SBVLIFA: Linguagens Formais e Autômatos

Aula 05: Expressões Regulares

Bacharelado em Ciência da Computação
Prof. Dr. David Buzatto

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Linguagens Regulares

- Linguagens Regulares

Tipo	Classe de Linguagens	Modelo de Gramática	Modelo de Reconhecedor
0	Recursivamente enumeráveis	Irrestrita	Máquina de Turing
1	Sensíveis ao contexto	Sensível ao contexto	Máquina de Turing com fita limitada
2	Livres de contexto	Livre de contexto	Autômato de pilha
3	**Regulares**	**Linear (direita ou esquerda)**	**Autômato finito**

Expressões Regulares

- As expressões regulares são uma forma de descrição algébrica para linguagens regulares;
- Elas têm a mesma capacidade de definição dos autômatos finitos, ou seja, elas são também capazes de definirem linguagens regulares;
- Ao passo que nos autômatos finitos a descrição da linguagem é feita través de algo semelhante à uma máquina, nas expressões regulares a descrição é feita de forma declarativa, expressando como são as strings que queremos aceitar;
- Dada essa característica, as expressões regulares são empregadas largamente em sistemas que processam strings:
  - Comandos de pesquisa como o `grep` do UNIX;
  - Geradores de analisadores léxicos;
  - Reconhecedores de padrões em APIs de linguagens de programação etc.

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

Texto cortesia do Espetacular Gerador de Lero Lero

https://lerolero.com/

- **Exemplo em Java:**

```
public static void main( String[] args ) {

    String texto = "Desde ontem a noite o último \n" +
                   "pull request desse SCRUM superou \n" +
                   "o desempenho do polimorfismo \n" +
                   "aplicado nas classes.";

    texto = texto.replaceAll( "\\b[Dd]\\S*", "!" );
    System.out.println( texto );

}
```

```
SAÍDA:
! ontem a noite o último
pull request ! SCRUM superou
o ! ! polimorfismo
aplicado nas classes.
```

Expressões Regulares

Exemplo em Java:

```
public static void main( String[] args ) {

    String texto = "Desde ontem a noite o último
                   "pull request desse SCRUM s
                   "o desempenho do polimorf    mo
                   "aplicado nas classes.";

    texto = texto.replaceAll( "\\b[Dd]\\S*", "!" );
    System.out.println( texto );

}
```

\b: fronteira de palavra
[Dd]: D ou d
\S: qualquer coisa que não seja espaço em branco
*: zero ou mais vezes

```
SAÍDA:
! ontem a noite o último
pull request ! SCRUM superou
o ! ! polimorfismo
aplicado nas classes.
```

Expressões Regulares

Exemplo

- A expressão regular:

$$01^{*} + 10^{*}$$

- Denota a linguagem que consiste em todas as strings que começam com 0 e têm qualquer número de 1's ou começam em 1 e tem qualquer número de 0's

Expressões Regulares
Operadores

- Os operadores das expressões regulares representam três operações sobre linguagens:
  - União (operador $+$): Sejam L e M duas linguagens e se $L = \{01, 10, 101\}$ e $M = \{\varepsilon, 101, 111\}$, $L \cup M = \{\varepsilon, 01, 10, 101, 111\}$
  - Concatenação (operador $\cdot$ (ou nada)): Sejam L e M duas linguagens e se $L = \{01, 10\}$ e $M = \{\varepsilon, 11\}$, $L.M = \{01, 10, 0111, 1011\}$
  - Fechamento (ou estrela, ou fechamento de Kleene, operador $*$):

    Seja L uma linguagem e se $L = \{0, 11\}$,

$$L^* = \bigcup_{i \geq 0} L^i = L^0 \cup L^1 \cup L^2 \ldots = \{\varepsilon\} \cup L \cup LL \ldots = \{\varepsilon\} \cup \{0, 11\} \cup \{0, 11\}\{0, 11\} \ldots$$

$$\{\varepsilon\} \cup \{0, 11\} \cup \{00, 011, 110, 1111\} \ldots =$$

$$\{\varepsilon, 0, 11, 00, 011, 110, 1111\} \ldots$$


Stephen Cole Kleene

Expressões Regulares

Construção de Expressões Regulares

- Escrever a expressão regular para o conjunto de strings que consistem em 0's e 1's alternados:

Expressões Regulares

Construção de Expressões Regulares

- Escrever a expressão regular para o conjunto de strings que consistem em 0's e 1's alternados:

$$(01)^* + (10)^* + 0(10)^* + 1(01)^*$$

ou

$$(\varepsilon + 1)\,(01)^*\,(\varepsilon + 0)$$

Expressões Regulares

Precedência de Operadores

Agrupamento com Parênteses > Estrela > Concatenação > União

$$01^* + 1 = \big(0(1^*)\big) + 1$$

Expressões Regulares

Definição Formal

- R é uma expressão regular se R for:
  1. a para algum a no alfabeto Σ,
  2. ε,
  3. $\emptyset$,
  4. $(R_1 + R_2)$, onde R_1 e R_2 são expressões regulares,
  5. $(R_1 R_2)$, onde R_1 e R_2 são expressões regulares, ou
  6. R_1^*, onde R_1 é uma expressão regular.
- Nos itens 1 e 2, as expressões regulares a e ε representam as linguagens $\{a\}$ e $\{\varepsilon\}$, respectivamente. No item 3, a expressão regular $\emptyset$ representa a linguagem vazia. Nos itens 4, 5 e 6 as expressões representam as linguagens obtidas tomando-se a união ou concatenação das linguagens R_1 e R_2, ou o fechamento da linguagem R_1. Note que as definição é indutiva, pois R_1 e R_2 são menores que R.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

Leis Algébricas

- Associatividade e Comutatividade:
  - $L + M = M + L$: lei comutativa para união;
  - $(L + M) + N = L + (M + N)$: lei associativa para união;
  - $(LM)N = L(MN)$: lei associativa para concatenação;
  - $LM = ML$: cuidado! Essa afirmação é falsa! Não há lei comutativa para concatenação!
- Identidades e Aniquiladores:
  - $\emptyset + L = L + \emptyset = L$: Essa lei afirma que $\emptyset$ é a identidade para a união;
  - $\varepsilon L = L\varepsilon = L$: Essa lei afirma que ε é identidade para a concatenação;
  - $\emptyset L = L\emptyset = \emptyset$: Essa lei afirma que $\emptyset$ é o aniquilador para a concatenação;
- Leis Distributivas:
  - $L(M + N) = LM + LN$: lei distributiva à esquerda da concatenação sobre união;
  - $(M + N)L = ML + NL$: lei distributiva à direita da concatenação sobre a união.

Expressões Regulares

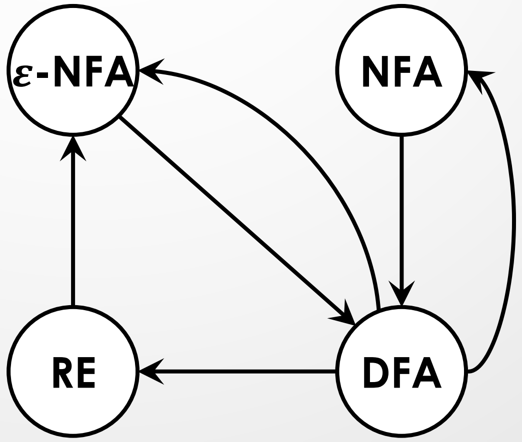
Leis Algébricas

- Lei de Idempotência:
  - Um operador é idempotente se, ao atuar sobre dois operandos iguais, resultar no operando;
  - $L + L = L$: lei da idempotência para a união;
- Leis Envolvendo Fechamentos:
  - $(L^*)^* = L^*$: fechar uma expressão já fechada não muda nada;
  - $\emptyset^* = \{\varepsilon\}$: o fechamento de $\emptyset$ contém apenas a string ε;
  - $\varepsilon^* = \varepsilon$: a concatenação de qualquer número de ε é ε;
  - $L^+ = LL^* = L(\varepsilon + L + LL + LLL + \cdots) = L\varepsilon + LL + LLL + LLLL + \cdots = L + LL + LLL + LLL + \cdots$
  - $L^+ = L^*L = (\varepsilon + L + LL + LLL + \cdots)L = \varepsilon L + LL + LLL + LLLL + \cdots = L + LL + LLL + LLL + \cdots$
  - $L^* = L^+ + \varepsilon$
  - $L? = \varepsilon + L$: definição do operador ?, que significa que o operando é opcional, ou seja, o resultado é vazio ou o operando

Expressões Regulares

Autômatos Finitos e Expressões Regulares

- Os autômatos finitos e as expressões regulares são capazes de representar exatamente o mesmo conjunto de linguagens, as chamadas linguagens regulares. Para provar que essa afirmação é verdadeira, deve-se mostrar que:
  - Toda linguagem definida por um autômato finito também é definida por uma expressão regular. Prova: supor que a linguagem é aceita por um DFA;
  - Toda linguagem definida por uma expressão regular também é definida por um autômato finito. Prova: mostrar que existe um ε-NFA que aceita a mesma linguagem.



Expressões Regulares

DFA para RE

Teorema: Se $L = L(A)$ para algum DFA A, então existe uma expressão regular R tal que $L = L(R)$

- **Prova:** $R_{ij}^{(k)}$ é uma expressão regular cuja linguagem é o conjunto de strings w tais que w é o rótulo de um caminho do estado i ao estado j em A, e esse caminho não tem nenhum nó intermediário cujo número seja maior que k.

Expressões Regulares

DFA para RE

- **Base:** A base é $k = 0$, o que implica que a restrição sobre os caminhos é que o caminho não deve ter absolutamente nenhum estado intermediário, existindo assim somente dois tipos de caminhos que satisfazem tal condição:
  1. Um arco/transição do nó/estado i para o nó j;
  2. Um caminho de comprimento 0 que consiste em apenas algum nó i.

