Algoritmos e Fundamentos da Teoria de Computação

Lista de Exercícios 00a

1 Prove, utilizando indução sobre o comprimento da *string*, que $(w^R)^R = w$, para todas as *strings* $w \in \Sigma^*$.

Segue a prova por indução da propriedade pedida.

Base: O caso base é a *string* nula. Neste caso, é trivial ver que $(\lambda^R)^R = (\lambda)^R = \lambda$, como queríamos.
Hipótese Indutiva (HI): Assuma que $(w^R)^R = w$ vale para todas as *strings* $w \in \Sigma^*$ de tamanho n ou menor.

Passo indutivo: Seja w uma *string* de comprimento $n + 1$. Então temos que $w = ua$, aonde u é uma *substring* de tamanho n e $a \in \Sigma$ é um elemento arbitrário do alfabeto. (O termo arbitrário quer dizer que qualquer elemento de Σ serve, e vamos usar o símbolo a para representar o elemento escolhido.) A prova segue com os passos e justificativas abaixo.

$$
\begin{aligned}
(w^R)^R &= ((ua)^R)^R && \text{(Definição de } w\text{)} \\
&= (a^R u^R)^R && \text{(Teorema 2.1.6)} \\
&= (au^R)^R && \text{(Identidade de } R\text{)} \\
&= (u^R)^R a^R && \text{(Teorema 2.1.6)} \\
&= ua^R && \text{(HI)} \\
&= ua && \text{(Identidade de } R\text{)} \\
&= w && \text{(Definição de } w\text{)}
\end{aligned}
$$

2 Apresente uma definição recursiva para o conjunto de *strings* sobre $\{a, b\}$ que contêm duas vezes mais a's do que b's.

Seja L a linguagem formada pelo conjunto pedido. Podemos definir L de forma recursiva como a seguir.

Base: $\lambda \in \mathsf{L}$.
Passo recursivo: Se $u \in \mathsf{L}$ e u pode ser escrito como $xyzw$, onde $x, y, z, w \in \Sigma^*$, então:

i) $xayazbw \in \mathsf{L}$,
ii) $xaybzaw \in \mathsf{L}$, e
iii) $xbyazaw \in \mathsf{L}$.

Fecho: Uma *string* u pertence a L se, e somente se, u pode ser obtida a partir de λ através de um número finito de aplicações do passo recursivo.

É fácil perceber que qualquer *string* gerada pela definição acima possui duas vezes mais a's do que b's. Isto acontece por conta do passo recursivo, aonde garantimos que cada b introduzido também leva à introdução de dois a's. Se você quiser se convencer disto de uma forma mais precisa, desenvolva uma prova por indução sobre o tamanho das *strings* para mostrar que todas elas satisfazem a propriedade pedida.

3 Seja $\mathsf{L}_1 = \{aaa\}^*$, $\mathsf{L}_2 = \{a, b\}\{a, b\}\{a, b\}\{a, b\}$, e $\mathsf{L}_3 = \mathsf{L}_2^*$. Descreva as *strings* que pertencem às linguagens L_2, L_3, e $\mathsf{L}_1 \cap \mathsf{L}_3$.

A linguagem L_2 é formada por todas as *strings* sobre $\{a, b\}$ com comprimento quatro. De forma similar, a linguagem L_3, que é obtida pela aplicação do fecho de Kleene $(^*)$ em L_2, contém todas as *strings* cujo comprimento é *divisível* por quatro. Note que a *string* nula também pertence a L_3.

A linguagem $\mathsf{L}_1 \cap \mathsf{L}_3$ contém todas as *strings* que estão simultaneamente em L_1 e L_3. Para pertencer a L_1, toda *string* precisa ser formada somente por a's e ter um comprimento divisível por três. Como L_3 exige um comprimento divisível por quatro, uma *string* em $\mathsf{L}_1 \cap \mathsf{L}_3$ deve satisfazer a todas estas condições ao mesmo tempo. Logo, temos que $\mathsf{L}_1 \cap \mathsf{L}_3 = (a^{12})^*$, isto é, a linguagem formada por *strings* que só contém a's e possuem comprimento divisível por 12.

4 Apresente a expressão regular que representa o conjunto de *strings* sobre $\{a, b, c\}$ que começam com a, contêm exatamente dois b's, e terminam com cc.

