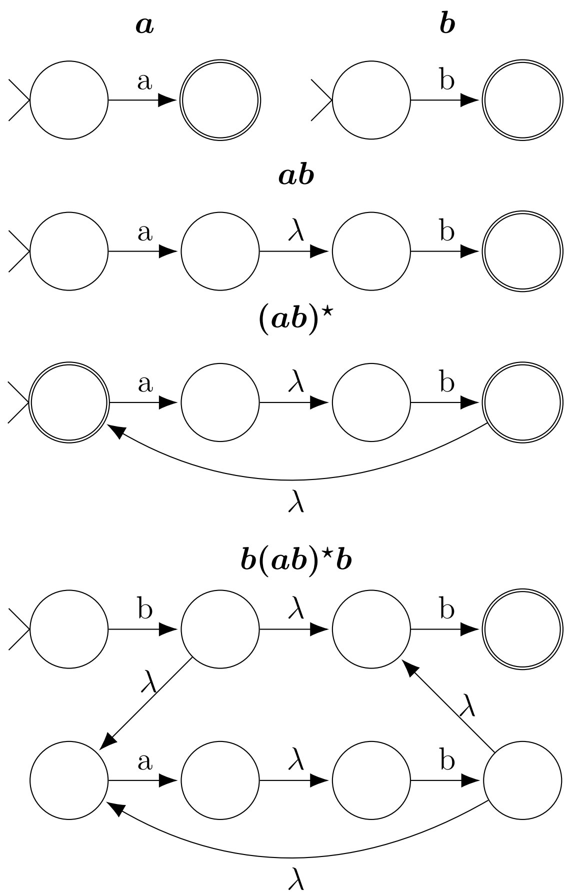
CS 3133: Homework 4

Adam Camilli (aocamilli@wpi.edu)

October 2, 2017

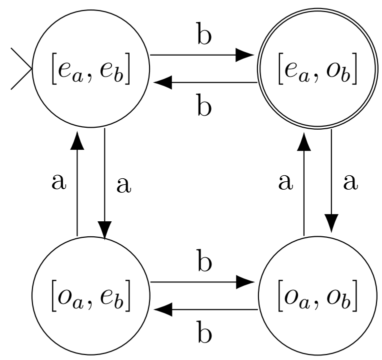
1. Use the technique from Section 6.1 in the book to build the state diagram of an NFA-λ that accepts the language $\boldsymbol{b(ab)^\star b}$.

   To build this language using the recursive definition of regular languages in the manner described in Section 6.1, we can build the language $b(ab)^\star b$ using the singleton sets $\{a\}$ and $\{b\}$. The only tricky part is that we are not interested in only $\{ab\}$ but $\{ab\}^\star$. This is easily represented, however, with a looping λ-transition:



2. **6.3** (p. 217) The language of the DFA M in Example 5.3.7 consists of all strings over $\{a, b\}$ with an even number of a's and an odd number of b's. Use Algorithm 6.2.2 to construct a regular expression L(M). Exercise 2.38 requested a nonalgorithmic construction of a regular expression for this language, which, as you now see, is a formidable task.

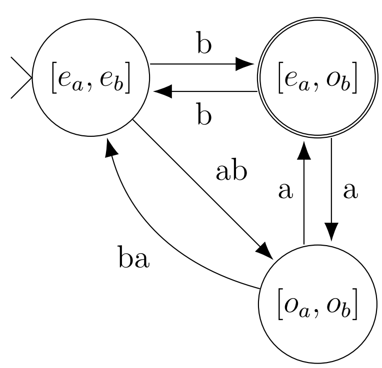
M:



We begin by deleting state $[o_a, e_b]$, or q_1. Following the algorithm, we obtain the following replacement transitions:

$q_0 \to q_1 \to q_2$	$q_0 \xrightarrow{ab} q_2$
$q_2 \to q_1 \to q_0$	$q_2 \xrightarrow{ba} q_0$
$q_0 \to q_1 \to q_0$	$q_0 \xrightarrow{aa} q_0$
$q_2 \to q_1 \to q_2$	$q_2 \xrightarrow{bb} q_2$

