CIS 511 Homework 1

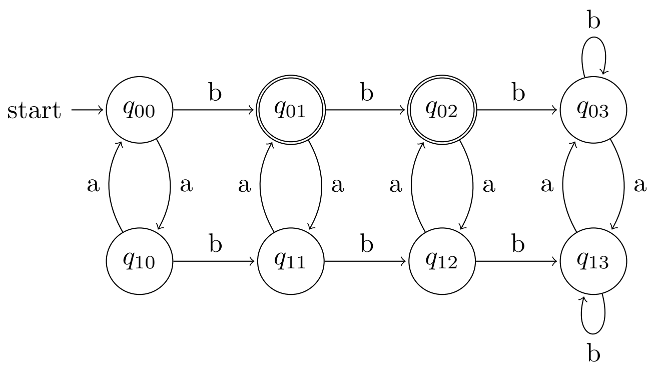
Stephen Phillips, Daegan Golomb

January 25, 2015

Problem 1

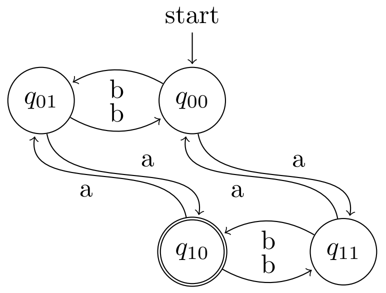
(c)

Here the states q_{ij} mean we have see i (mod 2) a symbols and j b symbols (until 3 where we stop counting).



(g)

Here the states q_{ij} mean we have see i (mod 2) total symbols and j (mod 2) a symbols. The top row is reversed to make the transitions more clear.



Problem 2

(a)

Using the method we did in class, we need to find R^k_{ij} for $i, j \in \{0, 1, 2\}, k \in \{1, 2\}$. The general formula is $R^{k+1}_{i,j} = R^k_{i,j} \cup (R^k_{i,k+1}(R^k_{k+1,k+1})^*)R^k_{k+1,j}$

$$\begin{aligned}
R^0_{11} &= a \\
R^0_{12} &= b \\
R^0_{21} &= b \\
R^0_{22} &= a \\
R^1_{11} &= R^0_{11} \cup R^0_{11}R^{0*}_{11}R^0_{11} = a \cup (aa^*a) \\
R^1_{12} &= R^0_{12} \cup R^0_{11}R^{0*}_{11}R^0_{12} = b \cup (aa^*b) \\
R^1_{21} &= R^0_{21} \cup R^0_{21}R^{0*}_{11}R^0_{11} = b \cup (ba^*a) \\
R^1_{22} &= R^0_{22} \cup R^0_{21}R^{0*}_{11}R^0_{12} = a \cup (ba^*b) \\
R^2_{12} &= R^1_{12} \cup R^1_{12}R^{1*}_{22}R^1_{22} = (b \cup aa^*b) \cup ((b \cup (aa^*b))(a \cup ba^*b)^*(a \cup (ba^*b)))
\end{aligned}$$

So the final regular expression is $R^2_{12} = (b \cup aa^*b) \cup ((b \cup (aa^*b))(a \cup ba^*b)^*(a \cup (ba^*b)))$. The simpler regular expression is $a^*ba^*(ba^*ba^*)^*$ (seen from inspection), which this matches.

(b)

We again use the same technique, this time in tabular form. The general formula is, again, $R^{k+1}_{i,j} = R^k_{i,j} \cup (R^k_{i,k+1}(R^k_{k+1,k+1})^*)R^k_{k+1,j}$