  Se $i \neq j$, apenas o caso 1 é possível, então deve-se examinar o DFA A e encontrar os símbolos de entrada a tais que exista uma transição do estado i para o estado j para o símbolo a.

  a) Se não existe tal símbolo a, então $R_{ij}^{(0)} = \emptyset$;

  b) Se existe exatamente um símbolo a, então $R_{ij}^{(0)} = a$;

  c) Se existem símbolos $a_1, a_2, \dots, a_k$ que rotulam arcos do estado i ao estado j, então $R_{ij}^{(0)} = a_1 + a_2 + \cdots + a_k$

  Se $i = j$, os caminhos válidos são os caminhos de comprimento 0, representados pela expressão regular ε e os loops/laços de i até ele mesmo.

  a) Se não existe tal símbolo a, então $R_{ij}^{(0)} = \varepsilon$;

  b) Se existe exatamente um símbolo a, então $R_{ij}^{(0)} = \varepsilon + a$;

  c) Se existem símbolos $a_1, a_2, \dots, a_k$ que rotulam arcos do estado i ao estado j, então $R_{ij}^{(0)} = \varepsilon + a_1 + a_2 + \cdots + a_k$

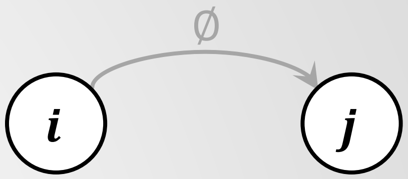
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

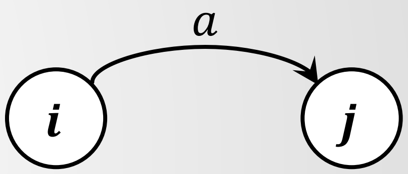
DFA para RE

- **Para $i \neq j$:**

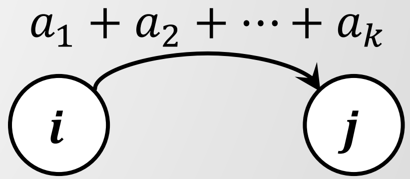
  a) Se não existe tal símbolo a, então $R_{ij}^{(0)} = \emptyset$;



  b) Se existe exatamente um símbolo a, então $R_{ij}^{(0)} = a$;



  c) Se existem símbolos $a_1, a_2, \dots, a_k$ que rotulam arcos do estado i ao estado j, então $R_{ij}^{(0)} = a_1 + a_2 + \cdots + a_k$



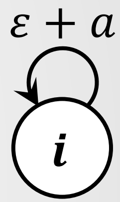
Expressões Regulares

DFA para RE

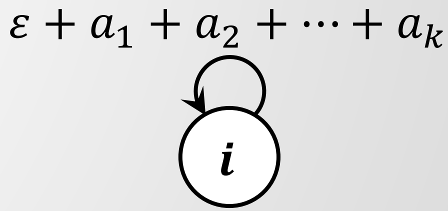
- **Para $i = j$:**

  a) Se não existe tal símbolo a, então $R_{ij}^{(0)} = \varepsilon$;

  b) Se existe exatamente um símbolo a, então $R_{ij}^{(0)} = \varepsilon + a$;



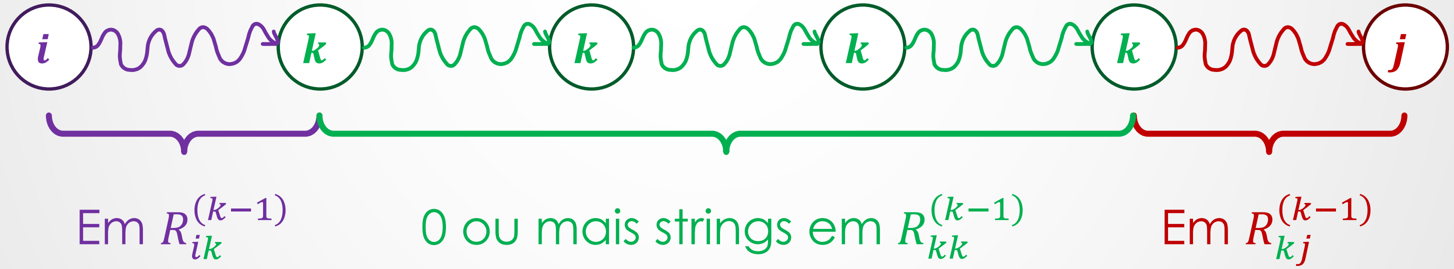
  c) Se existem símbolos $a_1, a_2, \dots, a_k$ que rotulam arcos do estado i ao estado j, então $R_{ij}^{(0)} = \varepsilon + a_1 + a_2 + \cdots + a_k$



Expressões Regulares

DFA para RE

- **Indução:** Suponha que exista um caminho do estado i para o estado j que não passe por nenhum estado mais alto que k. Há dois casos possíveis a considerar:
  1. O caminho não passa pelo estado k. Nesse caso, o rótulo do caminho está na linguagem de $R_{ij}^{(k-1)}$;
  2. O caminho passa pelo estado k pelo menos uma vez.



- Quando combinamos as expressões correspondentes aos dois caminhos anteriores, temos a expressão $R_{ij}^{(k)} = R_{ij}^{(k-1)} + R_{ik}^{(k-1)} \left(R_{kk}^{(k-1)}\right)^* R_{kj}^{(k-1)}$ para os rótulos de todos os caminhos, desde o estado i até o estado j, que não passam por nenhum estado mais alto que k.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

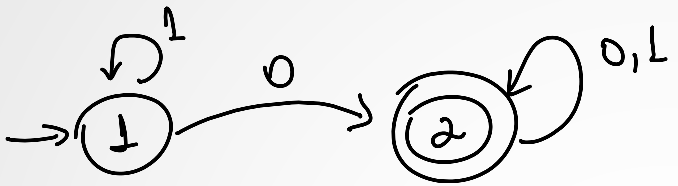
DFA para RE

- **Construção:** Efetuamos assim a construção dessas expressões em ordem crescente de sobrescrito, iniciando em $k = 0$ (base). Como cada $R_{ij}^{(k)}$ depende apenas de expressões com um sobrescrito menor, todas as expressões estarão disponíveis quando houver necessidade de sua utilização;
- Eventualmente, teremos $R_{ij}^{(n)}$ para todo i e j. Supondo que o estado 1 é o estado inicial, embora os estados de aceitação possam formar qualquer conjunto de estados. A expressão regular para a linguagem do autômato é então a soma/união de todas as expressões $R_{ij}^{(n)}$, tais que o estado i é o estado inicial e j é um estado de aceitação.

Expressões Regulares

DFA para RE - Exemplo

DFA:



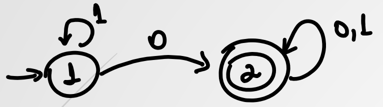
DFA que aceita todas as strings com pelo menos um zero.

Base: $k=0$

$R_{11}^{(0)}$	$\varepsilon + 1$
$R_{12}^{(0)}$	0
$R_{21}^{(0)}$	$\emptyset$
$R_{22}^{(0)}$	$\varepsilon + 0 + 1$

Expressões Regulares

DFA para RE - Exemplo

DFA: 

$k=0$

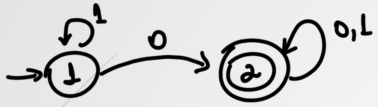
$R_{11}^{(0)}$	$\varepsilon+1$
$R_{12}^{(0)}$	0
$R_{21}^{(0)}$	$\emptyset$
$R_{22}^{(0)}$	$\varepsilon+0+1$

$k=1$ Aplicando $R_{ij}^{(1)} = R_{ij}^{(0)} + R_{i1}^{(0)}\left(R_{11}^{(0)}\right)^* R_{1j}^{(0)}$

	substituição direta	simplificação
$R_{11}^{(1)}$	$\varepsilon+1 + (\varepsilon+1)(\varepsilon+1)^*(\varepsilon+1)$	1^*
$R_{12}^{(1)}$	$0 + (\varepsilon+1)(\varepsilon+1)^*0$	1^*0
$R_{21}^{(1)}$	$\emptyset + \emptyset(\varepsilon+1)^*(\varepsilon+1)$	$\emptyset$
$R_{22}^{(1)}$	$\varepsilon+0+1 + \emptyset(\varepsilon+1)^*0$	$\varepsilon+0+1$

Expressões Regulares

DFA para RE - Exemplo

DFA: 

k=0

$R_{11}^{(0)}$	$\varepsilon+1$
$R_{12}^{(0)}$	0
$R_{21}^{(0)}$	$\emptyset$
$R_{22}^{(0)}$	$\varepsilon+0+1$

k=1

$R_{11}^{(1)}$	1^*
$R_{12}^{(1)}$	1^*0
$R_{21}^{(1)}$	$\emptyset$
$R_{22}^{(1)}$	$\varepsilon+0+1$

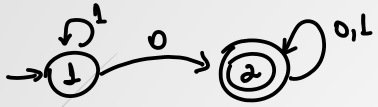
k=2 Aplicando $R_{ij}^{(2)} = R_{ij}^{(1)} + R_{i2}^{(1)}\left(R_{22}^{(1)}\right)^* R_{2j}^{(1)}$

	substituição direta	simplificação
$R_{11}^{(2)}$	$1^* + 1^*0(\varepsilon+0+1)^*\emptyset$	1^*
$R_{12}^{(2)}$	$1^*0 + 1^*0(\varepsilon+0+1)^*(\varepsilon+0+1)$	$1^*0(0+1)^*$
$R_{21}^{(2)}$	$\emptyset + (\varepsilon+0+1)(\varepsilon+0+1)^*\emptyset$	$\emptyset$
$R_{22}^{(2)}$	$\varepsilon+0+1 + (\varepsilon+0+1)(\varepsilon+0+1)^*(\varepsilon+0+1)$	$(0+1)^*$

Expressões Regulares

DFA para RE - Exemplo

DFA:



$k=0$

$R_{11}^{(0)}$	$\varepsilon+1$
$R_{12}^{(0)}$	0
$R_{21}^{(0)}$	$\emptyset$
$R_{22}^{(0)}$	$\varepsilon+0+1$

$k=1$

$R_{11}^{(1)}$	1^*
$R_{12}^{(1)}$	1^*0
$R_{21}^{(1)}$	$\emptyset$
$R_{22}^{(1)}$	$\varepsilon+0+1$

$k=2$

$R_{11}^{(2)}$	1^*
$R_{12}^{(2)}$	$1^*0(0+1)^*$
$R_{21}^{(2)}$	$\emptyset$
$R_{22}^{(2)}$	$(0+1)^*$

- A última expressão regular equivalente ao autômato acima é construída tomando-se a união de todas as expressões em que o primeiro estado é o estado inicial e o segundo estado é algum estado de aceitação. No exemplo, 1 é o estado inicial e 2 é o único estado de aceitação, sendo assim, só precisamos de $R_{12}^{(2)}$:

$$R = 1^*0(0+1)^*$$

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

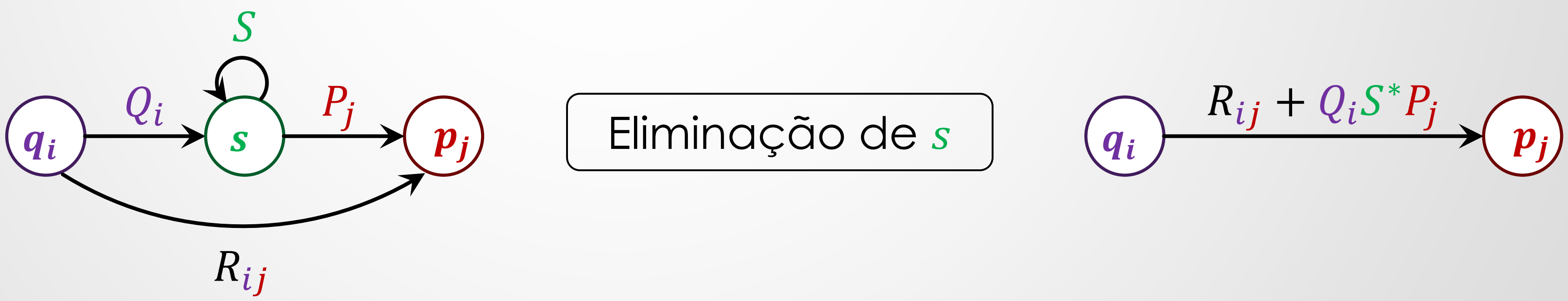
DFA para RE

- O método de conversão apresentado funciona também para NFA's e ε-NFA's, entretanto pode-se perceber o quão dispendioso ele é. No pior caso, o crescimento das expressões é da ordem de 4^n símbolos!
- Para tornar o processo de construção eficiente, veremos como realizar a eliminação de estados, o que implicará em termos autômatos que têm expressões regulares como rótulos das transições. Esses autômatos são denominados Autômatos Finitos Não-Determinísticos Generalizados (GNFA).