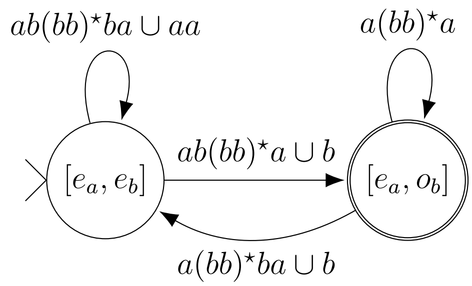
This gives the following DFA:



Now delete state $[o_a, o_b]$, or q_2:

$q_0 \to q_1 \to q_2$	$q_0 \xrightarrow{ab(bb)^\star a \cup b} q_2$
$q_2 \to q_1 \to q_0$	$q_2 \xrightarrow{a(bb)^\star ba \cup b} q_0$
$q_0 \to q_1 \to q_0$	$q_0 \xrightarrow{ab(bb)^\star ba \cup aa} q_0$
$q_2 \to q_1 \to q_2$	$q_2 \xrightarrow{a(bb)^\star a} q_2$

which gives the two state DFA



and at last allows us to construct an expression L(M) which accepts the language using the rule for a two-state DFA where $q_0 \neq q_f$.

$$
\begin{aligned}
u &= ab(bb)^\star ba \cup aa \\
v &= ab(bb)^\star a \cup b \\
w &= a(bb)^\star a \\
x &= a(bb)^\star ba \cup b
\end{aligned}
$$

$$\mathrm{L(M)} \;= u^\star v\big(w \cup x u^\star v^\star\big)^\star$$

3. **6.4** (p. 217) Let G be the grammar

$$\begin{aligned} G : S \to &\ aS \mid bA \mid a \\ A \to &\ aS \mid bA \mid b. \end{aligned}$$

(a) Use Theorem 6.3.1 to build an NFA M that accepts L(G).

Per the theorem, we define our new NFA-λ as having states Q such that

$$Q = \{S, A, Z\}$$

where Z contains all terminal productions, which here are

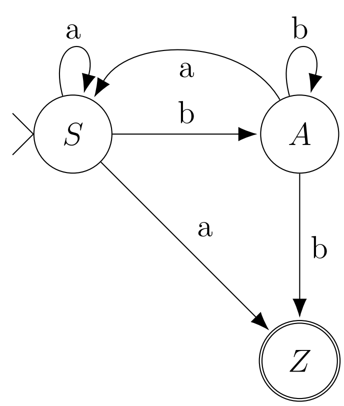
$$S \to a$$

$$A \to b$$

We now replace all productions of the form $A \to aB$ with a transition $\delta(A, a) = B$:

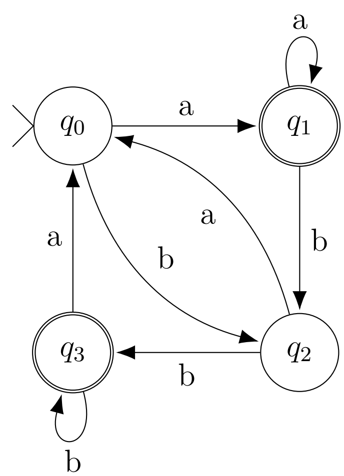
$$\begin{aligned} S \to aS &\Rightarrow \delta(S, a) = S \\ S \to bA &\Rightarrow \delta(S, b) = A \\ A \to aB &\Rightarrow \delta(A, a) = S \\ A \to bA &\Rightarrow \delta(A, b) = A \end{aligned}$$

Finally, we can construct the NFA with states S, A, Z where Z is our accepting state:



(b) Using the result of part (a), build a DFA M′ that accepts L(G). Using $q_0 = S, q_1 = \{S, Z\}, q_2 = A, q_3 = \{A, Z\}$:

	a	b
q_0	q_1	q_2
q_1	q_1	q_2
q_2	q_0	q_3
q_3	q_0	q_3



(c) Construct a regular grammar from M that generates L(M).

$$\begin{aligned} S &\to aS|aZ|bA \\ A &\to bA|bZ|aS \\ Z &\to \lambda \end{aligned}$$

(d) Construct a regular grammar from M′ that generates L(M′).

$$\begin{aligned} S &\to aA|a \\ A &\to aA|bB|a \\ B &\to aS|bZ|b \\ Z &\to bZ|aS|b \end{aligned}$$

(e) Give a regular expression for L(G).

$$(a^\star bb^\star a)^\star a^+ \cup (a^\star bb^\star a)^\star bb^+$$

4. **6.14.d** (p. 218)

   (a) Assume indirectly that $\text{L} = \{ww | w \in \{a, b\}^\star\}$ is regular. Therefore there must exist a DFA with k states that represents L such that $k > 0$.

