Teoria dos Autômatos e Linguagens Formais

aula 09: Autômatos não-determinísticos

1 Introdução

Na aula passada, a gente viu que é fácil construir a expressão regular que descreve as palavras que um autômato aceita.

Hoje nós vamos considerar o problema contrário.

Quer dizer, hoje nós queremos construir o autômato que aceita as palavras que são descritas por uma expressão regular.

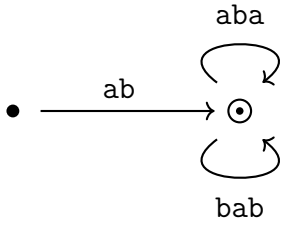
Considere o seguinte exemplo,

$$\texttt{ab}\left(\texttt{aba} \cup \texttt{bab}\right)^*$$

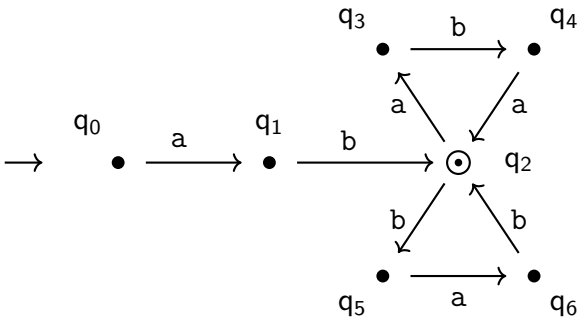
Decompondo a expressão em duas partes

$$\texttt{ab} \;\Big|\; \left(\texttt{aba} \cup \texttt{bab}\right)^*$$

a gente vê que cada parte corresponde a um pedaço do seguinte diagrama



Daí, decompondo a transição e os laços, nós chegamos a



E agora, basta acrescentar o estado de rejeição para completar a construção.

A ideia é que a expressão regular é um autômato disfarçado.

E que desmontando ela passo a passo, nós chegamos no diagrama do autômato.

Legal, não?

2 Expressões regulares e autômatos

Mas agora, nós vamos complicar um pouquinho as coisas.

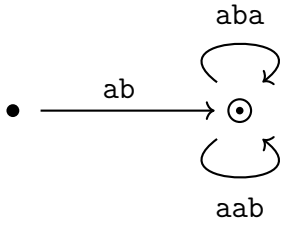
Considere a expressão regular

$$\texttt{ab}\left(\texttt{aba} \cup \texttt{aab}\right)^*$$

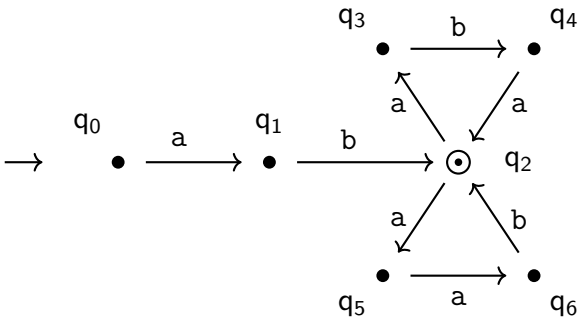
O primeiro passo é decompor a expressão em partes

$$\texttt{ab} \;\Big|\; \left(\texttt{aba} \cup \texttt{aab}\right)^*$$

Daí, a gente vê que cada parte corresponde a um pedaço do diagrama



E daí, decompondo a transição e os laços, nós chegamos a



Só que dessa vez a coisa não deu certo.

Porque existem duas transições com o símbolo a saindo do estado $\texttt{q}_2$.

E agora?

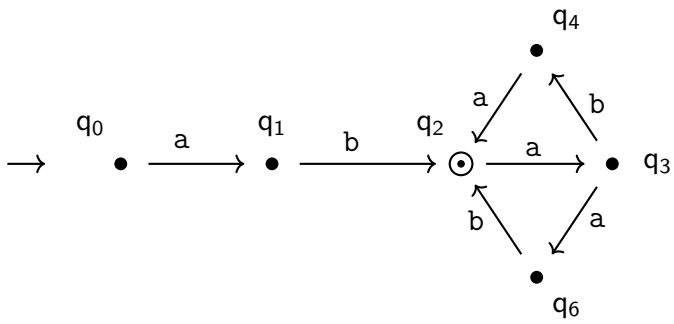
Bom, agora a gente precisa resolver esse problema.

