Harvard University Extension School
Computer Science E-207

Problem Set 1

Due Friday, September 21, 2012 at 11:59 PM Eastern Time.
Submit your solutions in a single PDF called lastname+ps1.pdf emailed to cscie207@seas.harvard.edu.
LATE PROBLEM SETS WILL NOT BE ACCEPTED.

Problem set by **ENTER YOUR NAME HERE**

Collaboration Statement: **FILL IN YOUR COLLABORATION STATEMENT HERE
(See the syllabus for information)**

See syllabus for collaboration policy.

PROBLEM 1 (5 points)

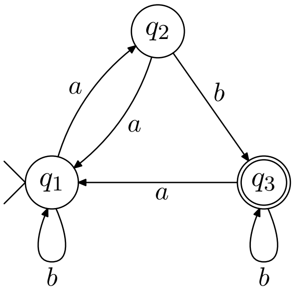
Consider the following game with two players:

Repeatedly flip a coin. On heads, player 1 gets a point. On tails, player 2 gets a point. A player wins (and the game ends) as soon as they are ahead by two points. Draw a DFA that recognizes the language of strings (with alphabet $\{H, T\}$) which represent a possible series of flips in which player 1 wins.

PROBLEM 2 (5+5 points)

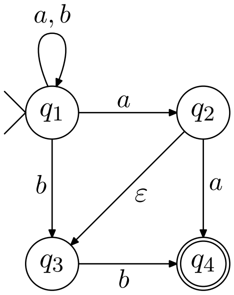
(A) Draw an NFA that recognizes $\{w \in \Sigma^* : w \text{ contains } aba \text{ or } w \text{ contains } bab\}$

(B) Give the 5-tuple representation for the DFA below, and then describe informally the language it distinguishes.



PROBLEM 3 (5+5 points)

Consider the following NFA.



(A) Give the 5-tuple representation for this NFA, and then describe informally the language it distinguishes.

(B) Convert this NFA to an equivalent DFA. (You may omit states not reachable from the start state.)

PROBLEM 4 (8 points)

For a language L, let $L^R = \{w^R : w \in L\}$ (where w^R is the reversal of w). Prove that if L is regular, then so is L^R.

PROBLEM 5 (3+3+3 points)

Are the following statements true or false for all languages L_1, L_2, and L_3? Justify your answers with a proof or counterexample.

(A) $(L_1 \cap L_2)^* = L_1^* \cap L_2^*$.

(B) $(L_1 \cup L_2) \cdot L_3 = (L_1 \cdot L_3) \cup (L_2 \cdot L_3)$, where $\cdot$ is concatenation.

(C) $\{\varepsilon\} \cdot L_1 = \emptyset \cdot L_1$.

PROBLEM 6 (Challenge 1 points)

A DFA M reads its input x once from left to right. What if M can read x again? That is, M reads x from left to right then goes back to the start and reads x from left to right again. Call this a *two-pass DFA*. Does re-reading the input help a DFA overcome its limited memory? Prove that a two-pass DFA is equivalent to a normal DFA.