# B501 Assignment 3

## Due Date: Wednesday, February 29, 2012
## Due Time: 11:00pm

1. (10 points) Prove that the following languages are not regular. You may use the pumping lemma and the closure of the class of regular languages under union, intersection, and complement.

   (a) $\{0^n 1^m 0^n | m, n \geq 0\}$

   **Solution:** The proof is by contradiction using the pumping lemma. Let $p > 0$ (pumping length), $q \geq 0$ and $w = 0^p 1^q 0^p \in L$. Then it is the case that $|w| = 2p + q > p$. Using the condition that $|xy| \leq p$, it follows that $xy$ (and $y$) consists of only 0s. Choose any $i$, say $i = 2$, then $xyyz \notin L$ (there will be more zeros in the left side on the string). Thus, the language is not regular.

   (b) $\{wtw | w, t \in \{0,1\}^+\}$

   **Solution:** The proof is by contradiction using the pumping lemma. Let $p > 0$ (pumping length) and $w = 0^p 1 0^p \in L$. Then is is the case that $|w| = 2p + 1 > p$. Using the condition that $|xy| \leq p$, it follows that $xy$ (and $y$) consists of only 0s. Choose any $i$, say $i = 2$, then $xyyz \notin L$ (there will be more zeros in the left side on the string). Thus, the language is not regular.

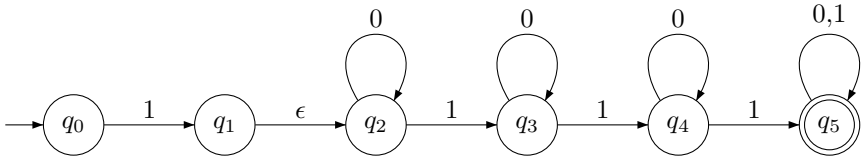
2. (10 points) Let $B = \{1^k y | y \in \{0,1\}^*$ and $y$ contains at least $k$ 1s, for $k > 1\}$. Show that $B$ is a regular language.

   **Solution:**
   To prove that $B$ is regular, we can construct an automaton that accepts $B$. But before doing that, it may be useful to work with an equivalent definition of $B$ that is easier to map to an automaton.

   $1^k y = 1 1^{k-1} y$ Taking one 1 out of $1^k$
   Let $x = 1^{k-1} y$, then $1^k y = 1x$

   Now we just need to construct the machine $1x$, which we can do easily via the closure properties of the class of regular languages, particulary the concatenation of 1 and $x$. To be able to do this, we need to know what $x$ represent: $x = 1^{k-1} y$, where $y \in \{0,1\}^*$ and $y$ contains at least $k$ 1s, for $k > 1$. Therefore, $x$ contains at least $k - 1 + k = 2k - 1$ ones. But by definition, $k > 1 \iff k \geq 2$, and thus, $x$ contains at least

$2(2) - 1 = 4 - 1 = 3$ ones. Now we can work with the following definition $B = \{1x | x \in \{0,1\}^* \text{and } x \text{ contains at least 3 ones}\}$. For which we can build the following NFA, via closure properties:




3. (15 points) The pumping lemma says that every regular language has a pumping length $p$, such that every string in the language can be pumped if it has length $p$ or more. If $p$ is a pumping length for language $A$, so is any length $p' \geq p$. The *minimum pumping length* for $A$ is the smallest $p$ that is a pumping length for $A$. For example, if $A = 01^*$, the minimum pumping length is 2. The reason is that the string $s = 0$ is in $A$ and has length 1 yet $s$ cannot be pumped, but any string in $A$ of length 2 or more contains a 1 and hence can be pumped by dividing it so that $x = 0$, $y = 1$, and $z$ is the rest. For each of the following languages, give the minimum pumping length and justify your answer.

   (a) $0001^*$
   **Solution:** if $s = 0$ or $s = 00$ or $s = 000$ it cannot be pumped. It follows that $p = 4$, i.e. $s = 0001$, where $x = 000$ and $y = 1$.

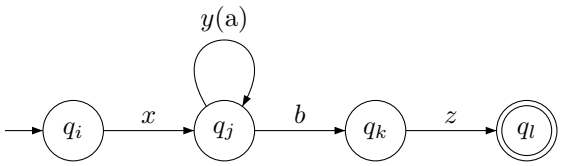
   (b) $001 \cup 0^*1^*$
   **Solution:** if we let $s = 0$, then $x = \epsilon$ and $y = 0$, then $s$ can be pumped and still be in $A$. It follows that $p = 1$

   (c) $1^*01^*01^*$
   **Solution:** the strings in this languages may or may not have ones, but they most have exactly two 0s. To be able to pump a string it must be of the form, $s = 1010$, where $x = 10$, $y = 1$ and $z = 0$. Therefore, $p = 4$. To see why this is the case, let us try to pump a string of less length $s' = 101$. There is no choice of $x$, $y$ and $z$ for this string (or even a smaller one) that would produce a string in the language for all $i \in N$.

4. (10 points) Let A be an infinite regular language. Prove that A can be split into two infinite disjoint regular subsets.