Mas isso não é difícil.

Quer dizer, quando o autômato está em $\texttt{q}_2$ e lê o símbolo a, ele não sabe se deve ir para cima ou para baixo — (*porque ele não sabe se vai ler um* aba *ou um* aab)

Então a ideia é que ele vai pra frente.

E toma a decisão depois



Mas, essa não é a única maneira de fazer as coisas.

Quer dizer, logo no início a gente podia ter reescrito a expressão assim

$$\texttt{ab}\left(\texttt{aba}\ \cup\ \texttt{aab}\right)^* \qquad \Rightarrow \qquad \texttt{ab}\Big(\texttt{a}\left(\texttt{ba}\ \cup\ \texttt{ab}\right)\Big)^*$$

E daí, o raciocínio de construção iria nos levar ao diagrama acima.

Certo.

Mas vamos complicar as coisas um pouquinho mais.

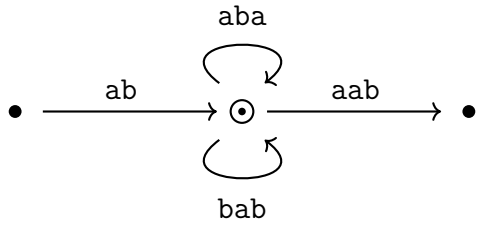
Considere a expressão regular

$$\texttt{ab}\left(\texttt{aba}\ \cup\ \texttt{bab}\right)^*\texttt{aab}$$

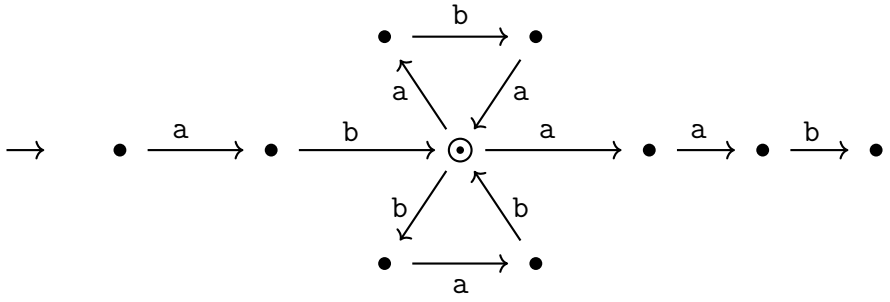
Dessa vez, a gente decompõe a expressão regular em 3 partes

$$\texttt{ab}\ \Big|\ \left(\texttt{aba}\ \cup\ \texttt{bab}\right)^*\ \Big|\ \texttt{aab}$$

Daí, cada parte corresponde a um pedaço do diagrama



E decompondo as transições e os laços, nós chegamos a

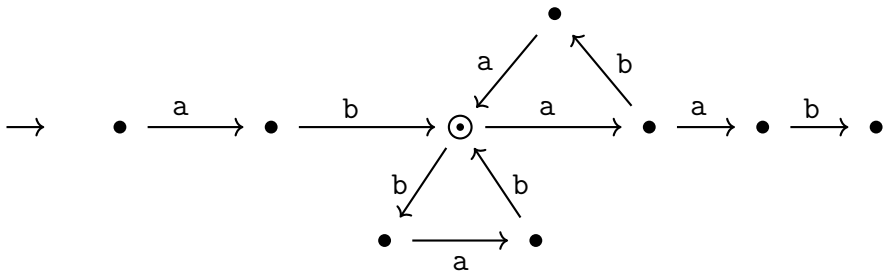


Mais uma vez a coisa não deu certo, porque existem duas transições com o símbolo a saindo do estado q_2

Daí que, quando o autômato está em q_2 e lê o símbolo a, ele não sabe se deve ir para cima ou ir para frente.