Expressões Regulares

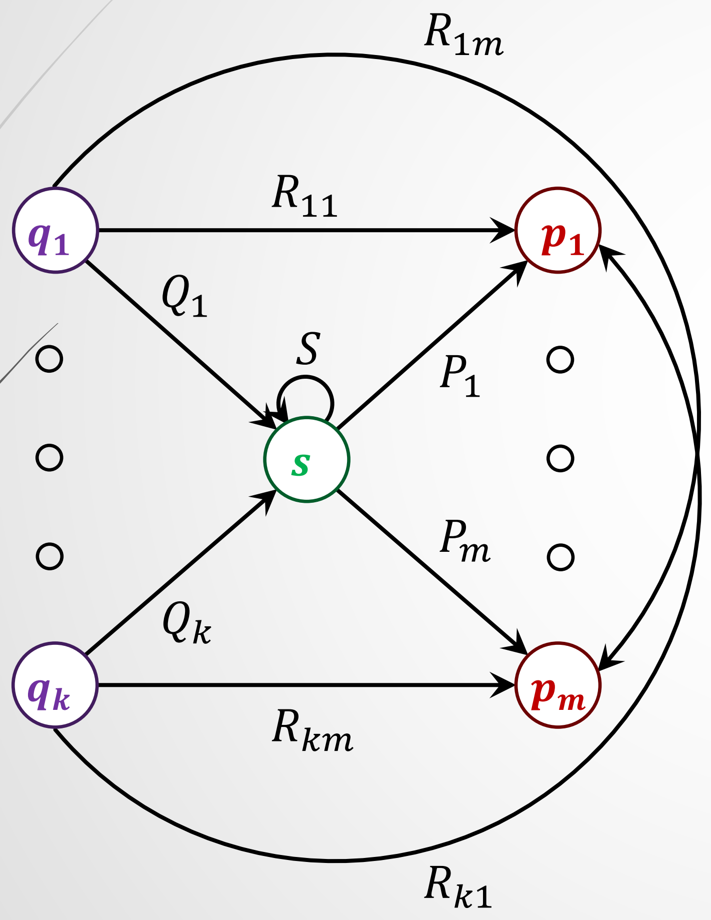
DFA para RE por eliminação de estados

- Eliminação de estados:
  - Para cada predecessor q_i de s e para cada sucessor p_j de s, introduz-se uma expressão regular que representa todos os caminhos que começam em q_i, vão até s, talvez façam um loop em torno de s zero ou mais vezes e, finalmente, vão até p_j. A expressão para esses caminhos é $Q_i S^* P_j$, sendo adicionada, com o operador de união, ao arco que vai de q_i a p_j. Se não houver nenhum arco $q_i \rightarrow p_j$, introduz-se esse arco com a expressão regular ∅.

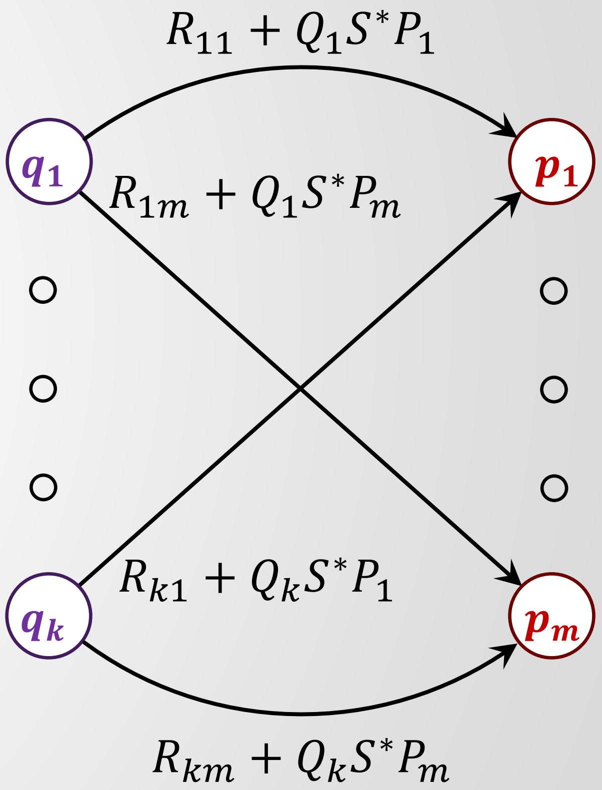


Expressões Regulares

DFA para RE por eliminação de estados

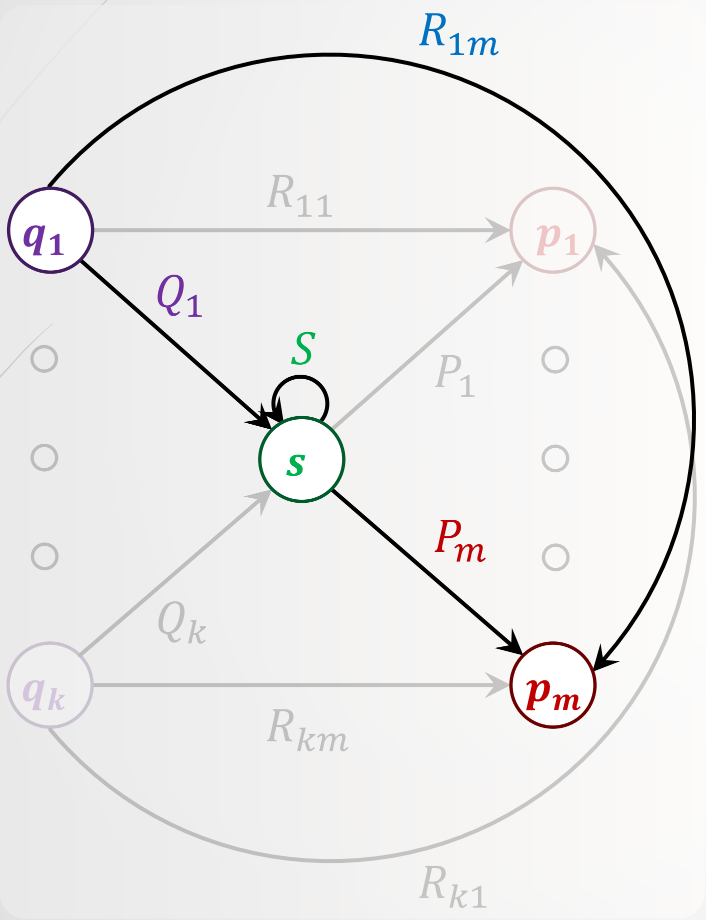


Eliminação de s



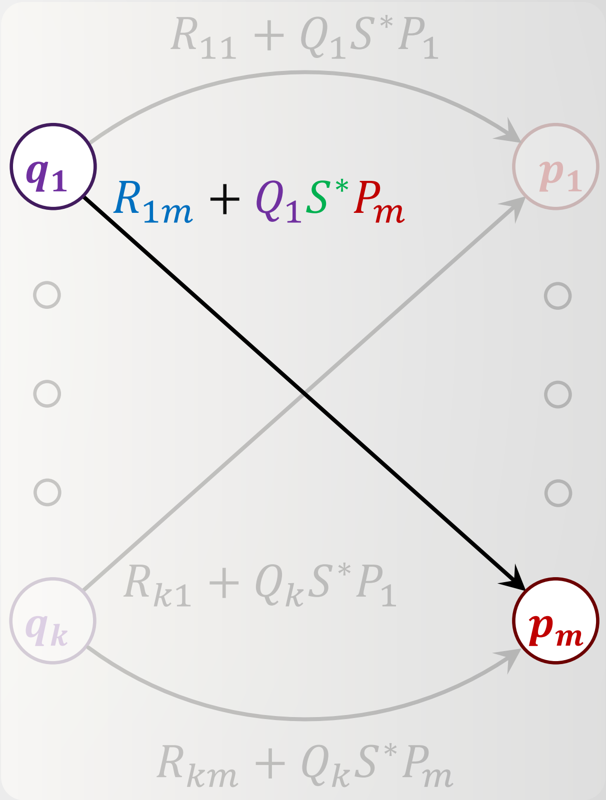
Expressões Regulares

DFA para RE por eliminação de estados



Eliminação de s

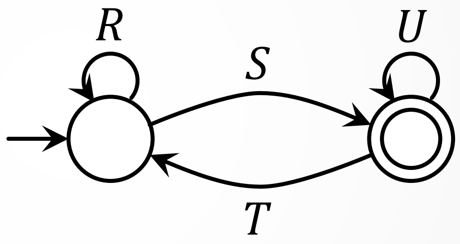
$q_1 \rightarrow p_m$: $R_{1m} + Q_1 S^* P_m$



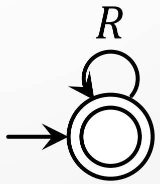
Expressões Regulares

DFA para RE por eliminação de estados

- Estratégia:
  1. Para cada estado de aceitação q, aplique o processo de redução, produzindo um GNFA equivalente. Elimine todos os estados, exceto q e o estado inicial q_0;
  2. Se $q \neq q_0$, ficaremos com um GNFA de dois estados equivalente à expressão regular $(R + SU^*T)^*SU^*$:



  3. Se $q = q_0$, ficaremos com um GNFA de um estado equivalente à expressão regular R^*:

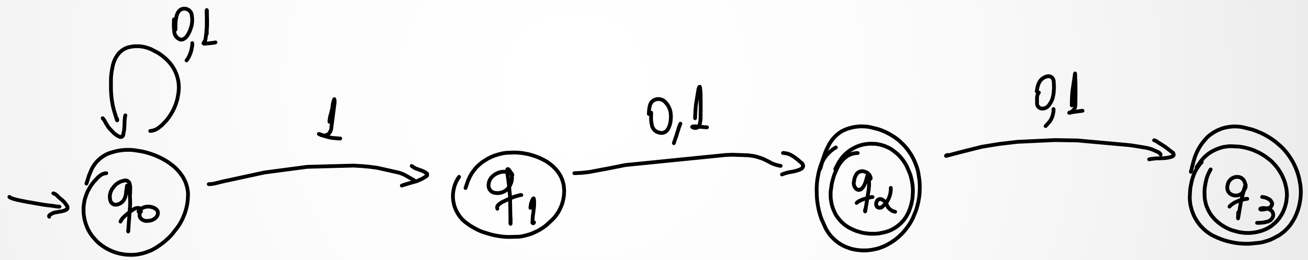


  4. A expressão regular desejada é a soma/união de todas as expressões derivadas dos autômatos reduzidos correspondentes à cada estado de aceitação pelas regras 2 e 3.

Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.

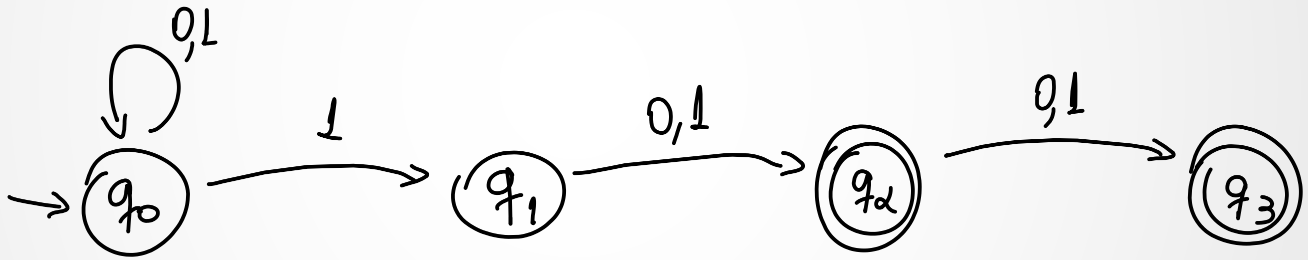


Expressões Regulares

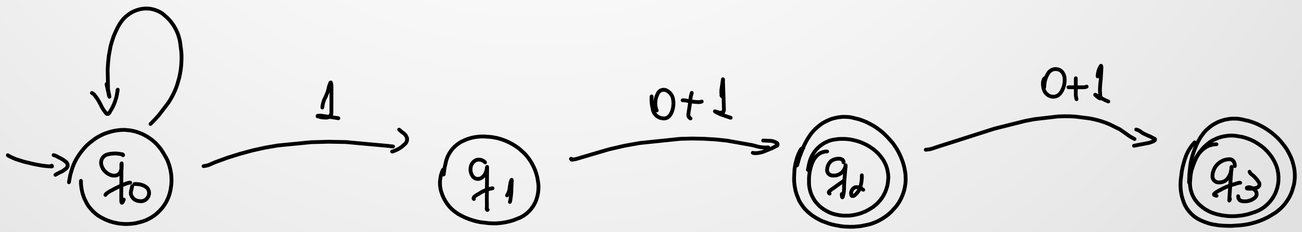
DFA para RE por eliminação de estados - Exemplo

- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.

Primeiro passo :



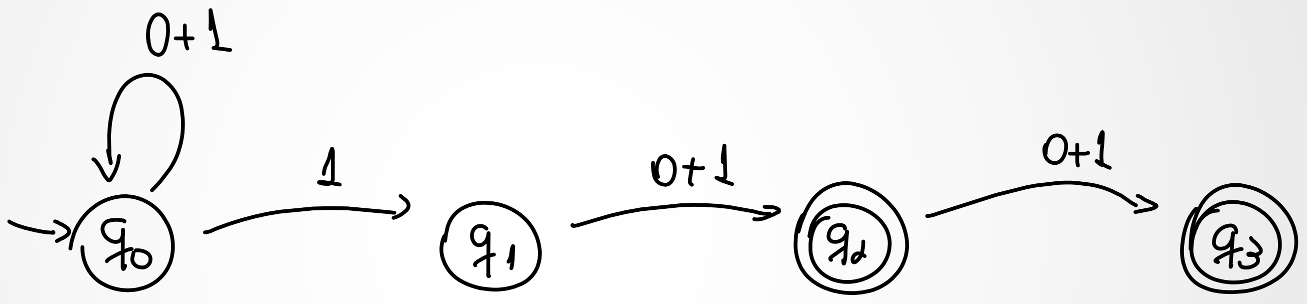
Gerar o GNFA equivalente :



Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

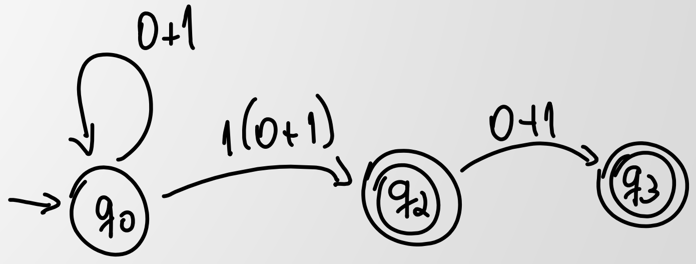
- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.



Realizando a eliminação de q_1:

$q_i \to p_j : R_{ij} + Q_i S^* P_j$

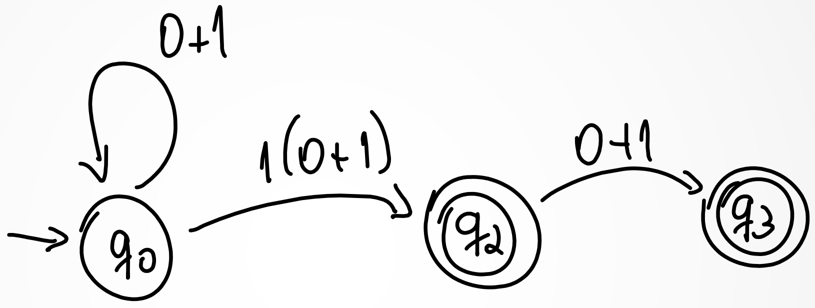
$q_0 \to q_2 : R_{02} + Q_0 S^* P_2 = \emptyset + 1\emptyset^*(0+1) = 1(0+1)$



Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

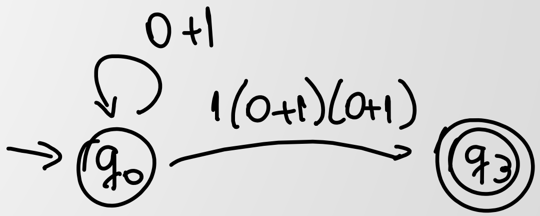
- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.



Realizando a eliminação de q_2:

$q_i \to p_j$: $R_{ij} + Q_i S^* P_j$

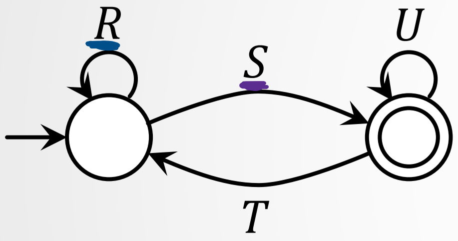
$q_0 \to q_3$: $R_{03} + Q_0 S^* P_3 = \emptyset + 1(0+1)\emptyset^*(0+1) = 1(0+1)(0+1)$



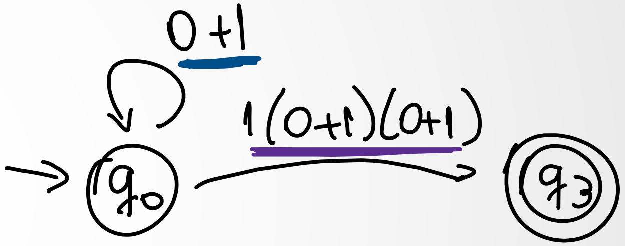
Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.



$(R + SU^*T)^*SU^*$



$(R + S\emptyset^*\emptyset)^* S\emptyset^* =$

$(R)^*S =$

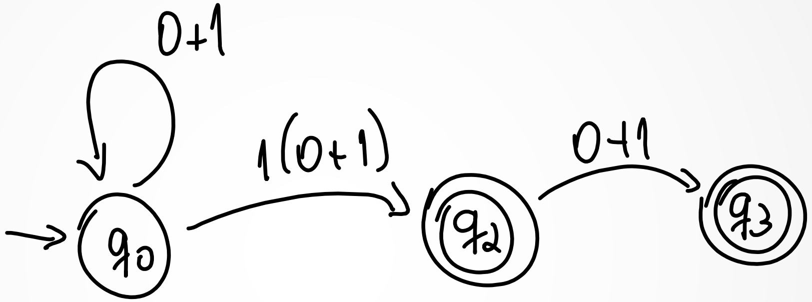
$R^*S \Rightarrow \boxed{(0+1)^*1(0+1)(0+1)}$

string com 1 na terceira posicao a partir do final

Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

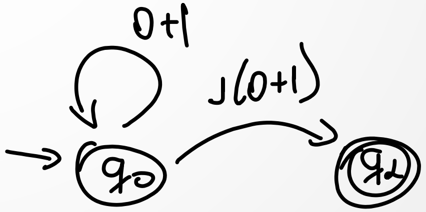
- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.



Realizando a eliminação de q_3:

q_3 ñ tem sucessor!

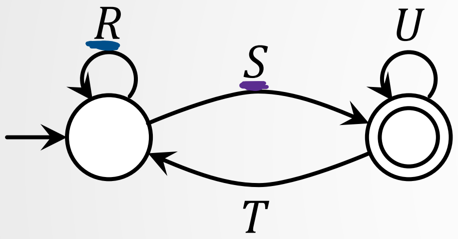
Pode ser eliminado, em conjunto com os arcos.



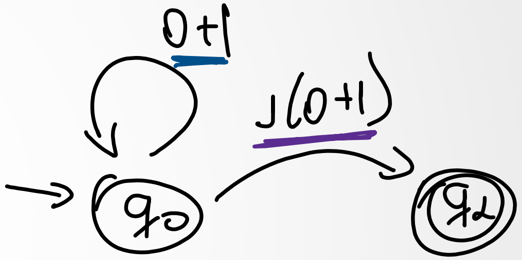
Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.



$(R + SU^*T)^*SU^*$



$(R + S\emptyset^*\emptyset)^* S\emptyset^* =$

$(R)^*S =$

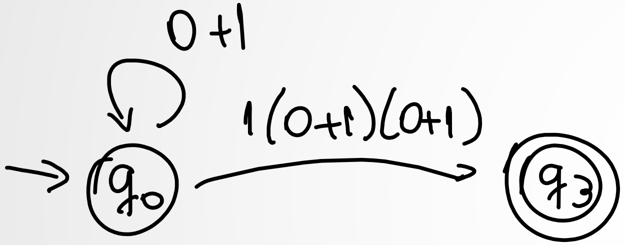
$R^*S \Rightarrow \boxed{(0+1)^* 1 (0+1)}$

string com 1 na segunda posição a partir do final.

