CS 311 Homework 5 Solutions

due 16:40, Thursday, 28^{th} October 2010

Homework must be submitted on paper, in class.

Question 1. [30 pts.; 15 pts. each]

Prove that the following languages are not regular using the pumping lemma.

a. $L = \{0^n 1^m 0^n \mid m, n \geq 0\}$.

Answer.

To prove that L is not a regular language, we will use a proof by contradiction. Assume that L is regular. Then by the Pumping Lemma for Regular Languages, there exists a pumping length, p for L such that for any string $s \in L$ where $|s| \geq p$, $s = xyz$ subject to the following conditions:

(a) $|y| > 0$

(b) $|xy| \leq p$, and

(c) $\forall i > 0, xy^i z \in L$.

Choose $s = 0^p 1 0^p$. Clearly, $|s| \geq p$ and $s \in L$. By condition (b) above, it follows that x and y are composed only of zeros. By condition (a), it follows that $y = 0^k$ for some $k > 0$. Per (c), we can take $i = 0$ and the resulting string will still be in L. Thus, $xy^0 z$ should be in L. $xy^0 z = xz = 0^{(p-k)} 1 0^p$. But, this is clearly not in L. This is a contradiction with the pumping lemma. Therefore our assumption that L is regular is incorrect, and L is not a regular language.

b. $L = \{wtw \mid w, t \in \{0, 1\}^+\}$.

Answer.

To prove that L is not a regular language, we will use a proof by contradiction. Assume that L is a regular language. Then by the Pumping Lemma for Regular Languages, there exists a pumping length p for L such that for any sring $s \in L$ where $|s| \geq p$, $s = xyz$ subject to the following conditions:

(a) $|y| > 0$

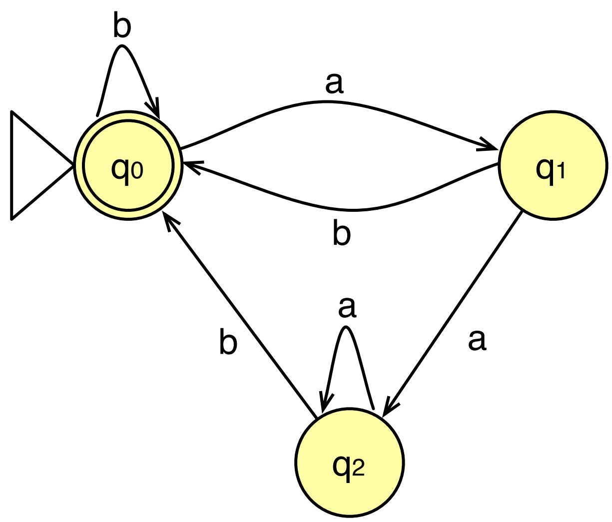
(b) $|xy| \leq p$, and

(c) $\forall i > 0, xy^i z \in L$.

Choose $s = 0^p110^p1$. Clearly $s \in L$ with $w = 0^p1$ and $t = 1$, and $|s| \geq p$. By condition (b), it is obvious that xy is composed only of zeros, and further, by (a) and (b), it follows that $y = 0^k$ for some $k > 0$. By condition (c), we can take any i and xy^iz will be in L. Taking $i = 2$, then $xy^2z \in L$. $xy^2z = xyyz = 0^{(p+k)}110^p1$. There is no way that this string can be divided into wtw as required to be in L, thus $xy^2z \notin L$. This is a contradiction with condition (c) of the pumping lemma. Therefore the assumption that L is a regular language is incorrect and thus L is not a regular language.

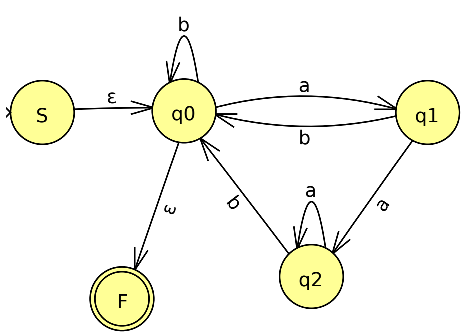
Question 2. [20 pts]

Convert the following DFA into a regular expression using state elimination. Be sure to show intermediate steps of the process.

