightarrow q_{aceita}$

Equivalência de ERs e AFs

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Prova:

Parte 1: (<=)

LEMA 1.55

Se uma linguagem é descrita por uma expressão regular, então ela é regular.

Prova parte 1: vamos construir um AFN que reconheça L(R), e portanto L(R) será regular (ou seja, vamos mostrar como converter uma ER em um AFN)

Parte 2: (=>)

LEMA 1.60

Se uma linguagem é regular, então ela é descrita por uma expressão regular.

Prova parte 2: se L é regular então um existe um AFD que a descreve. Então vamos mostrar que é sempre possível (e mostraremos como) converter um AFD em uma ER equivalente.

Equivalência de ERs e AFs – Parte 2

- Vamos:

  1) mostrar como converter AFDs em AFNGs

  2) mostrar como converter AFNGs em ERs

Equivalência de ERs e AFs – Parte 2

- Vamos:

  1) mostrar como converter AFDs em AFNGs

  2) mostrar como converter AFNGs em ERs

Conversão de AFD em AFNG

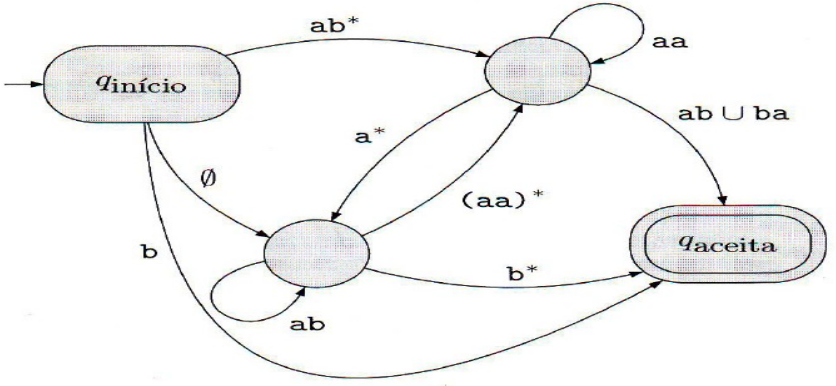
- O que eu preciso fazer no AFD para convertê-lo em AFNG?
- Lembrando...

Autômatos Finitos Não-Determinísticos Generalizados (AFNGs)

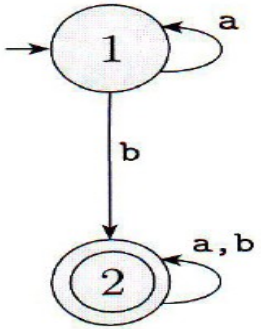
Por conveniência, requeremos que os AFNGs tenham sempre um formato especial que atenda às seguintes condições:

- O estado inicial tem setas de transição saindo para todos os outros estados, mas nenhuma seta chegando de qualquer outro estado.
- Existe apenas um estado de aceitação, e ele tem setas chegando de todos os outros estados, mas nenhuma seta saindo para qualquer outro estado. Além disso, o estado de aceitação não é o mesmo que o estado inicial.
- Com exceção dos estados inicial e de aceitação, uma seta sai de cada estado para todos os outros e também de cada estado para ele mesmo.

$\delta: (Q - \{q_{\text{aceita}}\}) \times (Q - \{q_{\text{inicio}}\}) \longrightarrow \mathcal{R}$
function,

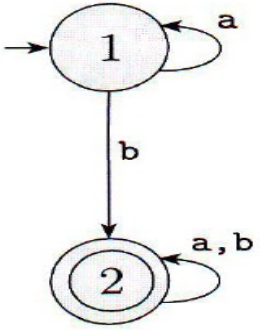


Exemplo (AFD → AFNG)

