

O algoritmo de conversão de AFND para ER

Eduardo Couto Dinarte

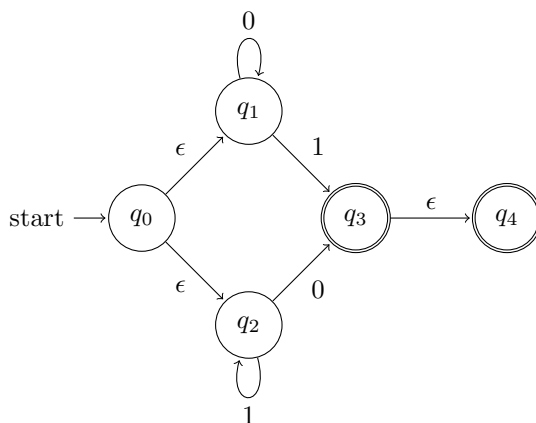
12 de Novembro de 2018

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.

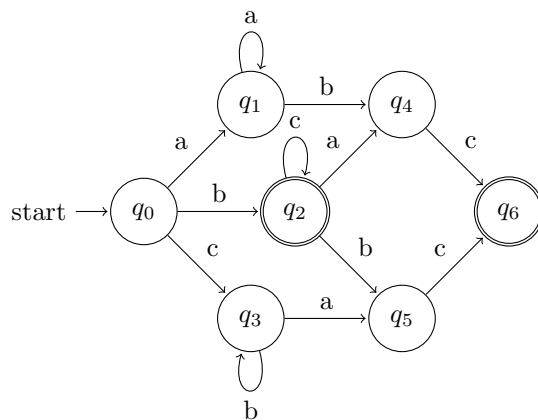
1 Introdução ao Automato Finito Não-Determinístico

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.



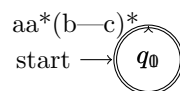
2 Introdução ao Autômato Finito Determinístico

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).



3 Introdução à Expressão Regular

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 '—', 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—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.

4 Implementação

Para conseguir blablabla

O algoritmo *Minimax* segue abaixo:

Algoritmo 1 Algoritmo Minimax

```
1: function MINIMAX(estado)                                ▷ retorna uma ação
2:   Entradas: estado é a configuração atual do jogo
3:    $v \leftarrow \text{maxvalor}(\text{estado})$ 
4:   retorna a ação  $a$  em sucessores(estado) cujo valor é  $v$ 
5: end function
6: function MAXVALOR(estado)                                ▷ retorna o valor estático
7:   if fim(estado) then
8:     retorna estatico(estado)
9:   end if
10:   $v \leftarrow -\infty$ 
11:  for todas ações  $a$  nos sucessores(estado) do
12:     $v \leftarrow \max(v, \text{minvalor}(a))$ 
13:  end for
14:  retorna  $v$ 
15: end function
16: function MINVALOR(estado)                                ▷ retorna o valor estático
17:   if fim(estado) then
18:     retorna estatico(estado)
19:   end if
20:   $v \leftarrow \infty$ 
21:  for todas ações  $a$  nos sucessores(estado) do
22:     $v \leftarrow \min(v, \text{maxvalor}(a))$ 
23:  end for
24:  retorna  $v$ 
25: end function
```