E agora?

Bom, a ideia é deixar o autômato ir para frente, e tomar a decisão depois



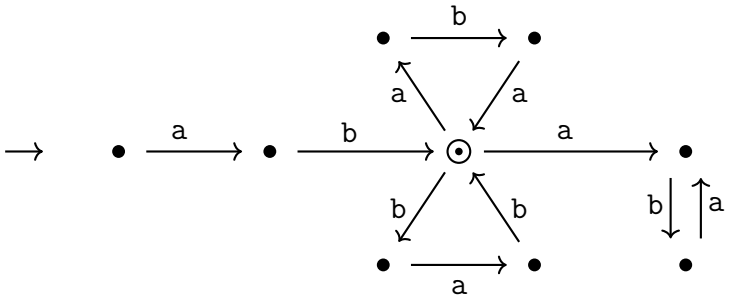
Deu certo.

Mas aqui já dá para advinhar que as coisas podem ficar ainda mais complicadas.

Por exemplo considere a seguinte expressão regular

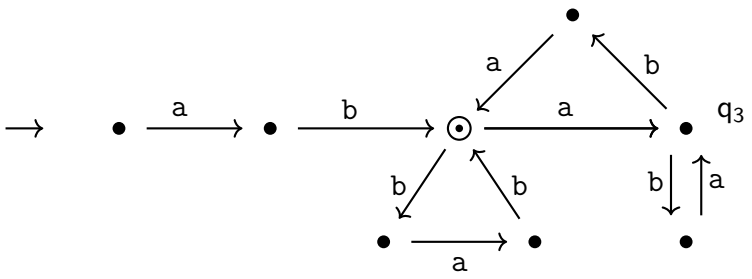
$$\mathtt{ab}\left(\mathtt{aba}\ \cup\ \mathtt{bab}\right)^*\mathtt{a}\left(\mathtt{ba}\right)^*$$

Seguindo o nosso raciocínio padrão, nós chegamos ao diagrama



Como antes, o problema é que existem duas transições com o símbolo a saindo do estado q_2.

Permitindo que o autômato vá para frente com a leitura do a, nós obtemos o diagrama



Mas aqui nós ainda temos um problema: existem duas transições com o símbolo b saindo de q_3.

E agora?

3 Autômatos não determinísticos

Bom, agora nós precisamos entender melhor o que está acontecendo.

E o que está acontecendo, infelizmente, é que uma expressão regular não é um autômato disfarçado.

Quer dizer, o autômato é uma máquina enquanto que a expressão regular é uma descrição.

Qual é a diferença?

Bom, a máquina é um sistema que opera de maneira determinística — (*tic-tac, tic-tac, tic-tac ...*)

E a descrição apenas diz como as coisas podem ser — (*assim ou assado ou assado ...*)

Daí que, em geral é fácil descrever aquilo que uma máquina faz.

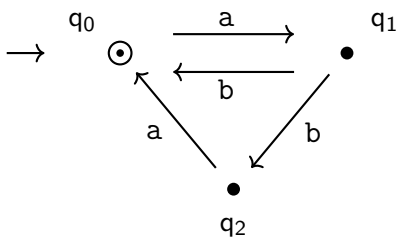
Mas não é tão fácil assim mecanizar aquilo que uma descrição diz.

E agora?

Bom, agora a esperteza é imaginar uma máquina não-determinística — (*tic ou tac - tac ou tic ...*)

Quer dizer, a ideia é que um autômato não-determinístico é um autômato onde pode existir duas (ou mais) transições com o mesmo símbolo saindo de um estado.

Por exemplo,



Certo.

Mas como é que isso funciona?

Bom, a ideia é que o autômato não-determinístico descreve as palavras por meio dos caminhos no diagrama que começam no estado inicial e terminam no estado final.