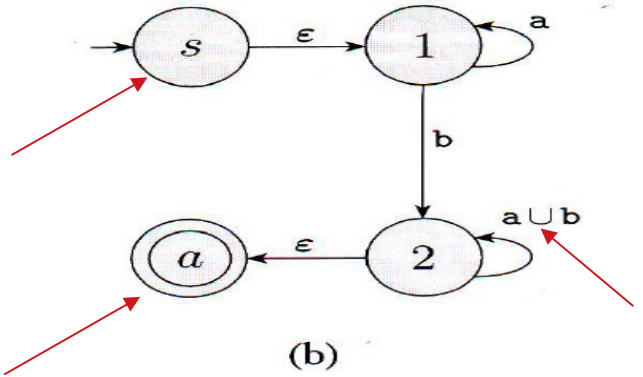


(a)

Exemplo (AFD → AFNG)

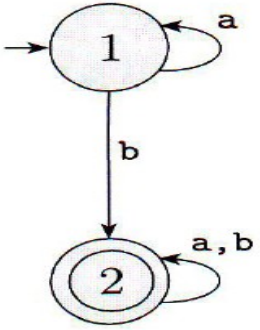


(a)

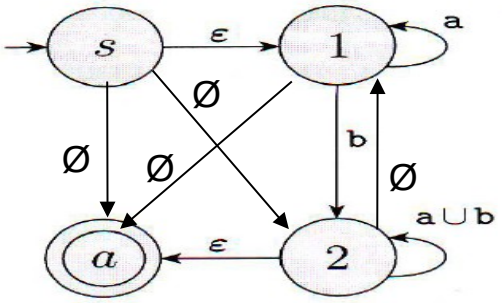


(b)

Exemplo (AFD → AFNG)

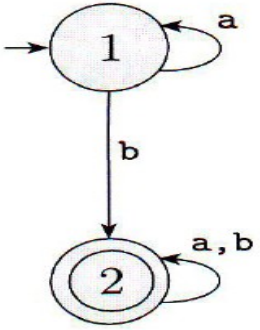


(a)

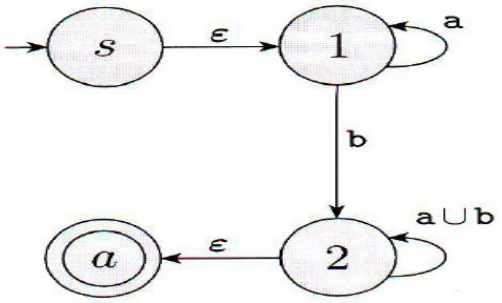


(b)

Exemplo (AFD → AFNG)



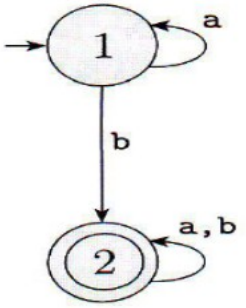
(a)

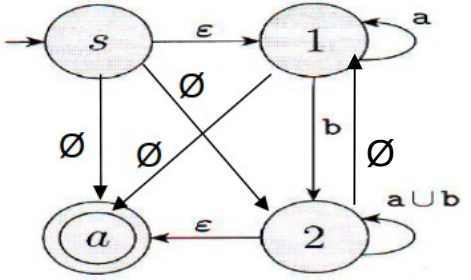


(b)

Conversão de AFD em AFNG

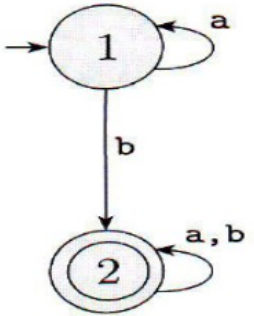
- Novo estado inicial apontando para o antigo com uma seta ε
- Novo estado final com setas ε chegando dos estados finais antigos (que deixam de ser finais)
- Setas com múltiplos rótulos (ou múltiplas setas entre 2 estados na mesma direção) viram uma seta com a união dos rótulos
- Setas com rótulo Ø onde não havia setas (e deveria ter no AFNG)