   **Solution:** By the pumping lemma, we know that any string in $A$ of length at least $p$ (the pumping length) can be divided into three pieces $x, y$ and $z$. The following is an schematic of that situation. Please note that this diagram is intended to represent an abstract machine (as in diagram on page 79 of Sipser), and that transition from state $q_i$ to $q_j$ is a transition where there can be intermediate states in between, but that the machine upon consuming the piece $x$ of the word arrives from $q_i$ to $q_j$. A

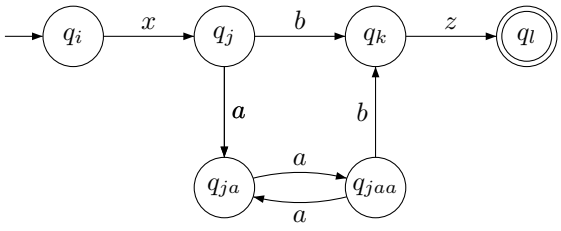
similar case occurs with the other transitions. In particular, the loop over $q_j$ represent all of the states that the piece $y$ goes through so that it can be pumped and still be on the language.



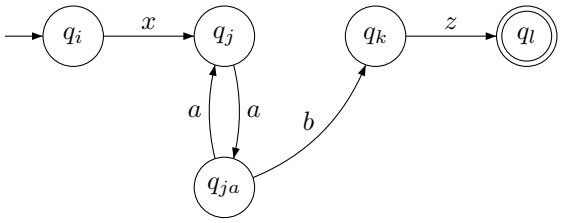

Letters a and b above represent the consumption over some finite alphabet. The alphabet can be changed and the argument will still hold. I prove this only for the case with alphabet $\Sigma = \{a, b\}$, which is the simpler. A more complicated alphabet can be reduced to this one via binary representation, which I not show here.

Now, I will modify this abstract description to yield two NFA that hold the necessary conditions.

$A_1$




$A_2$

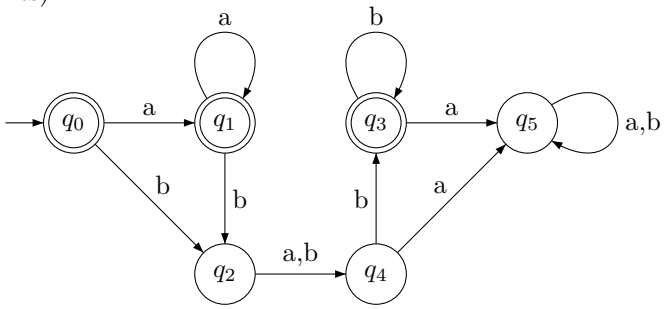



We have that:
$L(A_1) = \{w|$ the words that are in $A$ with an even number of a's or no a in the middle$\}$
$L(A_2) = \{w|$ the words that are in $A$ with an odd number of a's in the middle$\}$.
By this definition, it holds that $A = A_1 \cup A_2$ and $A_1 \cap A_2 = \emptyset$. Also, it is obvious that this regular subsets are infinite. In summary, the idea is to use the pumping lemma and rearrange the states where the string can

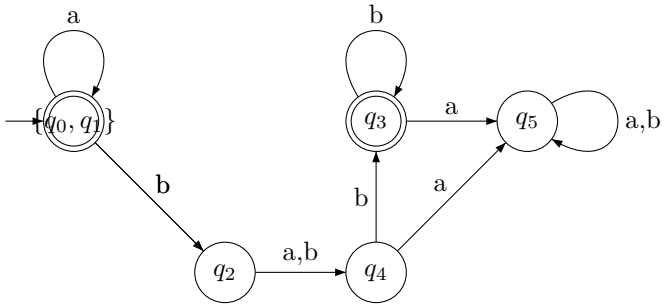
be pumped such that we can yield to NFAs which together represent the same language as the original NFA. Now, what happens if there are some final states between $q_i$ and $q_j$? In that case, make those final states not final in one of $A_1$ or $A_2$, such that only one of them accepts them. All properties still hold. Q.E.D

5. Minimize the following DFA's

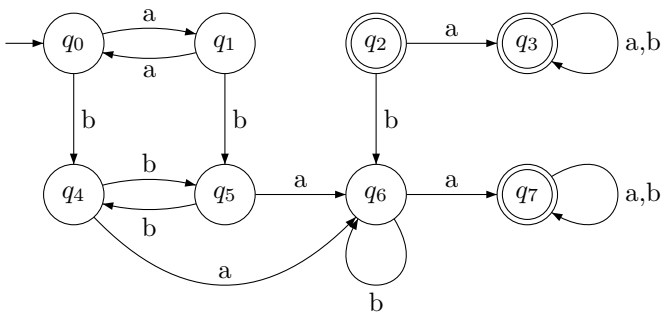
   (a) (10 points)




   **Solution:** The following is the trail of partitions when applying the minimization algorithm to the DFA:
   (1) $A = \{q_2, q_4, q_5\}$, $B = \{q_0, q_1, q_3\}$
   (2) $C = \{q_2, q_5\}, D = \{q_4\}$, $E = \{q_0, q_1\}, F = \{q_3\}$
   (2) $G = \{q_2\}, H = \{q_5\}, D = \{q_4\}$, $E = \{q_0, q_1\}, F = \{q_3\}$
   The only minimization possible is to combine states $q_o$ and $q_1$ resulting in the following DFA:




   (b) (10 points)




**Solution:** States $q_2$ and $q_3$ are not reachable from the start state and thus, can be discarded right away.
The following is the trail of partitions when applying the minimization algorithm to the DFA:
(1) $A = \{q_0, q_1, q_4, q_5, q_6\}$, $B = \{q_7\}$
(2) $C = \{q_0, q_1, q_4, q_5\}, D = \{q_6\}$, $B = \{q_7\}$
(3) $E = \{q_0, q_1\}, F = \{q_4, q_5\}, D = \{q_6\}$, $B = \{q_7\}$
The resulting, minimal DFA is: