

# O algoritmo de conversão de Autômato Finito Não-Determinístico - AFND para Expressão Regular - ER

Eduardo Couto Dinarte,  
Iago Gade Gusmao Carrazzoni,  
Lucca Maciel de Moraes

21 de Novembro de 2018

## **Resumo**

Este artigo consiste na apresentação e explicação de um algoritmo para converter um autômato finito não determinístico num autômato finito determinístico e, por fim, converter este numa expressão regular. O método consiste em apresentar a teoria com imagens dos três estados da conversão seguida de um exemplo prático. O objetivo deste texto é fixar o conteúdo de conversão de autômatos e familiarizar os autores com a produção de artigos científicos utilizando a linguagem Latex.

# 1 Introdução aos Autômatos

Um autômato é uma máquina abstrata que deve operar entre estados previamente definidos. É um modelo matemático utilizado para representar programas ou circuitos lógicos. É bem definido por uma quintupla, cujos elementos são:

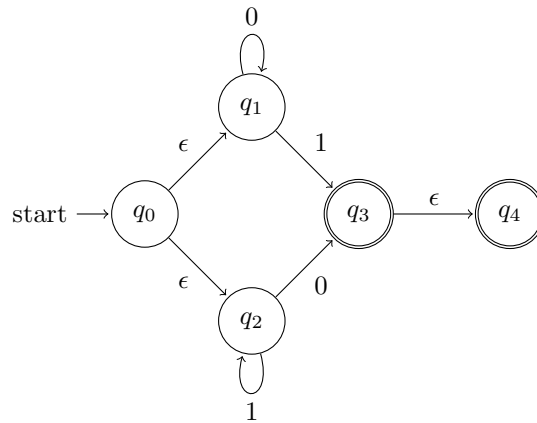
- Conjunto de estados ( $K$ );
- Alfabeto ( $A$ );
- Estado inicial ( $S$ );
- Conjunto de estados finais ( $F$ );
- Função de transição (ou função delta).

A função de transição, por sua vez, é representada por uma tripla ordenada, onde os elementos são:

- Estado inicial;
- Transição;
- Estado final;

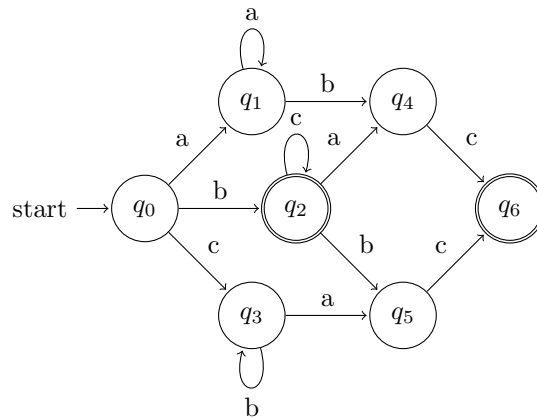
## 2 Introdução ao Autômato Finito Não-Determinístico - AFND

Autômato finito não determinístico é aquele em que, em algum momento, não se tem certeza de qual é o estado atual, ou seja, é aquele que tem a palavra vazia ligando algum de seus estados. Segue um exemplo a seguir:



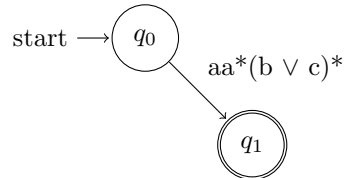
### 3 Introdução ao Autômato Finito Determinístico - AFD

Autômato finito determinístico é aquele em que se sabe exatamente qual o estado atual, ou seja, é aquele que não tem estados simultâneos (estados ligados por palavras vazias).



## 4 Introdução à Expressão Regular - ER

Expressão regular é uma cadeia de caracteres que engloba todas as palavras aceitas pelo autômato. Um autômato reduzido a expressão regular possui apenas um estado inicial e um estado final, ligados pela expressão regular.



No tipo citado, é comum a aparição do caractere  $\vee$ , assim como o parêntese. Este se aplica da mesma forma que na matemática. Aquele é o conectivo lógico 'ou', que se aplica da mesma forma que na lógica.

Também é comum a aparição do caractere '\*' na expressão regular. Ele se chama estrela de Kleene, e denota zero ou mais repetições do caractere ( ou cadeia de caracteres ) ao qual foi aplicado.