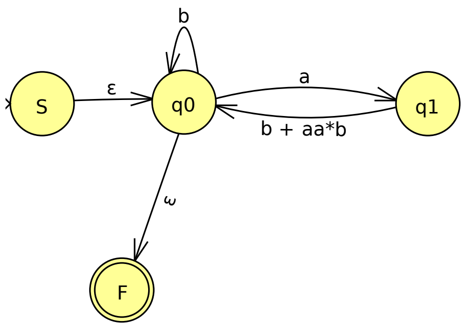


Answer.

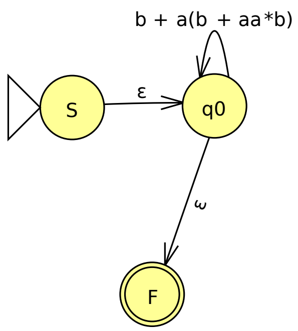
First we introduce a new start and final state, with ε transitions to and from the original start and final states.



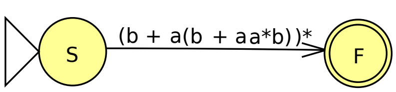
Now we remove state q2, and reconnect state q1 to q0, including the regular expression for the path through q2 along with the original path from q1 to q0.



Now remove q1, adding the regular expression for the path through q1 to the self-loop on q0.



Finally, remove state q0, connecting the start and final state with the regular expression for the self-loop on q0. This regular expression represents all the strings that this NFA accepts.



Question 3. **[20 pts.; 10 pts. each]**

Write context free grammars that generate the following languages. In each case use the alphabet $\Sigma = \{0, 1\}$.

a. $\{x\#y \mid |x| \neq |y|\}$.

Answer.

To construct this grammar, we will build a balanced string of arbitrary length, and then force the generation to choose between a path that forces either the left or the right side to be arbitrarily longer than the other side.

$$\begin{array}{rcl} S & \rightarrow & XSX|XL|RX \\ L & \rightarrow & \#|XL \\ R & \rightarrow & \#|RX \\ X & \rightarrow & 0|1 \end{array}$$

b. $\{w \mid w$ contains at least two occurences of the substring $101\}$

This language is straightforward. Force the inclusion of two occurences of the substring 101 right in the first rule. Then allow arbitrary other substrings to be placed in all other positions.

$$\begin{array}{rcl} S & \rightarrow & A101A101A \\ A & \rightarrow & AA|0|1|\varepsilon \end{array}$$

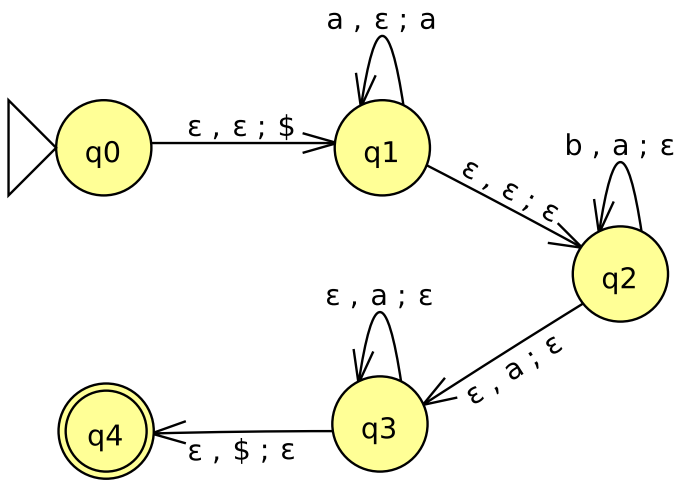
Question 4. [30 pts; 15 pts. each]

Construct PDAs that recognize the following languages:

a. $L = \{a^i b^j \mid i > j\}$

Answer.

This machine will count the number of `a`s by pushing them on the stack. Then it will start comparing `b`s from the input with `a`s on the stack. When all the `b`s are consumed, then the machine will drain the stack and accept if there are still `a`s on the stack. So long as there are `a`s on the stack, then the string of `b`s must be shorter.



b. $L = \{xcy \mid x, y \in \{a, b\}^* \text{ and } \wedge x \neq y^R\}$

This machine will push everything onto the stack until it reads the `c`. Then it will attempt to match the stack against the input, consuming input so long as it matches. The machine will accept if it sees one set of mis-matched characters, or if either part is longer than the other. In all other cases, the machine will get stuck without accepting.