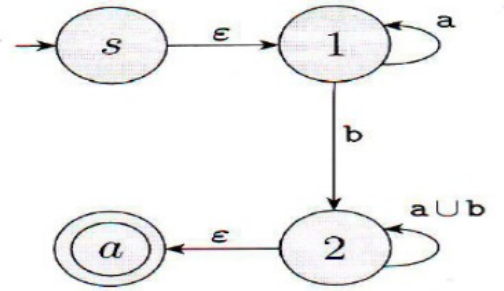




Conversão de AFD em AFNG

- Novo estado inicial apontando para o antigo com uma seta ε
- Novo estado final com setas ε chegando dos estados finais antigos (que deixam de ser finais)
- Setas com múltiplos rótulos (ou múltiplas setas entre 2 estados na mesma direção) viram uma seta com a união dos rótulos
- Setas com rótulo Ø onde não havia setas (e deveria ter no AFNG)





Equivalência de ERs e AFs – Parte 2

- Vamos:

  1) mostrar como converter AFDs em AFNGs

  2) mostrar como converter AFNGs em ERs

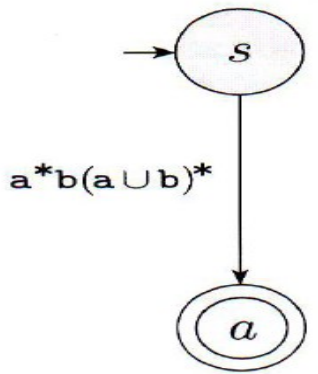
Equivalência de ERs e AFs – Parte 2

Conversão de AFNG em ER

- O AFNG tem k estados, k >= 2 (estados inicial e de aceitação são distintos)

Conversão de AFNG em ER

- O AFNG tem k estados, k >= 2 (estados inicial e de aceitação são distintos)
- Se k = 2, há só uma aresta contendo a expressão regular que descreve a linguagem reconhecida pelo AFNG



Equivalência de ERs e AFs – Parte 2

Conversão de AFNG em ER

- O AFNG tem k estados, k >= 2 (estados inicial e de aceitação são distintos)
- Se k = 2, há só uma aresta contendo a expressão regular que descreve a linguagem reconhecida pelo AFNG
- Se k > 2, construímos um AFNG **equivalente** com k-1 estados
- Continuo o processo até que k =2

Equivalência de ERs e AFs – Parte 2

Conversão de AFNG em ER

- O AFNG tem k estados, k >= 2 (estados inicial e de aceitação são distintos)
- Se k = 2, há só uma aresta contendo a expressão regular que descreve a linguagem reconhecida pelo AFNG
- Se k > 2, construímos um AFNG **equivalente** com k-1 estados
- Continuo o processo até que k =2

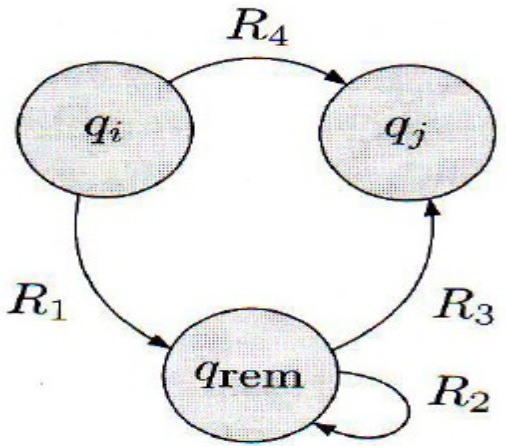
Equivalência de ERs e AFs – Parte 2

Conversão de AFNG em ER

- Construindo um AFNG equivalente com k-1 estados:
  - Escolha o estado a ser removido – chamaremos de q_{rem} (estado não inicial e não final)
  - Remova q_{rem}
  - Ajuste a expressão de CADA transição $q_i \rightarrow q_j$ (i e j de um caminho (q_i, q_{rem}, q_j) que existia) de forma a compensar a remoção de q_{rem} (para todo i e j, mesmo que i=j)

Equivalência de ERs e AFs – Parte 2
Conversão de AFNG em ER

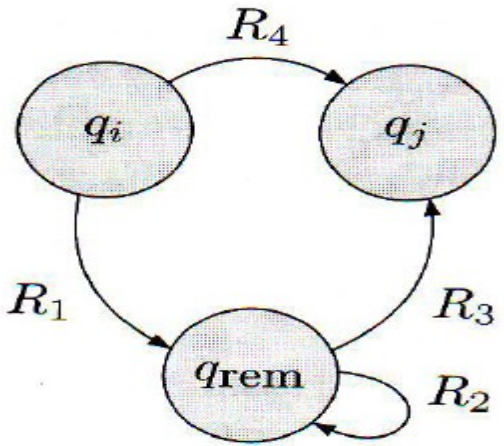
- Ex:



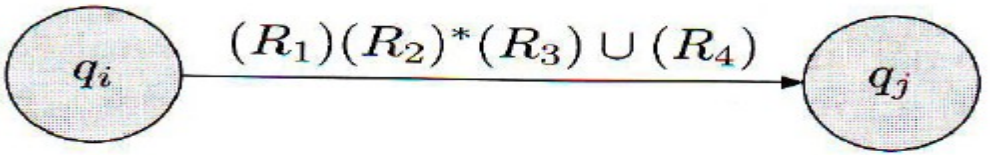
antes

Conversão de AFNG em ER

- Ex:



antes



depois

Conversão de AFNG em ER

Também para $q_i \leftarrow q_j$ e quando i = j

E lembre-se que quando não aparecer uma aresta, na verdade há a transição com R = Ø

Equivalência de ERs e AFs – Parte 2

Conversão de AFNG em ER

CONVERT(G):

1. Seja k o número de estados de G.
2. Se $k = 2$, então G deve consistir de um estado inicial, um estado de aceitação, e uma única seta conectando os dois rotulada com uma expressão regular R.
   Retorne a expressão R.
3. Se $k > 2$, selecionamos qualquer $q_{\text{rem}} \in Q$ diferente de $q_{\text{início}}$ e de q_{aceita} e seja G' o AFNG $(Q', \Sigma, \delta', q_{\text{início}}, q_{\text{aceita}})$, onde

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

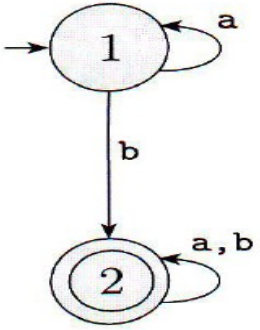
   e para qualquer $q_i \in Q' - \{q_{\text{aceita}}\}$ e qualquer $q_j \in Q' - \{q_{\text{início}}\}$ seja

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

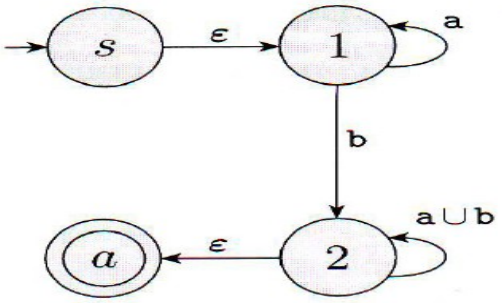
   para $R_1 = \delta(q_i, q_{\text{rem}})$, $R_2 = \delta(q_{\text{rem}}, q_{\text{rem}})$, $R_3 = \delta(q_{\text{rem}}, q_j)$ e $R_4 = \delta(q_i, q_j)$.

4. Compute CONVERT(G') e retorne esse valor.

Exemplo (AFD → ER)

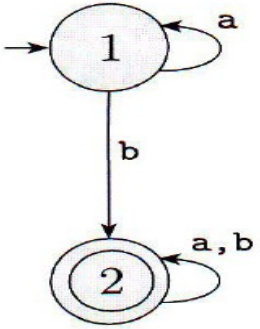


(a)

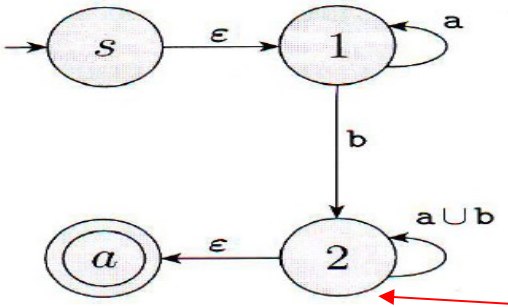


(b)