Expressões Regulares

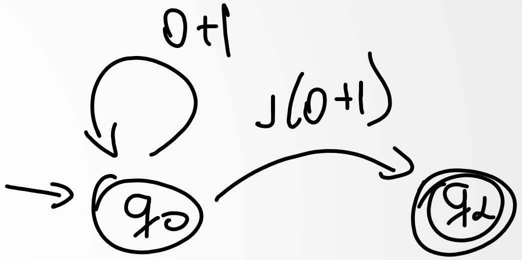
DFA para RE por eliminação de estados - Exemplo

- NFA que aceita todos as strings de 0's e 1's tais que a segunda ou a terceira posição a partir do final tem um símbolo 1.



$$\boxed{(0+1)^* 1 (0+1)(0+1)}$$

$$+$$



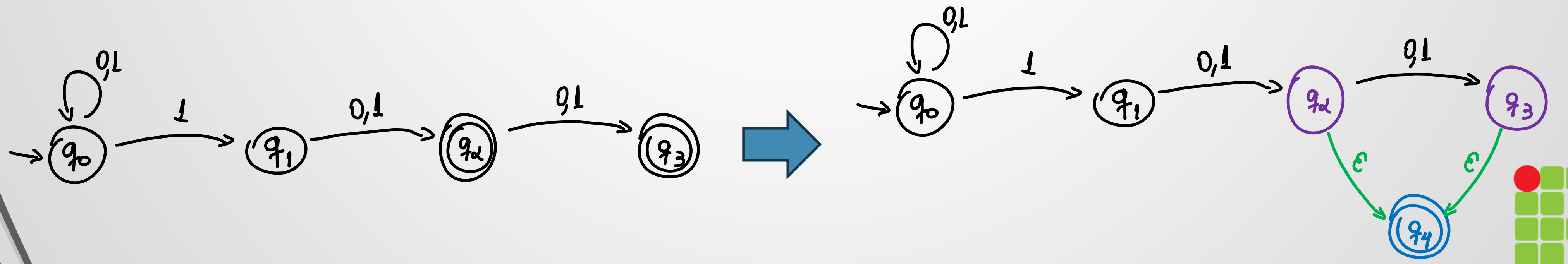
$$\boxed{(0+1)^* 1 (0+1)}$$

$$(0+1)^* 1 (0+1)(0+1) + (0+1)^* 1 (0+1)$$

Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

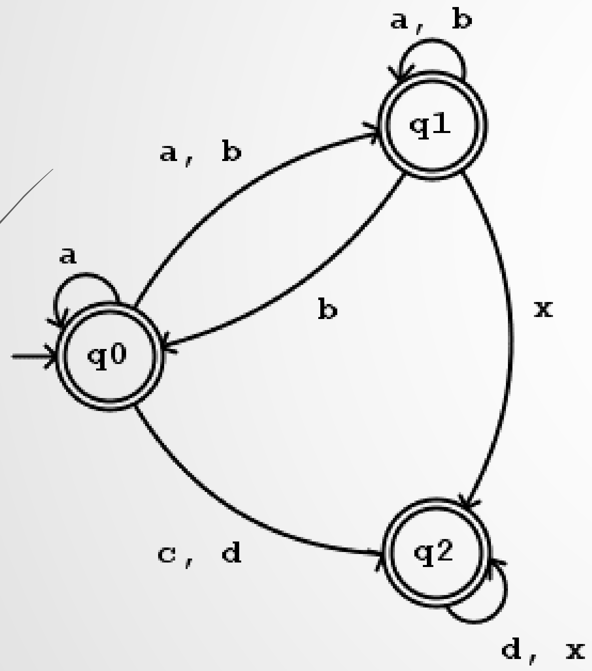
- Como pôde ser percebido no exemplo, tínhamos dois estados finais, o que nos obrigou a gerar dois GNFAs com um estado inicial e um final e, após isso, unimos as duas expressões regulares resultantes. Uma forma de abreviar ou facilitar tal processo, é fazer com o que o autômato finito inicial tenha exatamente um estado inicial e um estado final distintos e então proceder com a técnica de eliminação de estados. Para isso:
  - Criamos um novo estado que será o único estado final do autômato finito;
  - Criamos transições etiquetadas com ε partindo dos antigos estados finais para esse novo estado final;
  - Tornamos os antigos estados finais em estados de não aceitação.

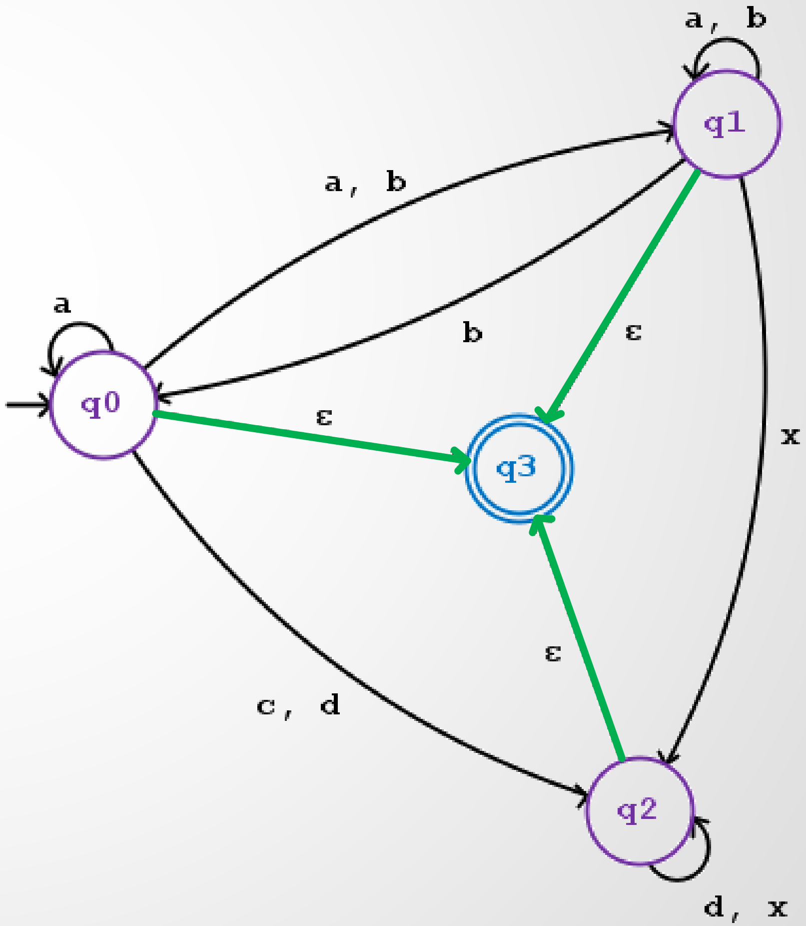


Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

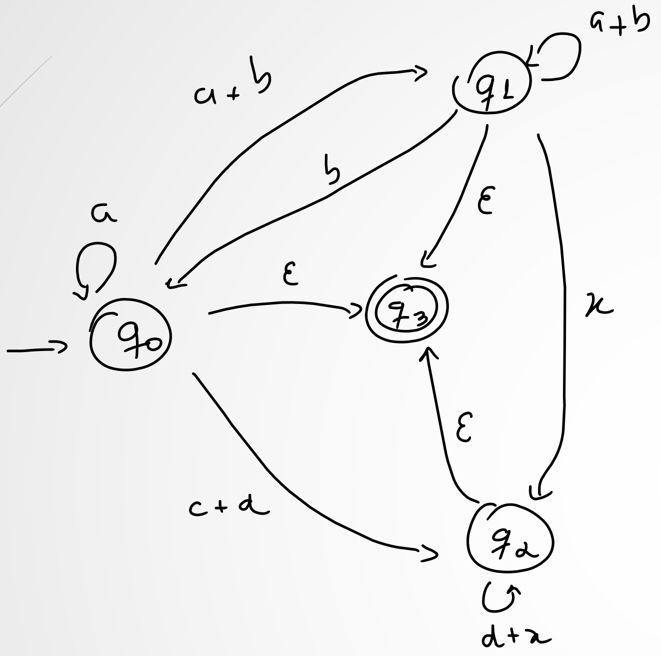
- Vamos fazer mais um exemplo:





Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

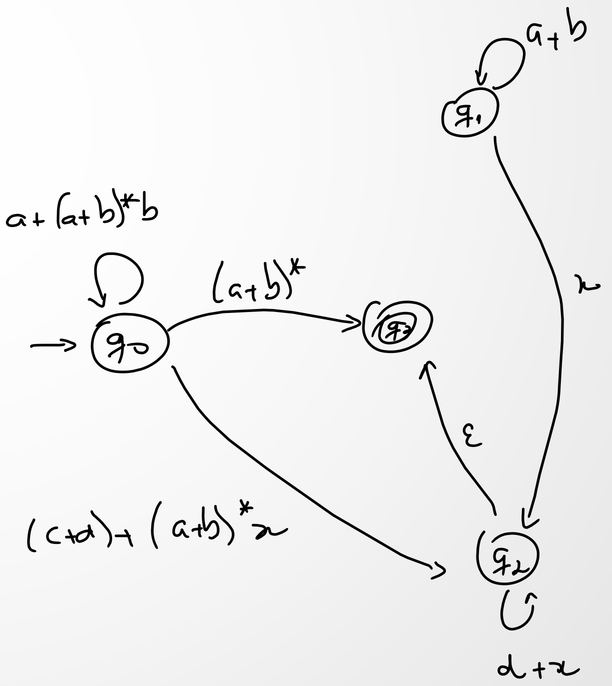


Eliminação de q_1:

$q_0 \to q_3$: $R_{03} + Q_0 S^* P_3 = \varepsilon + (a+b)(a+b)^* \varepsilon = (a+b)^*$

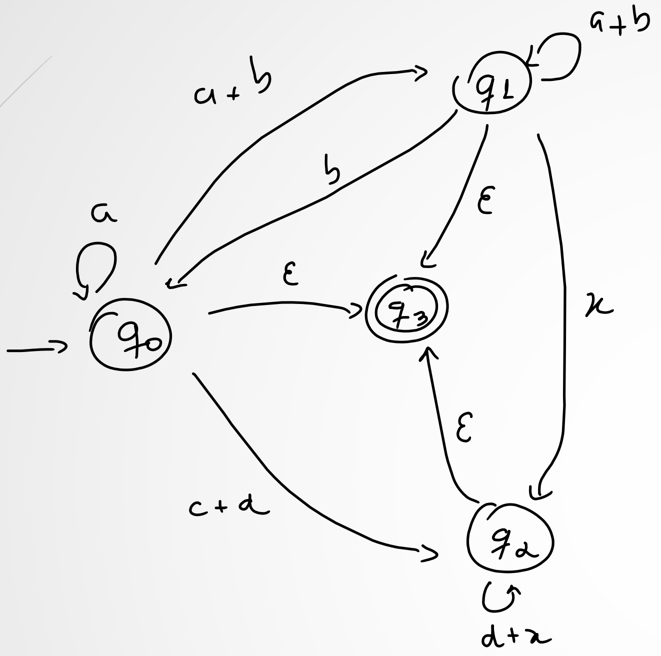
$q_0 \to q_2$: $R_{02} + Q_0 S^* P_2 = (c+d) + (a+b)(a+b)^* x = (c+d) + (a+b)^* x$