A expressão abaixo atende às condições do enunciado:

$$\boldsymbol{a(a \cup c)^* b(a \cup c)^* b(a \cup c)^* cc} \quad .$$

O a inicial e o cc final foram colocados explicitamente na expressão. Qualquer número de a's e c's, representados pela expressão $(a \cup c)^*$, pode cercar os dois b's da *string*.

5 Utilize as identidades de expressões regulares da Tabela 2.1 (página 54 do livro do Sudkamp) e outras definições do capítulo para estabelecer a seguinte relação:

$$\boldsymbol{(ba)^+(a^*b^* \cup a^*) = (ba)^*ba^+b^*} \quad .$$

$$\begin{aligned}
\boldsymbol{(ba)^+(a^*b^* \cup a^*)} &= \boldsymbol{(ba)^+(a^*)(b^* \cup \lambda)} && \text{(Identidade 9)} \\
&= \boldsymbol{(ba)^*ba(a^*)(b^* \cup \lambda)} && \text{(Definição do fecho positivo } ^+\text{)} \\
&= \boldsymbol{(ba)^*ba^+(b^* \cup \lambda)} && \text{(Definição do fecho positivo } ^+\text{)} \\
&= \boldsymbol{(ba)^*ba^+b^*} && (b^* \text{ já contém } \lambda)
\end{aligned}$$

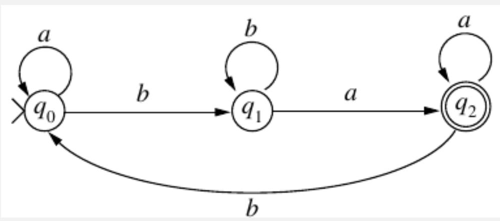
6 Seja M o autômato finito determinístico definido por

$$\begin{aligned} Q &= \{q_0, q_1, q_2\} \\ \Sigma &= \{a, b\} \\ F &= \{q_2\} \end{aligned}$$

δ	a	b
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_2	q_0

a. Apresente o diagrama de estados de M.
b. Faça o *trace* das computações de M para as *strings* $abaa$, $bbbabb$, $bababa$, e $bbbaa$.
c. Quais das *strings* do item (b) pertencem a $\mathsf{L}(\mathsf{M})$?

a. O diagrama de estados de M é



b.

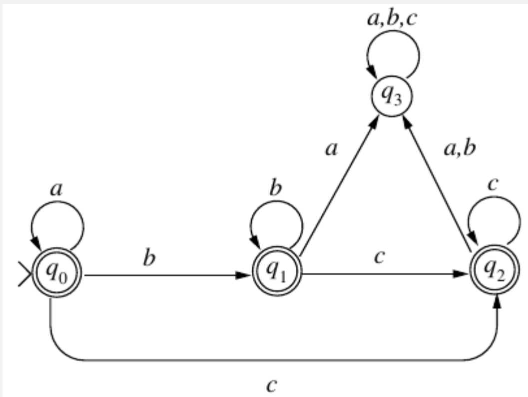
$$
\begin{array}{llll}
\text{i)} & [q_0, abaa] & \text{ii)} & [q_0, bbbabb] \\
\vdash & [q_0, baa] & \vdash & [q_1, bbabb] \\
\vdash & [q_1, aa] & \vdash & [q_1, babb] \\
\vdash & [q_2, a] & \vdash & [q_1, abb] \\
\vdash & [q_2, \lambda] & \vdash & [q_2, bb] \\
 & & \vdash & [q_0, b] \\
 & & \vdash & [q_1, \lambda]
\end{array}
$$

$$
\begin{array}{llll}
\text{iii)} & [q_0, bababa] & \text{iv)} & [q_0, bbbaa] \\
\vdash & [q_1, ababa] & \vdash & [q_1, bbaa] \\
\vdash & [q_2, baba] & \vdash & [q_1, baa] \\
\vdash & [q_0, aba] & \vdash & [q_1, aa] \\
\vdash & [q_0, ba] & \vdash & [q_2, a] \\
\vdash & [q_1, a] & \vdash & [q_2, \lambda] \\
\vdash & [q_2, \lambda] & &
\end{array}
$$

c. As computações em (i), (iii) e (iv) terminam no estado de aceite q_2. Portando, as *strings* $abaa$, $bababa$, e $bbbaa$ pertencem a $\mathsf{L(M)}$.