Por exemplo

$$q_0 \longrightarrow q_1 \longrightarrow q_0 \longrightarrow q_1 \longrightarrow q_2 \longrightarrow q_0 \qquad \Rightarrow \qquad \texttt{ababa}$$

Daí, se alguém nos dá a palavra, a gente pode tentar encontrar o caminho.

Por exemplo considere a palavra

`abaabaab`

Agora, a gente pode raciocinar assim

- o primeiro 'a' nos leva para q_1
- daí, a leitura do 'b' pode nos levar para q_0 ou q_2

- imagine que nós vamos para q_0
    - então, o próximo 'a' nos leva de volta para q_1
    - mas ali a gente não consegue ler o 'a' que vem em seguida ...
- logo, a leitura do 'b' deve nos levar para q_2
- daí, a leitura dos próximos dois 'a's nos leva para q_1
- e ali a gente fica na dúvida outra vez sobre o que fazer com o 'b'
- imagine que ...

Raciocinando dessa maneira, a gente acaba encontrando o caminho

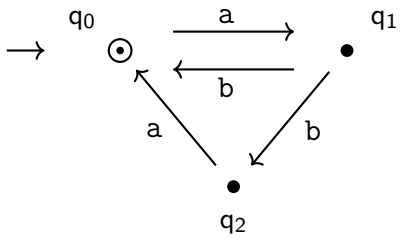
$$\mathtt{abaabaab} \quad \Rightarrow \quad q_0 \to q_1 \to q_2 \to q_0 \to q_1 \to q_2 \to q_0 \to q_1 \to q_0$$

E daí, depois de examinar alguns exemplos desse tipo, a gente acaba descobrindo que o autômato descreve as palavras

$$\big(\,\mathtt{ab} \cup \mathtt{aba}\,\big)^*$$

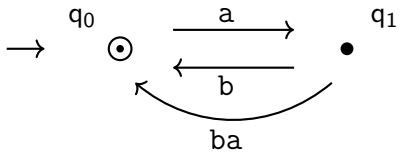
Mas, essa não é a única maneira de fazer as coisas.

Quer dizer, vamos olhar outra vez para o diagrama

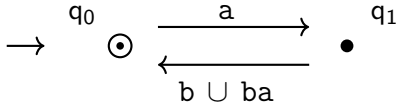


Aqui a gente nota que sempre que o autômato está em q_2 ele precisa ir para q_0.

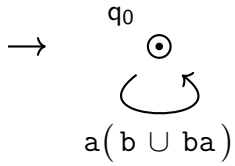
Então, a gente elimina o estado q_2 e reorganiza o diagrama assim



e depois assim



e depois assim



Ora, mas isso já é o mesmo que a expressão regular

$$\Big(\mathtt{a}\big(\mathtt{b} \cup \mathtt{ba} \big) \Big)^*$$

que a gente pode reescrever assim

$$\big(\mathtt{ab} \cup \mathtt{aba} \big)^*$$

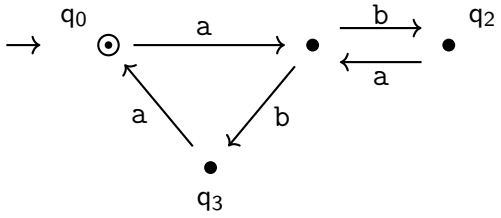
Legal, não é?

Vejamos outros exemplos.

Exemplos

a. Estado de rejeição

Considere o autômato não-determinístico abaixo



Você percebeu que o autômato não-determinístico não tem um estado de rejeição?

Mas isso faz sentido, não é?

Quer dizer, o autômato não-determinístico é uma descrição de um conjunto de palavras — (*como a expressão regular*)

Daí que, ele descreve as palavras que estão no conjunto, mas não precisa descrever as palavras que não estão no conjunto.

Certo.

Mas, como é que a gente descobre que uma palavra não está no conjunto?