$q_0 \to q_0$: $R_{00} + Q_0 S^* P_0 = a + (a+b)(a+b)^* b = a + (a+b)^* b$



Expressões Regulares

DFA para RE por eliminação de estados - Exemplo

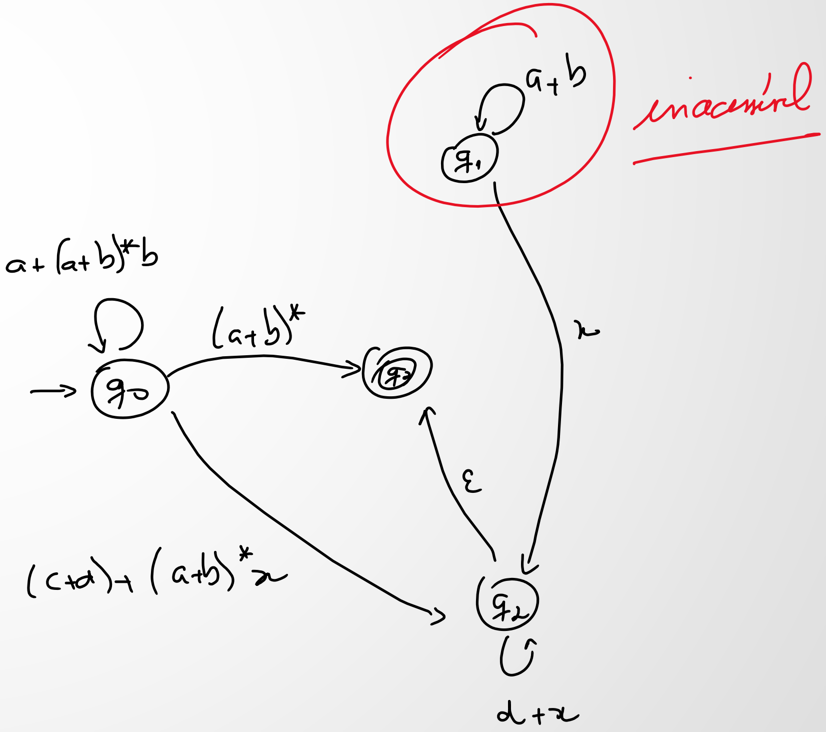


Eliminação de q_1:

$q_0 \to q_3$: $R_{03} + Q_0 S^* P_3 = \varepsilon + (a+b)(a+b)^* \varepsilon = (a+b)^*$

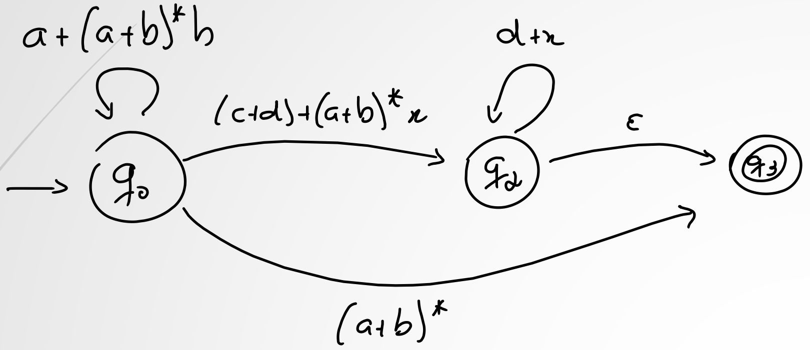
$q_0 \to q_2$: $R_{02} + Q_0 S^* P_2 = (c+d) + (a+b)(a+b)^* x = (c+d) + (a+b)^* x$

$q_0 \to q_0$: $R_{00} + Q_0 S^* P_0 = a + (a+b)(a+b)^* b = a + (a+b)^* b$



Expressões Regulares

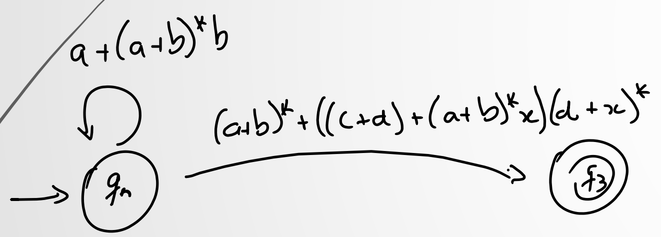
DFA para RE por eliminação de estados - Exemplo



Eliminação de q_2:

$q_0 \to q_3$: $R_{03} + Q_0 S^* P_3 = (a+b)^* + ((c+d)+(a+b)^*x)(d+x)^*\varepsilon =$

$(a+b)^*((c+d)+(a+b)^*x)(d+x)^*$



$E = (R+SU^*T)^*SU^* = (R+S\emptyset^*\emptyset)^*S\emptyset^* = R^*S =$

$$(a+(a+b)^*b)^*((c+d)+(a+b)^*x)(d+x)^*$$

Expressões Regulares

RE para ε-NFA

Teorema: Se $L = L(R)$ para alguma expressão regular R, então existe um ε-NFA E tal que $L = L(E)$

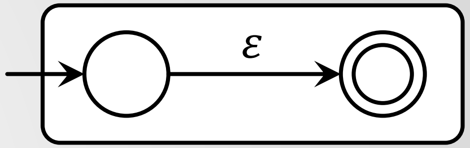
- **Prova:** Suponha que $L = L(R)$ para uma expressão regular R. Mostramos que $L = L(E)$ para algum ε-NFA E com:
  1. Exatamente um estado de aceitação;
  2. Nenhum arco chega no estado inicial;
  3. Nenhum arco sai do estado de aceitação.

  Prova por indução estrutural sobre R, seguindo a definição recursiva das expressões regulares que já vimos.

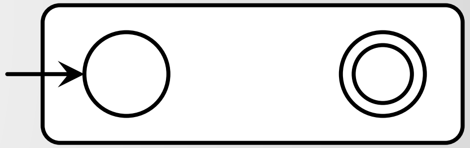
Expressões Regulares

RE para ε-NFA

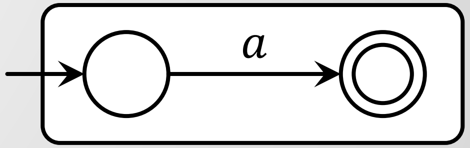
- **Base:**
  a) Manipulação da expressão ε. A linguagem do autômato deve ser $\{\varepsilon\}$;



  b) Construção para $\emptyset$. Como não existe caminho entre o estado inicial e o estado de aceitação, a linguagem do autômato é $\emptyset$;



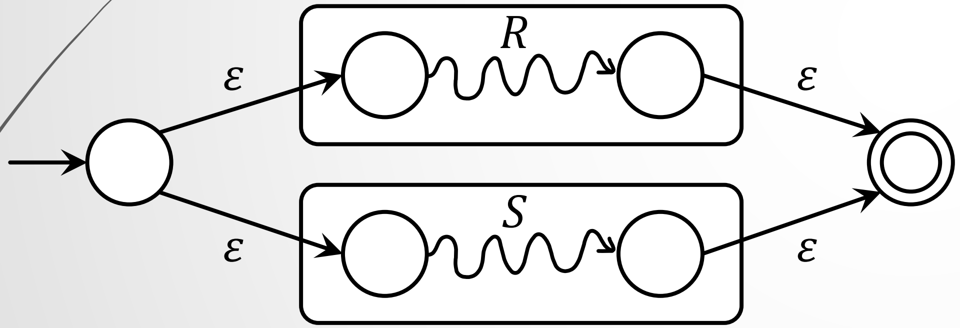
  c) Autômato para a expressão regular a. A linguagem desse autômato consiste na string a, que também é $L(a)$.



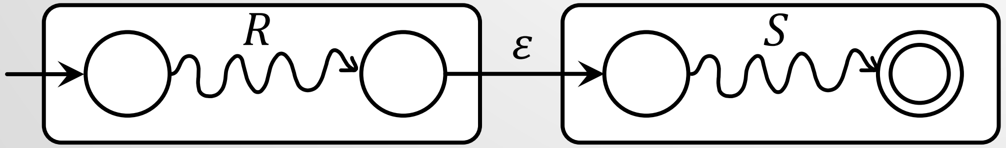
Expressões Regulares - RE para ε-NFA

- **Indução:** Supondo que a afirmação do teorema é verdadeira para subexpressões imediatas de uma dada expressão regular; isto é, as linguagens dessas subexpressões também são as linguagens dos ε-NFA's com um único estado de aceitação. Os quatro casos são:

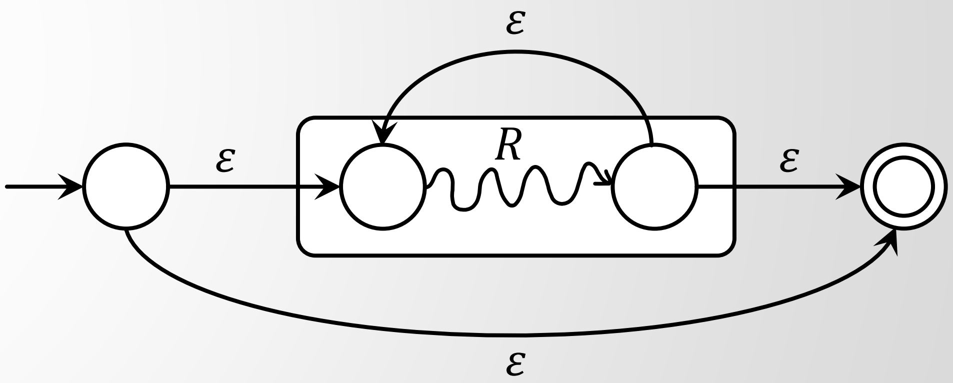
1. A expressão é $R + S$ para algumas expressões menores R e S.



2. A expressão é RS para algumas expressões menores R e S.



3. A expressão é R^* para alguma expressão menor R.



4. A expressão é (R) para alguma expressão menor R.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

RE para ε-NFA - Exemplo

- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.

Expressões Regulares

RE para ε-NFA - Exemplo

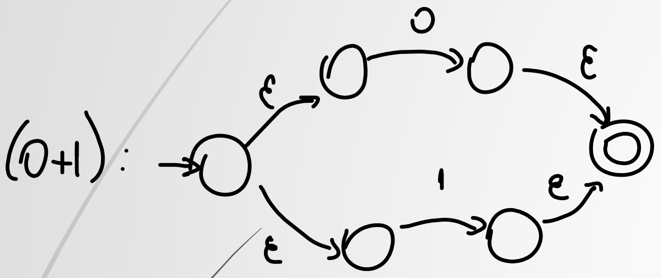
- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.

$(0+1)$:

Expressões Regulares

RE para ε-NFA - Exemplo

- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.

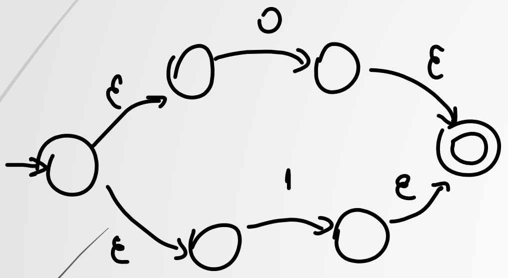


Expressões Regulares

RE para ε-NFA - Exemplo

- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.

$(0+1)$:

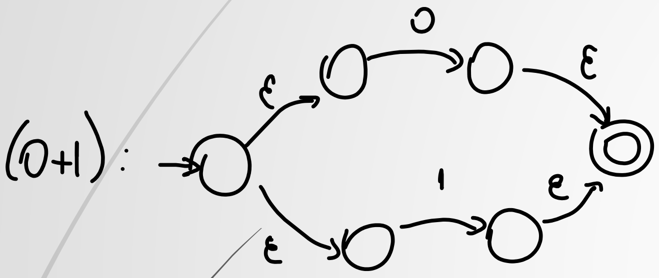


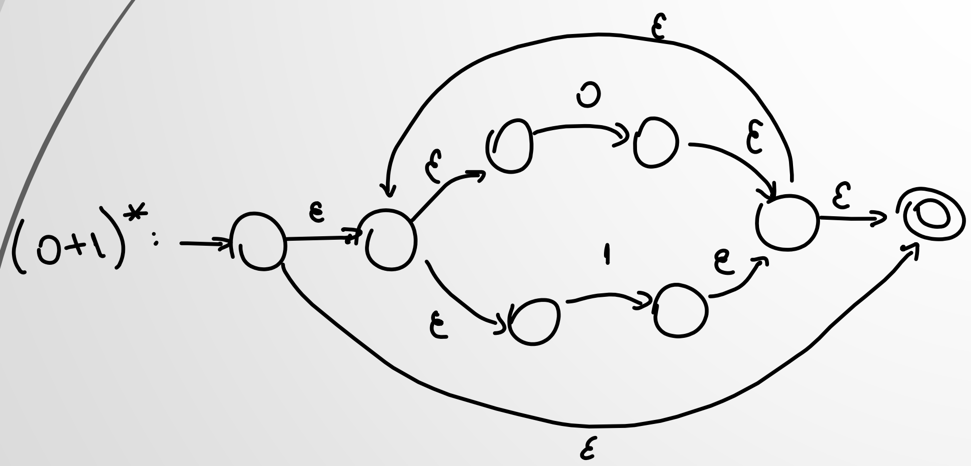
$(0+1)^*$:

Expressões Regulares

RE para ε-NFA - Exemplo

- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.



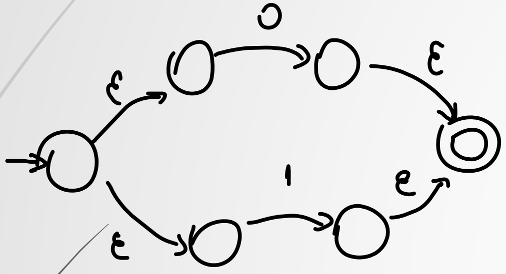


Expressões Regulares

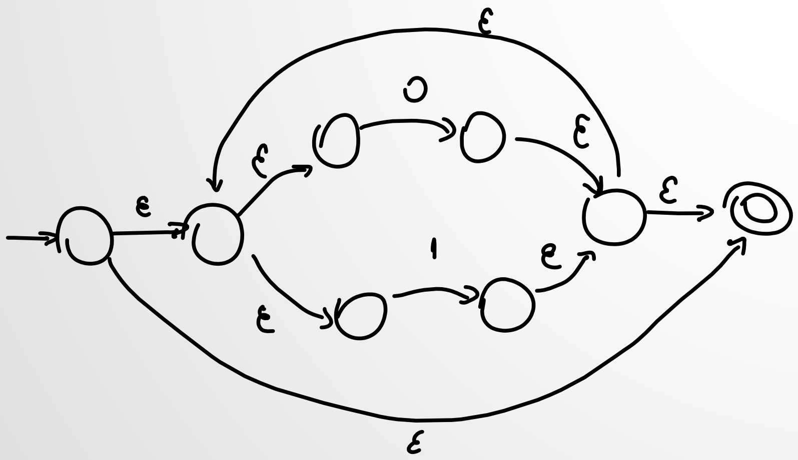
RE para ε-NFA - Exemplo

- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.

$(0+1)$:



$(0+1)^*$:



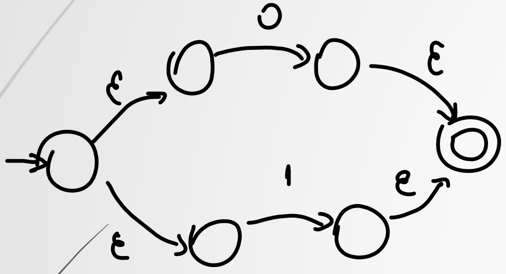
$(0+1)^*1(0+1)$:

Expressões Regulares

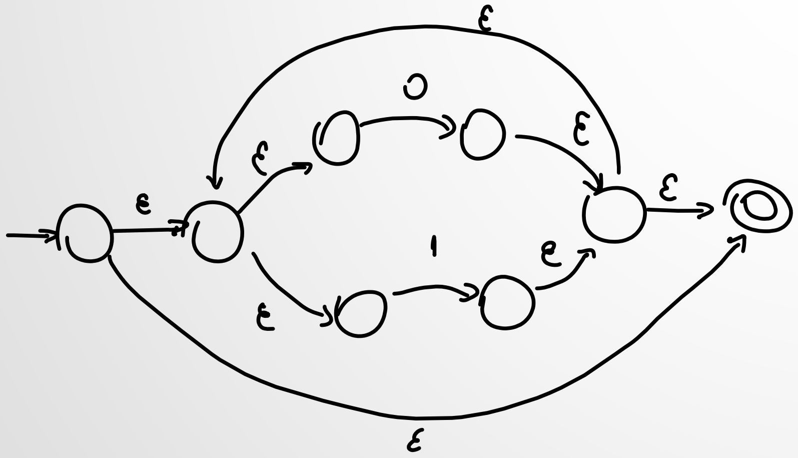
RE para ε-NFA - Exemplo

- Converter a expressão regular $(0+1)^*1(0+1)$ em um ε-NFA.

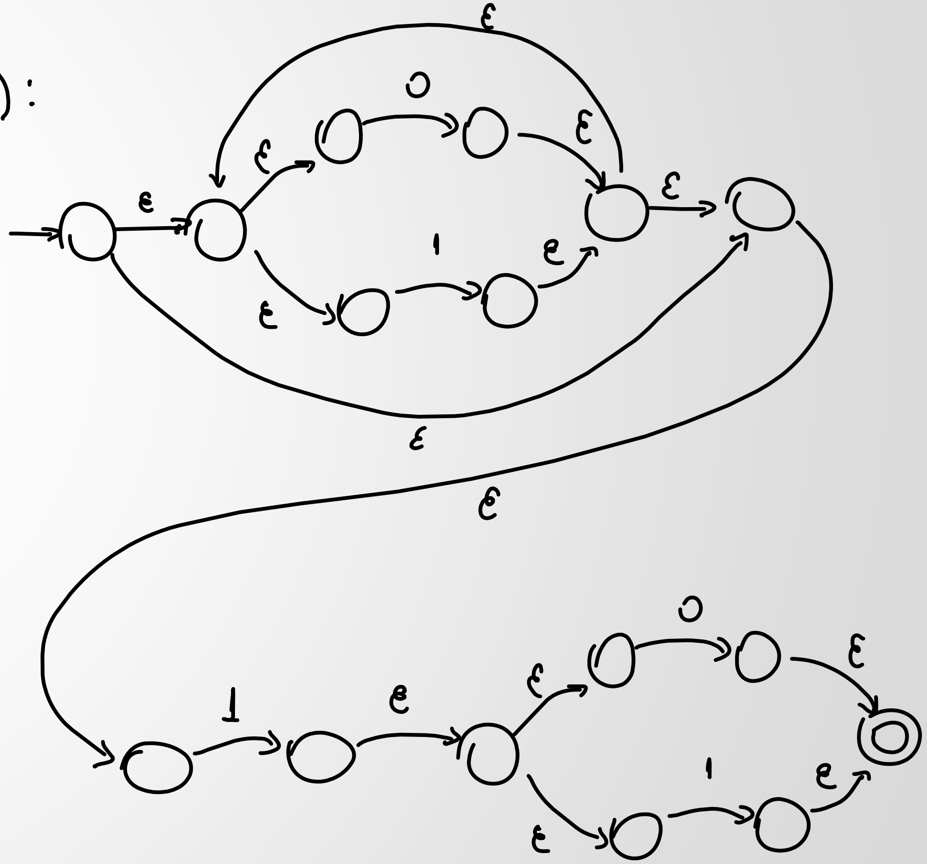
$(0+1)$:



$(0+1)^*$:



$(0+1)^*1(0+1)$:



Expressões Regulares

Exercícios Escritos

Exercício e5.1: Escreva expressões regulares correspondentes às seguintes linguagens:

a) O conjunto de strings sobre o alfabeto $\{a, b, c\}$ que contém pelo menos um a e pelo menos um b;

b) O conjunto de strings de 0's e 1's cujo décimo símbolo a partir da extremidade direita é 1;

c) O conjunto de strings de 0's e 1's com no máximo um par de 1's consecutivos.

Expressões Regulares

Exercícios Escritos

Exercício e5.2: Considerando o autômato finito a seguir:

a) Forneça todas as expressões $R_{ij}^{(0)}$;

b) Forneça todas as expressões $R_{ij}^{(1)}$. Procure simplificar ao máximo as expressões;

c) Construa o diagrama de transições para o DFA e forneça uma expressão regular para sua linguagem, eliminando o estado q_2.

	0	1
$\rightarrow q_1$	q_2	q_1
q_2	q_3	q_1
$* \, q_3$	q_3	q_2

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Campus São João da Boa Vista

Expressões Regulares

Exercícios Escritos

Exercício e5.3: Considerando o autômato finito a seguir:

a) Forneça todas as expressões $R_{ij}^{(0)}$;

b) Forneça todas as expressões $R_{ij}^{(1)}$. Procure simplificar ao máximo as expressões;

c) Construa o diagrama de transições para o DFA e forneça uma expressão regular para sua linguagem, eliminando o estado q_2.