Exemplo (AFD → ER)



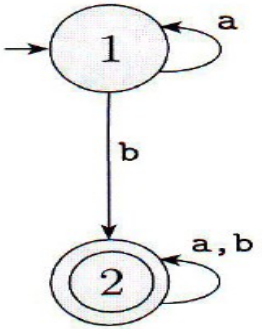
(a)



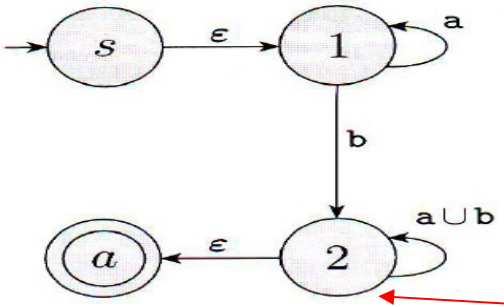
(b)

qrem

Exemplo (AFD → ER)

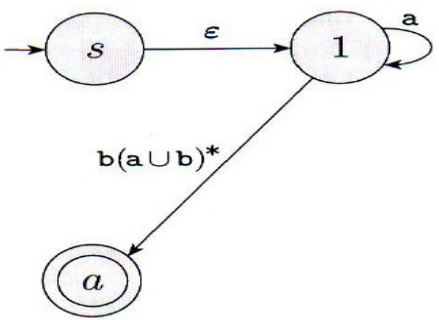


(a)



qrem

(b)

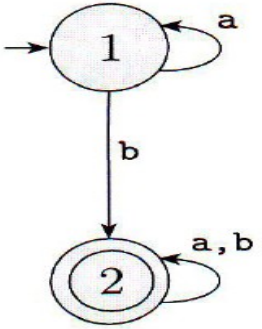


(c)

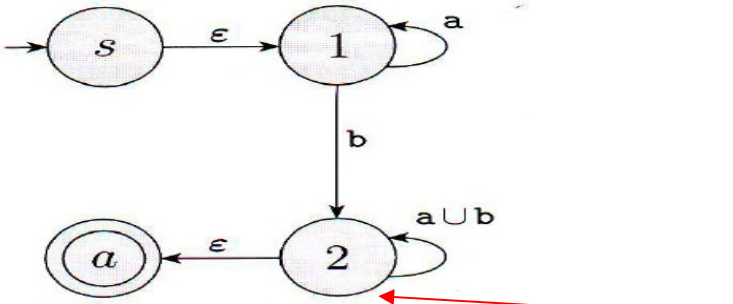
Ajustando as transições que saem de s:

- Caminho (s, 2, 1) não existia, então δ(s,1) continua igual
- Caminho (s, 2, a) não existia, então δ(s,a) continua igual

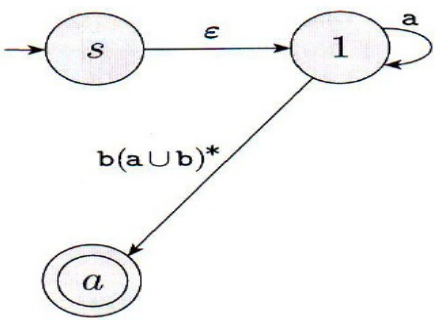
Exemplo (AFD → ER)



(a)



(b)

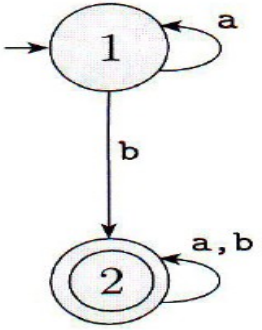


(c)

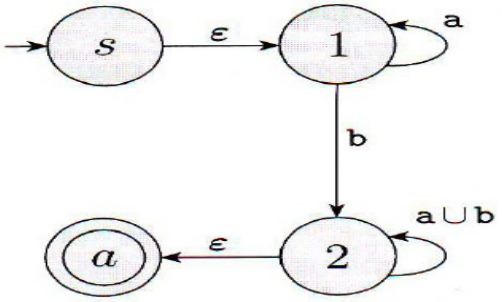
Ajustando as transições que saem de 1:

- Caminho (1, 2, 1) não existia, então δ(1,1) continua igual
- Caminho (1, 2, a) existia, então δ(1,a) muda!
  $\emptyset$ U b(aUb)*ε = b(aUb)*

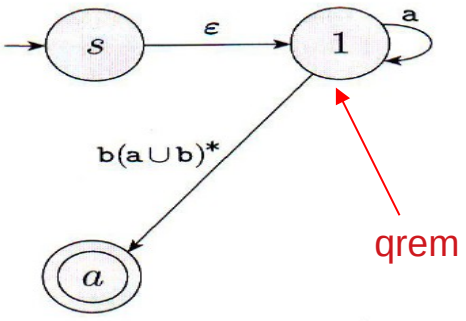
Exemplo (AFD → ER)



(a)

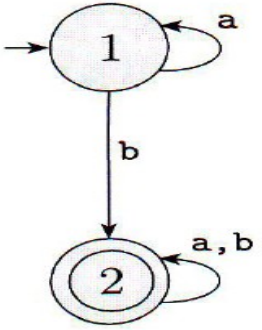


(b)

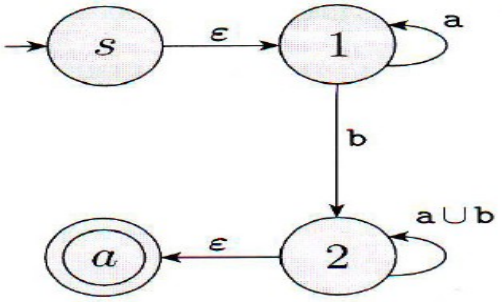


(c)

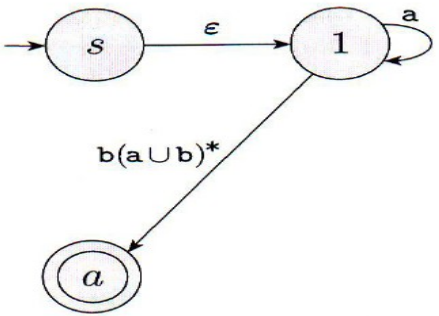
Exemplo (AFD → ER)



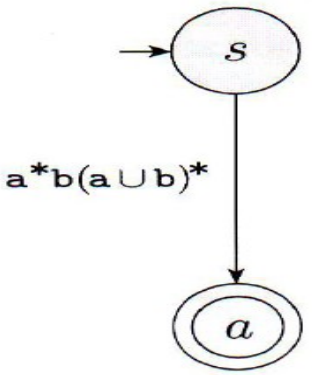
(a)



(b)



(c)



(d)

AFIRMATIVA 1.65

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Base**: se k = 2

Base: Prove que a afirmação é verdadeira para $k = 2$ estados. Se G tem apenas dois estados, ele só pode ter uma única seta, que vai do estado inicial para o estado de aceitação. A expressão regular que é o rótulo sobre essa seta descreve todas as cadeias que propiciam a G chegar ao estado de aceitação. Logo, essa expressão é equivalente a G.

AFIRMATIVA **1.65**

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados

Ex: se a ER retornada pela função CONVERT(G) é equivalente a G quando G tem apenas 2 estados, será também equivalente quando G tem 3 estados?

AFIRMATIVA 1.65

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados

Ou seja, vamos mostrar que cada chamada da recursão produz um autômato equivalente a G

CONVERT(G):

1. Seja k o número de estados de G.
2. Se $k = 2$, então G deve consistir de um estado inicial, um estado de aceitação, e uma única seta conectando os dois rotulada com uma expressão regular R.
   Retorne a expressão R.
3. Se $k > 2$, selecionamos qualquer $q_{\text{rem}} \in Q$ diferente de $q_{\text{início}}$ e de q_{aceita} e seja G' o AFNG $(Q', \Sigma, \delta', q_{\text{início}}, q_{\text{aceita}})$, onde

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

e para qualquer $q_i \in Q' - \{q_{\text{aceita}}\}$ e qualquer $q_j \in Q' - \{q_{\text{início}}\}$ seja

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

para $R_1 = \delta(q_i, q_{\text{rem}})$, $R_2 = \delta(q_{\text{rem}}, q_{\text{rem}})$, $R_3 = \delta(q_{\text{rem}}, q_j)$ e $R_4 = \delta(q_i, q_j)$.