7 Construa um autômato finito determinístico (DFA) que aceita a linguagem formada pelas *strings* sobre $\{a, b, c\}$ aonde todos os a's precedem os b's, que por sua vez precedem os c's. É possível que não hajam a's, b's, ou c's.

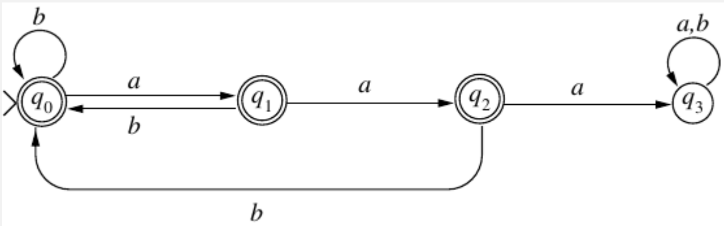
O DFA abaixo aceita a linguagem correspondente a $\boldsymbol{a^*b^*c^*}$.



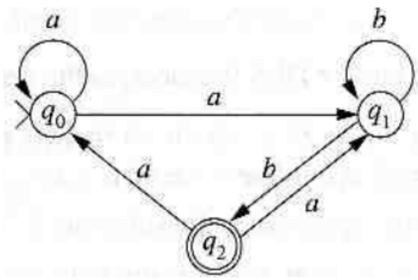
O autômato usa os *loops* nos estados q_0, q_1, e q_2 para ler a's, b's, e c'c, respectivamente. Qualquer desvio desta ordem faz a computação entrar em q_3 e rejeitar a *string*.

8 Construa um DFA que aceita a linguagem formada pelas *strings* sobre $\{a, b\}$ que não contêm a *substring* aaa.

O DFA abaixo aceita as *strings* pedidas no enunciado. Os estados q_0, q_1, e q_2 são usados para contar o número de a's consecutivos que foram processados. Caso três a's seguidos forem encontrados, o DFA entra no estado q_3, consome o restante da entrada, e rejeita a *string*.



9 Seja M o autômato finito não-determinístico (NFA) abaixo:



a. Construa a tabela de transições de M.
b. Faça o *trace* de todas as computações possíveis de M para a *string* $aaabb$.
c. A *string* $aaabb$ pertence a $\mathsf{L}(\mathsf{M})$?
d. Apresente uma expressão regular para $\mathsf{L}(\mathsf{M})$.

a. A tabela de transições de M é dada abaixo.

δ	a	b
q_0	$\{q_0, q_1\}$	$\emptyset$
q_1	$\emptyset$	$\{q_1, q_2\}$
q_2	$\{q_0, q_1\}$	$\emptyset$

b.

$$\begin{aligned} & [q_0, aaabb] \\ \vdash\ & [q_1, aabb] \end{aligned}$$

$$\begin{aligned} & [q_0, aaabb] \\ \vdash\ & [q_0, aabb] \\ \vdash\ & [q_1, abb] \end{aligned}$$

$$\begin{aligned} & [q_0, aaabb] \\ \vdash\ & [q_0, aabb] \\ \vdash\ & [q_0, abb] \\ \vdash\ & [q_1, bb] \\ \vdash\ & [q_1, b] \\ \vdash\ & [q_1, \lambda] \end{aligned}$$

$$\begin{aligned} & [q_0, aaabb] \\ \vdash\ & [q_0, aabb] \\ \vdash\ & [q_0, abb] \\ \vdash\ & [q_0, bb] \end{aligned}$$

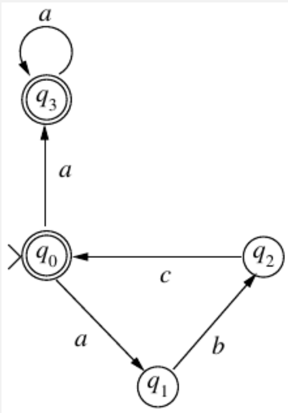
$$\begin{aligned} & [q_0, aaabb] \\ \vdash\ & [q_0, aabb] \\ \vdash\ & [q_0, abb] \\ \vdash\ & [q_1, bb] \\ \vdash\ & [q_1, b] \\ \vdash\ & [q_2, \lambda] \end{aligned}$$

c. Sim. A *string* $aaabb$ pertence a $\mathsf{L(M)}$, pois a última das possíveis computações de M acima lê a *string* toda e para no estado de aceite q_2.