	0	1
$\rightarrow q_1$	q_2	q_3
q_2	q_1	q_3
$* q_3$	q_2	q_1

Expressões Regulares

Exercícios Escritos

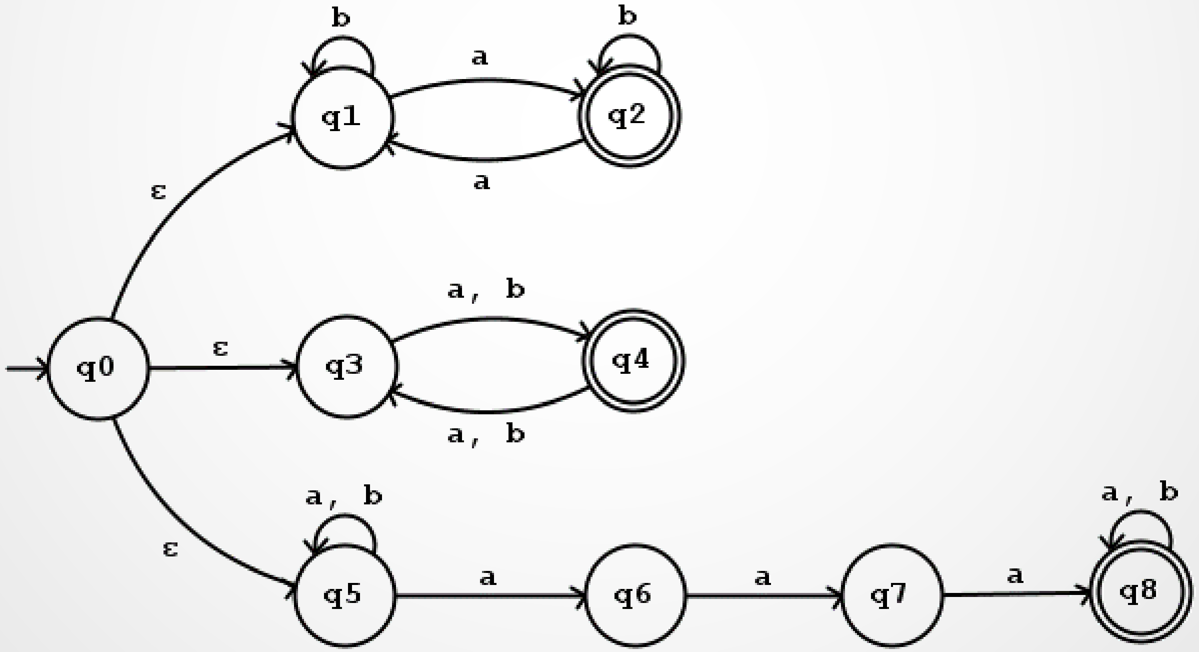
Exercício e5.4: Considerando o autômato finito a seguir, converta-o em uma expressão regular, usando a técnica de eliminação de estados.

	0	1
$\rightarrow * p$	s	p
q	p	s
r	r	q
s	q	r

Expressões Regulares

Exercícios Escritos

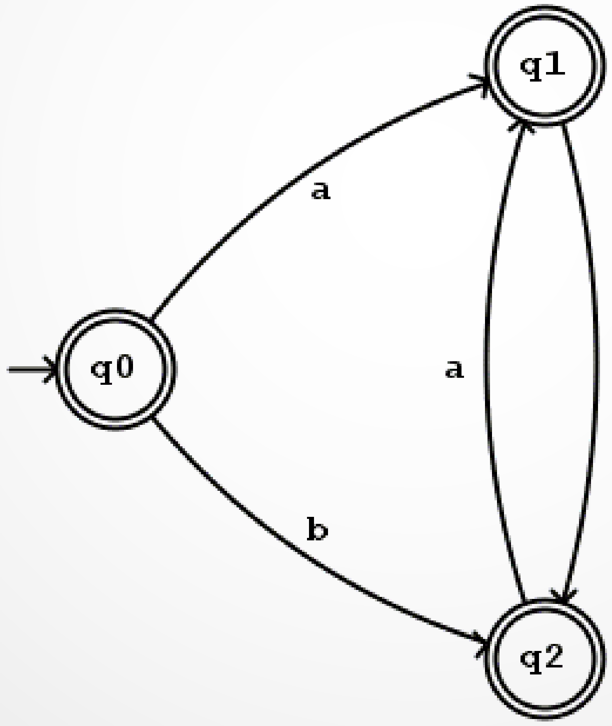
Exercício e5.5: Considerando o autômato finito a seguir, converta-o em uma expressão regular, usando a técnica de eliminação de estados.



Expressões Regulares

Exercícios Escritos

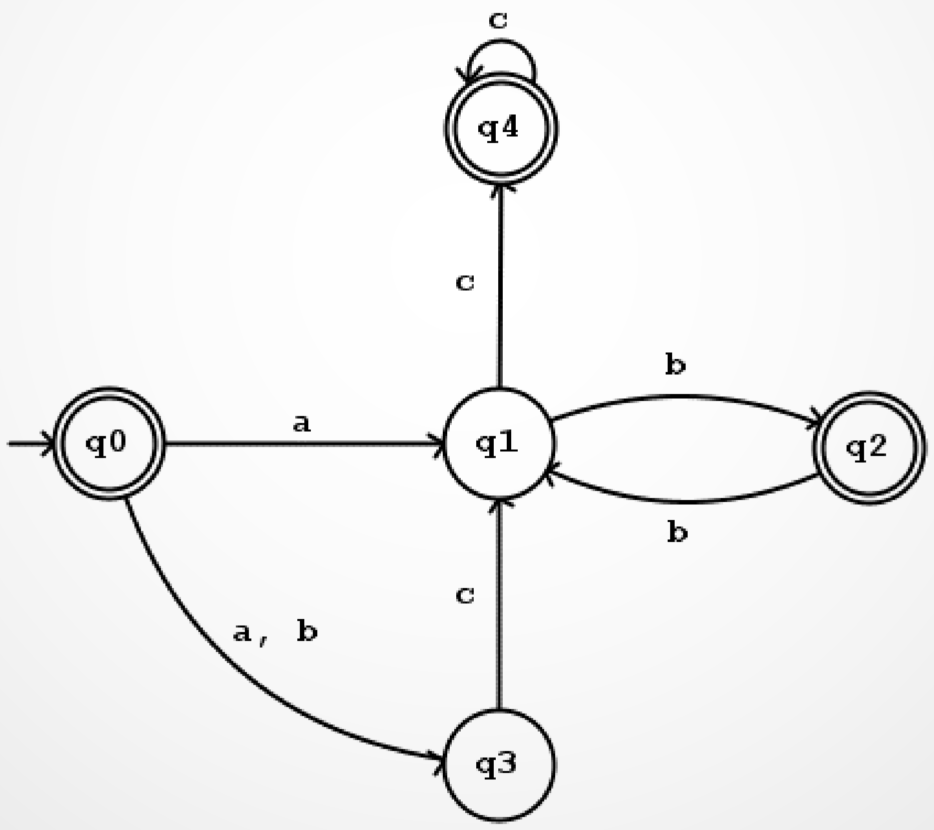
Exercício e5.6: Considerando o autômato finito a seguir, converta-o em uma expressão regular, usando a técnica de eliminação de estados.



Expressões Regulares

Exercícios Escritos

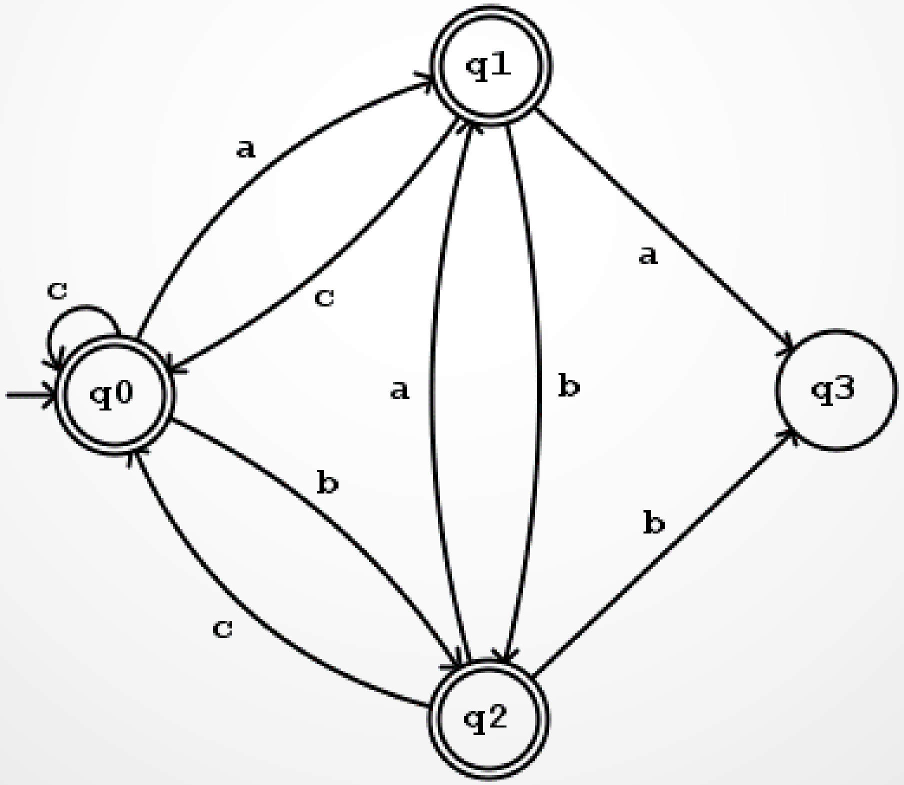
Exercício e5.7: Considerando o autômato finito a seguir, converta-o em uma expressão regular, usando a técnica de eliminação de estados.



Expressões Regulares

Exercícios Escritos

Exercício e5.8: Considerando o autômato finito a seguir, converta-o em uma expressão regular, usando a técnica de eliminação de estados.



Expressões Regulares

Exercícios Escritos

Exercício e5.9: Converta as expressões regulares a seguir em ε-NFA's.

a) 01^*

b) $(0+1)01$

c) $00(0+1)$

d) $\left(a^*b^*(a+ac^*)\right)^*$

e) $(a+b+c)^*a(a+b+c)^*b(a+b+c)^*c(a+b+c)^*$

f) $(a+b)^*(a+ab)(a+b)^*$

g) $a^*b^*a^*b^*$

Bibliografia

HOPCROFT, J. E.; ULLMAN, J. D.; MOTWANI, R. **Introdução à Teoria de Autômatos, Linguagens e Computação**. 2. ed. Rio de Janeiro: Elsevier, 2002. 560 p.

RAMOS, M. V. M.; JOSÉ NETO, J.; VEGA, I. S. **Linguagens Formais: Teoria, Modelagem e Implementação**. Porto Alegre: Bookman, 2009. 656 p.

SIPSER, M. **Introdução à Teoria da Computação**. 2. ed. São Paulo: Cengage Learning, 2017. 459 p.