4. Compute CONVERT(G') e retorne esse valor.

AFIRMATIVA **1.65**

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados
    - Primeiro: provamos que, se G reconhece uma cadeia w, G' (G - qrem) também reconhece (<=)
    - Segundo: provamos que, se G' reconhece uma cadeia w, G também reconhece (=>)

AFIRMATIVA **1.65**

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados
    - Primeiro: provamos que, se G reconhece uma cadeia w, G' (G - qrem) também reconhece (<=)
    - Segundo: provamos que, se G' reconhece uma cadeia w, G também reconhece (=>)

AFIRMATIVA **1.65**

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados (vou remover um estado)
    - Primeiro: provamos que, se G reconhece uma cadeia w, G' (G - qrem) também reconhece (<=)

G reconhece uma cadeia w => existe pelo menos um caminho por G

$q_{início}, q_1, q_2, ..., q_{aceita}$

Se nenhum desses estados é q_{rem}, G' aceita w (pois as expressões antigas (de G) rotulando cada transição desse caminho estão contidas nas novas expressões de G', como parte da união)

Se o caminho contém q_{rem}, cada retirada de q_{rem}'s consecutivos não altera o fato de G' aceitar w, pois q_i/q_j anterior/posterior a essa série q_{rem}'s possuem em G' uma transição q_i->q_j com uma expressão que descreve cadeias que levam de q_i a q_j via q_{rem}

Logo, G' também reconhece w

AFIRMATIVA **1.65**

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados
    - Primeiro: provamos que, se G reconhece uma cadeia w, G' (G - qrem) também reconhece (<=)
    - Segundo: provamos que, se G' reconhece uma cadeia w, G também reconhece (=>)

Cada transição de G' q_i->q_j descreve cadeias que, em G, vão de q_i a q_j diretamemente ou via q_{rem}

Logo, G reconhece w.

AFIRMATIVA **1.65**

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

- Prova: por indução sobre k (número de estados de G)
  - **Indução**: vale para k-1 estados. Vamos provar que vale para k estados
    - Primeiro: provamos que, se G reconhece uma cadeia w, G' (G - qrem) também reconhece (<=)
    - Segundo: provamos que, se G' reconhece uma cadeia w, G também reconhece (=>)

Cada transição de G' q_i->q_j descreve cadeias que, em G, vão de q_i a q_j diretamemente ou via q_{rem}

Logo, G reconhece w.

LOGO...

AFIRMATIVA 1.65

Para qualquer AFNG G, CONVERT(G) é equivalente a G.

LEMA 1.60

Se uma linguagem é regular, então ela é descrita por uma expressão regular.

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Equivalência de ERs e AFs

TEOREMA 1.54

Uma linguagem é regular se e somente se alguma expressão regular a descreve.

Prova:

Parte 1: (<=)

LEMA 1.55

Se uma linguagem é descrita por uma expressão regular, então ela é regular.

Prova parte 1: vamos construir um AFN que reconheça L(R), e portanto L(R) será regular (ou seja, vamos mostrar como converter uma ER em um AFN)

Parte 2: (=>)

LEMA 1.60

Se uma linguagem é regular, então ela é descrita por uma expressão regular.

Prova parte 2: se L é regular então um existe um AFD que a descreve. Então vamos mostrar que é sempre possível (e mostraremos como) converter um AFD em uma ER equivalente.

Exercícios Sipser (2ª edição)

Exercícios de 1.1 a 1.28

ACH2043
INTRODUÇÃO À TEORIA DA COMPUTAÇÃO

Aula 7

Expressões Regulares

Profa. Ariane Machado Lima
ariane.machado@usp.br