d. A expressão regular para $\mathsf{L(M)}$ é $\boldsymbol{a^+b^+(ab^+ \cup a^+ab^+)^*}$. O processamento de $\boldsymbol{a^+b^+}$ deixa a computação no estado de aceite q_2. Uma vez que a máquina entrou em q_2 há duas formas de se sair e depois retornar a este estado, tomando o ciclo q_0, q_1, q_2, ou o ciclo q_1, q_2. O primeiro caminho processa a *string* $\boldsymbol{a^+ab^+}$ e o segundo $\boldsymbol{ab^+}$. Como podemos escolher entre qualquer um dos ciclos, temos que uma única saída e retorno para q_2 é representada por $\boldsymbol{ab^+ \cup a^+ab^+}$. Como é possível sair e retornar a q_2 várias vezes, temos $\boldsymbol{(ab^+ \cup a^+ab^+)^*}$. Esta expressão pode ser simplificada para $\boldsymbol{(a^+b^+)^*}$. Logo, a linguagem de M se reduz a $\mathsf{L(M)} = \boldsymbol{a^+b^+(ab^+ \cup a^+ab^+)^*} = \boldsymbol{a^+b^+(a^+b^+)^*} = \boldsymbol{(a^+b^+)^+}$.

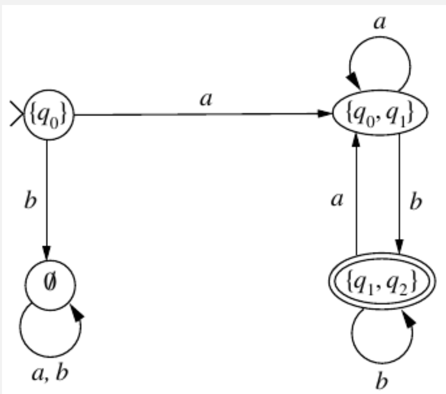
10 Desenhe o diagrama de estados de um NFA que aceita a linguagem $\boldsymbol{(abc)^*a^*}$.

O NFA abaixo aceita a linguagem $\boldsymbol{(abc)^*a^*}$. *Strings* da forma $\boldsymbol{(abc)^*}$ são aceitas em q_0. O estado q_3 aceita $\boldsymbol{(abc)^*a^+}$.



11 Construa o diagrama de estados do DFA equivalente ao NFA do exercício 9. *(Obs.: Veja o algoritmo 5.6.3 na página 172 do livro.)*

O diagrama de estados do DFA equivalente ao NFA M do exercício 9 é dado abaixo.



Como M não possui transições λ, a função de transição de entrada utilizada para o algoritmo 5.6.3 já é a própria função de transição de M, apresentada na solução do exercício 9.

Os nós do DFA equivalente são construídos no passo 2 do algoritmo. A geração do conjunto Q' dos estados do DFA pode ser vista no *trace* da tabela abaixo. O estado inicial do DFA é $\{q_0\}$. O algoritmo procede escolhendo um estado $X \in Q'$, e um símbolo $a \in \Sigma$ para o qual já não haja um arco saindo de X. O conjunto Y consiste dos estados que podem ser alcançados pelo processamento de um a a partir do estado X.

X	input	Y	Q′
			$Q' := \{\{q_0\}\}$
$\{q_0\}$	a	$\{q_0, q_1\}$	$Q' := Q' \cup \{\{q_0, q_1\}\}$
$\{q_0\}$	b	$\emptyset$	$Q' := Q' \cup \{\emptyset\}$
$\{q_0, q_1\}$	a	$\{q_0, q_1\}$	
$\{q_0, q_1\}$	b	$\{q_1, q_2\}$	$Q' := Q' \cup \{\{q_1, q_2\}\}$
$\emptyset$	a	$\emptyset$	
$\emptyset$	b	$\emptyset$	
$\{q_1, q_2\}$	a	$\{q_0, q_1\}$	
$\{q_1, q_2\}$	b	$\{q_1, q_2\}$	

Para determinar os estados de aceite do DFA, basta selecionar todos aqueles que contêm o estado final q_2 de M.