No exemplo acima, a estrela de Kleene foi aplicada ao caractere 'a' e à expressão  $(b \vee c)$ . Neste, quer dizer que zero ou mais repetições da cadeia denotada serão aceitas, enquanto naquele, zero ou mais repetições do caractere denotado serão aceitos.

## 5 Conversão de AFND para AFD

Para essa conversão, é utilizado o Algoritmo de Conversão de um Autômato Finito Não-determinístico (AFND) em um Autômato Finito Determinístico, que consiste em:

- Identificar os estados simultâneos do AFND;
- Identificar o estado inicial P0, o qual seu conjunto possui apenas o estado inicial da AFND;
- Aplicar em P0 a leitura de todo o alfabeto. O conjunto novo será composto pelo lugar da chegada;
- Identificar os estados resultantes;
- Para cada estado resultante criado, aplica-se o alfabeto;
- Repetir o procedimento até que não existam mais estados novos;
- Identificar os estados finais, que serão aqueles estados que possuírem os estados finais da AFND;
- Montar a quintupla do AFD;
- Por fim, esboçar o grafo.

Para exemplificar, será realizada a conversão do AFND a seguir, cuja quintupla

é:

$K = \{0, 1, 2, 3, 4, 5\}$

$A = \{a, b, c\}$

$S = \{0\}$

$F = \{3, 4\}$

$D =$

$(0, \epsilon, 1),$

$(0, \epsilon, 2),$

$(1, b, 3)$

$(1, a, 5),$

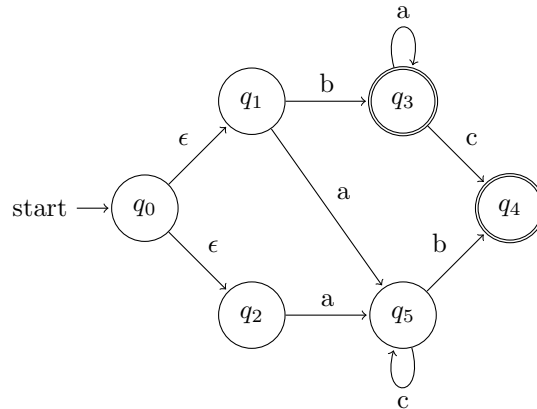
$(2, a, 5)$

$(3, a, 3),$

$(3, c, 4),$

$(5, c, 5),$

$(5, b, 4).$



Daqui em diante as seguintes notações serão usadas:

- $E(x)$ : denota o conjunto de estados simultâneos a  $x$ ;
- $P(x)$ : denota um estado maior, que engloba vários outros;
- $D(P(x), A)$ : denota a função delta de  $P(x)$  aplicando o alfabeto.

Seguindo o algoritmo, o procedimento será o seguinte:

- Identificar os estados simultâneos do AFND:  
 $E(0) = \{0, 1, 2\}$   
 $E(1) = \{1\}$   
 $E(2) = \{2\}$   
 $E(3) = \{3\}$   
 $E(4) = \{4\}$   
 $E(5) = \{5\}$
- identificar o estado inicial  $P_0$ , o qual seu conjunto possui o estado inicial da AFND:  
 $P(0) = E(0) = \{0, 1, 2\}$
- aplicar em  $P_0$  a leitura de todo o alfabeto. O conjunto novo será composto pelo lugar da chegada:  
 $D(P(0), a) = (1, a, 5) \cup (2, a, 5) = E(5) \cup E(5) = \{5\}$   
 $D(P(0), b) = (1, b, 3) = E(3) = \{3\}$   
 $D(P(0), c) = \text{vazio}$
- Identificar os estados resultantes:  
 $P(1) = D(P(0), a) = E(5) = \{5\}$   
 $P(2) = D(P(0), b) = E(3) = \{3\}$
- para cada estado resultante criado, aplica-se o alfabeto:  
 $D(P(1), a) = \text{vazio}$   
 $D(P(1), b) = (5, b, 4) = E(4) = \{4\}$   
 $D(P(1), c) = (5, c, 5) = E(5) = \{5\}$   
 $D(P(2), a) = (3, a, 3) = E(3) = \{3\}$   
 $D(P(2), b) = \text{vazio}$   
 $D(P(2), c) = (3, c, 4) = E(4) = \{4\}$

- repetir o procedimento até que não existam mais estados novos:

$$D(P(1), b) = P(3) = E(4) = \{4\}$$

$$D(P(1), c) = P(1) = E(1) = \{1\}$$

$$D(P(2), a) = P(2) = E(2) = \{2\}$$

$$D(P(2), c) = P(3) = \{4\}$$

$$D(P(3), a) = \text{vazio}$$

$$D(P(3), b) = \text{vazio}$$

$$D(P(3), c) = \text{vazio}$$

- identificar os estados finais, que serão aqueles estados que possuírem os estados finais do AFND:

$$F = P(1), P(3)$$

- montar a quintupla do AFD:

$$K = \{P(0), P(1), P(2), P(3)\}$$

$$A = \{a, b, c\}$$

$$S = \{P(0)\}$$

$$F = \{P(2), P(3)\}$$

Pode ser difícil observar os elementos da função delta. Eles serão todos os  $D(P(x), \text{'caractere'})$  que calculamos. Assim:

$D =$

$$(P_0, a, P_1),$$

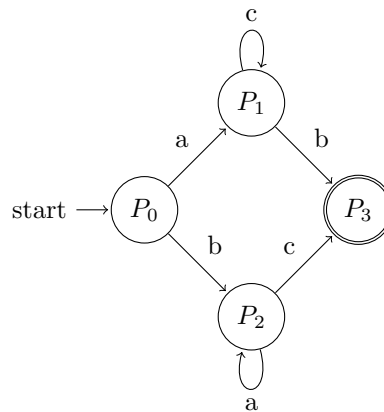
$$(P_0, b, P_2),$$

$$(P_1, b, P_3),$$

$$(P_1, c, P_1),$$

$$(P_2, a, P_2),$$

$$(P_2, c, P_3).$$

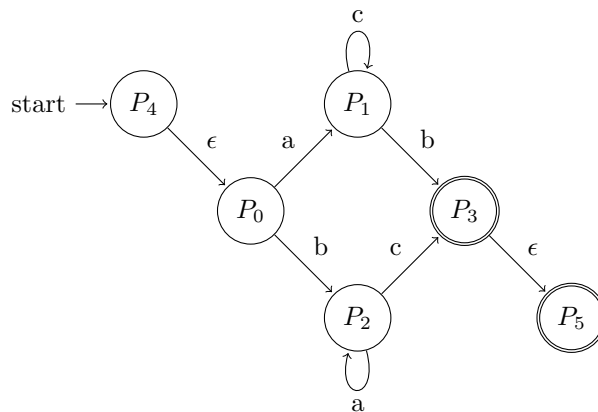




## 6 Conversão AFD - ER

Nessa conversão, o número de transições que o estado inicial possui será muito importante: a ER será uma composição das expressões obtidas seguindo cada um dos caminhos a partir do estado inicial. Por exemplo, se o estado inicial possui três transições para outros estados, a ER será uma composição de três expressões, separadas, na ER final, pelo conectivo lógico 'OU'. Em suma, se há três caminhos, é possível seguir pelo primeiro OU pelo segundo OU pelo terceiro. Essa lógica se mantém na ER. Esclarecida esta questão, inicia-se o algoritmo.

- Identificar a quantidade de transições do estado inicial.  
Nº de transições do estado inicial: 2.
- Criar dois outros estados. Um estará ligado aos estados finais do autômato por uma palavra vazia, enquanto o outro estará ligado ao estado inicial pela mesma. Aquele ligado ao estado inicial passará a ser o novo estado inicial, enquanto o ligado aos estados finais, passará a ser o estado final. Eles serão os únicos existentes na ER.



- Colapsar estados até que só reste os dois criados. Consiste em retirar um estado, ligando o estado de chegada nele ao estado no qual ele chega. A escolha do estado a se colapsar é completamente arbitrária. No exemplo, os primeiros estados a serem colapsados serão o P4 e o P2. A única transição nesses estados tem os mesmos como estado inicial e final. Portanto, é a mais simples situação a se considerar. Basta inserir  $A^*$  na transição, onde A é um caractere qualquer.