Bom, a gente descobre isso quando verifica que não existe caminho no diagrama com a sequência de símbolos da palavra — (*que termina no estado final*)

Por exemplo, considere a palavra

$$\mathtt{abaab}$$

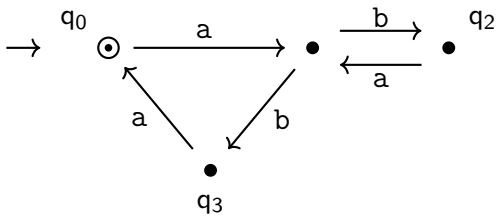
E agora a gente raciocina assim

- o fragmento 'ab' nos leva a q_2 ou q_3
- se nós vamos para q_2,
  então nós não conseguimos ler o 'aa' que vem a seguir
- logo, nós devemos ir para q_3
- mas indo para q_3,
  o fragmento que vem a seguir nos leva para q_2 ou q_3
- só que nem q_2 nem q_3 são estados finais
- logo a palavra não faz parte do conjunto que o autômato descreve ◇

b. Advinhação

Outra maneira de entender os autômatos não-determinísticos consiste em imaginar que eles são uma máquina que advinha — (*o que não é uma coisa que uma máquina pode fazer...*)

Por exemplo, considere outra vez o autômato



E imagine que nós damos a seguinte palavra para o autômato

a b a a b a b a

O autômato aceita ou não aceita?

Bom, vamos raciocinar

- a leitura do 1o 'a' leva o autômato para q_1
- daí, ele advinha que o 'b' deve levá-lo para q_3,
- e a leitura do fragmento 'aa' leva o autômato para q_1 outra vez
- daí, ele advinha que deve dar uma volta em q_2
  para fazer a leitura do fragmento 'ba'
- e depois, ele advinha que deve ir para q_3
  para fazer a leitura do fragmento final 'ba'

Pensando dessa maneira, a gente diz que o autômato não-determinístico aceita a palavra de entrada se existe uma sequência de advinhações que leva ele para o estado final.

E o autômato rejeita a palavra de entrada se não existe nenhuma sequência de advinhações que leva ele para o estado final.

◇

c. Transições vazias

Agora nós vamos aumentar as capacidades de advinhação dos autômatos nào-determinísticos.

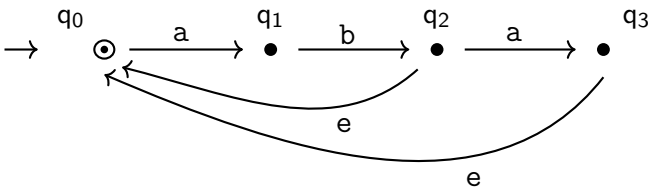
Quer dizer, agora nós vamos permitir que o autômato pule para outro estado (sem ler símbolo nenhum) para continuar a leitura da palavra de entrada por lá.

Para isso, nós introduzimos a noção de transição vazia

$\xrightarrow{e}$

— (*onde o* e *vem do inglês "empty"*)

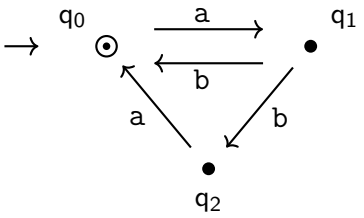
Por exemplo, considere o autômato



Examinando o diagrama, a gente vê que o autômato aceita as palavras

$$\big(\,\texttt{ab} \cup \texttt{aba}\,\big)^*$$

Essas também são as palavras aceitas pelo autômato



Só que é bem mais fácil ver isso no diagrama anterior.

◇

c. Programando o autômato não-determinístico

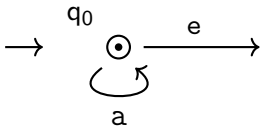
As transições vazias também são ótimas para programar o autômato não-determinítico.

Por exemplo, imagine que nós queremos um autômato que aceita as

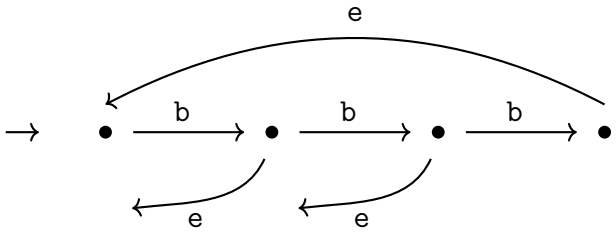
→ *palavras que não possuem blocos de* `b`*'s de tamanho ímpar*

Daí, nós podemos fazer a construção do autômato em etapas.