Reg. Exp./Level	0	1	2	3
R_{11}	ε	ε	ε	$(a \cup b)a^*b(ba^*b)^*a$
R_{12}	$a \cup b$	$a \cup b$	$(a \cup b) \cup (a \cup b)a^*a$	(Not ending on final state)
R_{13}	$\varnothing$	ε	$(a \cup b)a^*b$	$(a \cup b)a^*b(ba^*b)^*ba^*b$
R_{21}	$\varnothing$	$\varnothing$	$\varnothing$	(Not starting at initial state)
R_{22}	a	a	$a \cup aa^*a$	(Not starting at initial state)
R_{23}	b	b	$b \cup aa^*b$	(Not starting at initial state)
R_{31}	a	a	a	(Not starting at initial state)
R_{32}	b	$b \cup (b(a \cup b))$	$b \cup b(a \cup b) \cup ((b \cup b(a \cup b))a^*a$	(Not starting at initial state)
R_{33}	ε	ε	$b(a)^*b$	(Not starting at initial state)

So the final expression is $R^3_{11} \cup R^3_{13} = ((a \cup b)a^*b(ba^*b)^*a) \cup ((a \cup b)a^*b(ba^*b)^*ba^*b)$. This DFA does not have any intuitive regular expression that can be seen from inspection, so this technique is the best way to do it.

Problem 3

Claim: If a language A is regular, then its reverse $A^{\mathcal{R}}$ is regular. Proof: Let $M = (Q, \Sigma, \delta, q_0, F)$ be the DFA accepting the language A. Create the NFA $N = (Q', \Sigma, \delta', q'_0, F')$. The idea is to make N such that it reverses all the edges of M, and create a new start state that has epsilon transitions to the final states of M, with the accepting state being the starting state of M. Specifically:

$$\begin{aligned}
Q' &= Q \cup \{q'_0\} \\
\delta(q \in Q, a \in \Sigma) &= \{q' \in Q | \delta(q', a) = q\} \\
\delta(q'_0, \epsilon) &= F \\
F' &= \{q_0\}
\end{aligned}$$

All transitions not specified map to the empty set. Now we have to show this machine accepts a string if and only if the string is in $A^{\mathcal{R}}$.

Proving that it accepts if the string is in $A^{\mathcal{R}}$ is rather simple. If a string $s \in A^{\mathcal{R}}$ then there is a path from one of the final paths to the initial state of M, since the reverse of the string would have been accepted by M. Therefore our machine N accepts this string.

Now we must prove it does not accept any other strings. Suppose not, and there was a string $s \notin A^{\mathcal{R}}$ that was accepted by this machine. Since it accepted, there must have been at least one path to the initial state of M. However, we started at all the final states of M, since our initial state q_0' only had epsilon transitions to the final states of M, and went to nothing else. So that means there is a path from a final state to the initial state. There cannot be more than one of these paths because M was a DFA. This means that the reverse of this string would have been accepted M, which is a contradiction since we assumed that $s \notin A^{\mathcal{R}}$.

We have proved this machine N accepts only strings in $A^{\mathcal{R}}$. Since any NFA can be converted to a DFA, that means $A^{\mathcal{R}}$ is regular for any regular language A.

Problem 4

So an all-NFA M is a five-tuple $(\Sigma, Q, \delta, q_0, F)$, that acts just like an NFA except it only accepts if all non-deterministic paths end up in states in F. We will show that this recognizes the set of regular languages.

In one direction, any DFA is an all-NFA. Since in a DFA's δ only maps a state to only one other state, we only ever have one path, and therefore if a string gives an accepting path, the final state it ends up in is the only final state it ends up in (you know, determinism).

In the other direction, we can create a DFA for any all-NFA. It is similar to the construction of DFA's for normal NFAs. So, given an all-NFA $M = (\Sigma, Q_M, \delta_M, q_{M0}, F_M)$, we can construct DFA $N = (\Sigma, Q_N, \delta_N, q_{N0}, F_N)$. For completeness, let the function $\mathcal{N}_\varepsilon(S)$ denote the 'epsilon closure' as described in the book - the set of all nodes in S and reachable by S via epsilon transitions.

$$\begin{aligned}
Q_N &= \mathbb{P}(Q_M) \\
\delta_M(q \in Q_N, a \in \Sigma) &= \mathcal{N}_\varepsilon(\{q' \in Q_M | q' \in \delta(q, a)\}) \\
q_{N0} &= \{q_{M0}\} \\
F_N &= \{S \subseteq\}
\end{aligned}$$