   (b) Setting $w = a^k b$, we must be able to create a partition $xyz = ww = a^k b a^k b$, where $|xy| = k$ and $|y| > 0$, and $xy^i z \in \text{L}$ for all $i \geq 0$.

   (c) We therefore set $x = a \dots a, y = a \dots a, z = ba^k b$ such that $|xy| \leq k$ and $xy^i z \in \text{L}$ for all $i \geq 0$.

   (d) Testing $i = 0$, we obtain
   $$xy^0 z = a^{k-|y|} b a^k b$$
   which generates a contradiction: Since the Pumping Lemma requires $|y| > 0$, the first "half" $xy = w = a^{k-|y|} b$ of ww must contain fewer a's than the second "half" $z = w = a^k b$. Thus L cannot be regular by the Pumping Lemma.

QED

5. **7.1** (p. 247) Let M be the PDA defined by

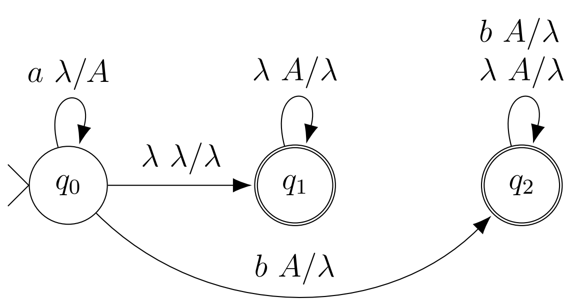
$$
\begin{aligned}
Q &= \{q_0, q_1, q_2\} & \delta(q_0, a, \lambda) &= \{[q_0, A]\} \\
\Sigma &= \{a, b\} & \delta(q_0, \lambda, \lambda) &= \{[q_1, \lambda]\} \\
\Gamma &= \{A\} & \delta(q_0, b, A) &= \{[q_2, \lambda]\} \\
F &= \{q_1, q_2\} & \delta(q_0, \lambda, A) &= \{[q_1, \lambda]\} \\
& & \delta(q_2, b, A) &= \{[q_2, \lambda]\} \\
& & \delta(q_2, \lambda, A) &= \{[q_2, \lambda]\}
\end{aligned}
$$

   (a) Describe the language accepted by M.

   M accepts the language $\{a^i b^j \mid 0 \geq i \geq j\}$. Each a pushes A onto the stack, and each b pops A. Strings with greater numbers of b than a inevitably halt before emptying the stack or will be stuck with no valid transitions in the case of invalid input such as aba.

   (b) Give the state diagram of M.

   The state diagram of M is



(c) Trace all computations of the strings aab, abb, aba in M.

$$\begin{array}{lll}
\quad [q_0, aab, \lambda] & \quad [q_0, abb, \lambda] & \quad [q_0, aba, \lambda] \\
\vdash [q_0, ab, A] & \vdash [q_0, bb, A] & \vdash [q_0, ba, A] \\
\vdash [q_0, b, AA] & \vdash [q_2, b, \lambda]\ \mathbf{X}\ \text{(Reject)} & \vdash [q_2, a, \lambda]\ \mathbf{X}\ \text{(Reject)} \\
\vdash [q_2, \lambda, A] & & \\
\vdash [q_2, \lambda, \lambda]\ \checkmark\ \text{(Accept)} & &
\end{array}$$

(d) Show that $aabb, aaab \in \mathrm{L(M)}$.

To show this, we simply trace their computations:

$$\begin{array}{ll}
\quad [q_0, aabb, \lambda] & \quad [q_0, aaab, \lambda] \\
\vdash [q_0, abb, A] & \vdash [q_0, aab, A] \\
\vdash [q_0, bb, AA] & \vdash [q_0, ab, AA] \\
\vdash [q_2, b, A] & \vdash [q_0, b, AAA] \\
\vdash [q_2, \lambda, \lambda]\ \checkmark\ \text{(Accept)} & \vdash [q_2, \lambda, AA] \\
 & \vdash [q_2, \lambda, A] \\
 & \vdash [q_2, \lambda, \lambda]\ \checkmark\ \text{(Accept)}
\end{array}$$