O primeiro componente faz a leitura dos `a`'s e aguarda o aparecimento de um bloco de `b`'s — (*ele advinha quando os* `b`*'s vão aparecer*)



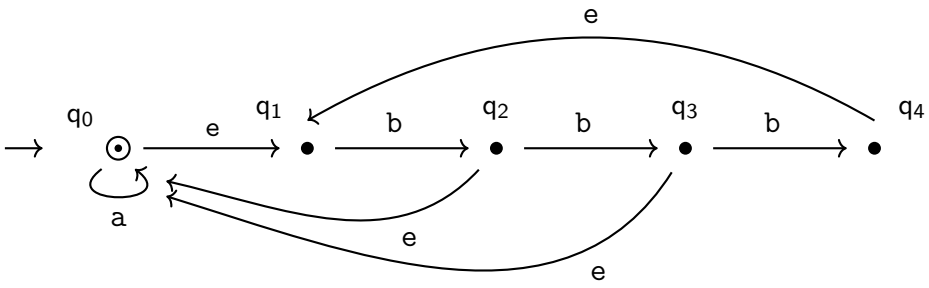
Daí, nós montamos uma arapuca para os blocos de `b`'s com tamanho múltiplo de 3



Quer dizer, os blocos de tamanho múltiplo de 3 ficam presos na arapuca e não conseguem sair.

Mas existem duas saídas por baixo para os blocos de outros tamanhos.

Finalmente, basta juntar uma coisa com a outra

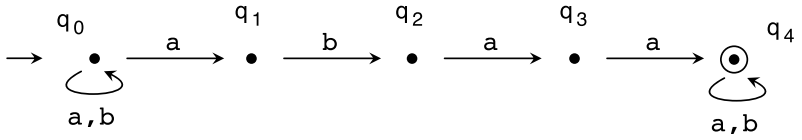


◇

d. Reconhecimento de padrões

Uma tarefa bem simples para os autômatos não-determinísticos é encontrar um padrão na palavra de entrada.

Por exemplo,



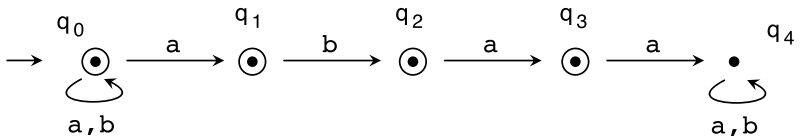
Note que apenas as palavras que contém o padrão `abaa` conseguem fazer a travessia do estado inicial q_0 para o estado final q_4.

◇

e. Quando o não-determinismo não ajuda

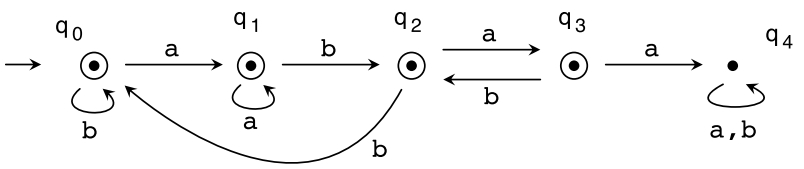
Por outro lado, o não-determinismo não ajuda em nada a verificar que o padrão <u>não</u> aparece na palavra de entrada.

À primeira vista, alguém poderia imaginar que bastaria inverter a marcação de estados finais no autômato do exemplo anterior.



Mas, note que esse autômato aceita a palavra `abaa`: basta ele ficar no estado q_0 e ler todos os símbolos sem sair de lá.

De fato, a melhor maneira de realizar essa tarefa consiste em operar deterministicamente



Dessa maneira, o autômato é forçado a seguir pelo caminho até q_4, caso o padrão `abaa` apareça.

◇