Problem Set 5

Spring 10: CS373

Due: Problems 1-4 due on Friday, March 12, by 5pm, in Elaine Wilson's office, 3229 SC, by 4pm.

Extra credit Problem 5 due on Thursday, March 18th, in class before class begins, at 2pm.

Please follow the homework format guidelines posted on the class web page:

`http://www.cs.uiuc.edu/class/sp10/cs373/`

1. Turing Machines [**Category**: Construction, **Points**: 20]

   We want to design a Turing machine that given a binary string $\$x$, where $x \in \{0,1\}^*$ is of odd length, computes the middle character of x. To do this our TM overwrites the first and last character of the string x with $\sqcup$, and repeats this procedure till only one character survives on the tape. Then the machine stops and accepts. Carefully draw the diagram of such a TM with input alphabet $\{0, 1, \$\}$ and tape alphabet $\{0, 1, \$, \sqcup\}$.

2. Non-Regularity [**Category**: Proof, **Points**: 20]

   Prove that the following language is not regular:

   $$L = \{0^{n^3} : n \geq 0\}$$

   Hint: If you are using the Myhill-Nerode Theorem, you may want to choose S to be L.

3. Non-determinism [**Category**: NFA design, **Points**: 20]

   Let P be a regular language over $\{0, 1\}$. Let L_0 and L_1 be two regular languages over $\{a, b\}$.

   Let L be the set of words w over $\{a, b\}$ such that w can be split into n words (for some n), $w = w_1 w_2 \ldots w_n$, and there exists a word of length n, $x = x_1, x_2 \ldots x_n \in P$ such that each $w_i \in L_{x_i}$. In other words,

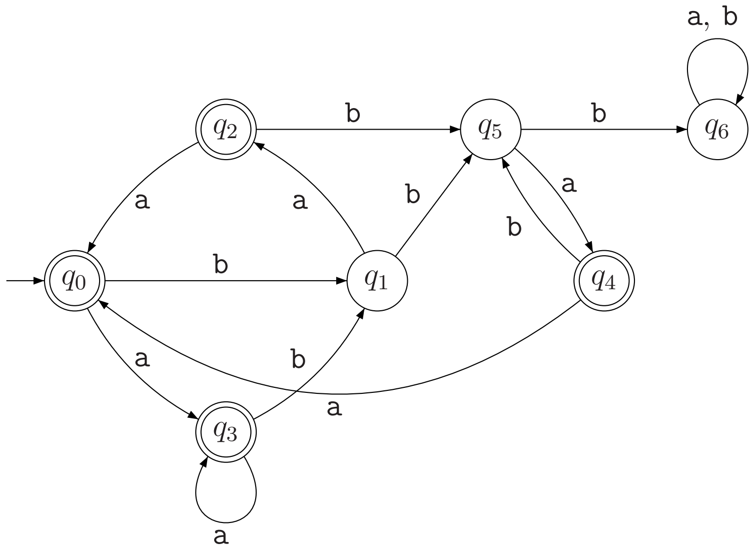
   $$L = \{w \in \{a,b\}^* \mid \exists x \in P, x = x_1 \ldots x_n, x_i \in \{0,1\}, w_i \in L_{x_i}, w = w_1 \ldots w_n\}$$

   Intuitively, L consists of words formed by concatenating words in L_0 and L_1, using a pattern described in P.

   Show L is regular by exhibiting an NFA for it. Give the formal description of your NFA. You do not need to prove formally that your NFA accepts this language, but you do need to give a description of how and why it works.

4. Minimization [**Category**: Construction., **Points**: 20]

   Recall the minimization algorithm by partition refinment(Lecture Note #11). Use this algorithm to minimize the following DFA and draw the resulting minimal DFA. Show the partitions of the states at every iteration clearly.



5. Extra Credit/Honors: Clause Finite Automata [**Category**: Construction. Due only on Thu, Mar 18th, 2pm., **Points**: 20]

   A Clause Finite Automaton(CFA) is a generalization of an NFA (without ϵ transitions) such that the value of the transition function is no longer a set of states, but a combination of states by disjunctions/conjunctions. For example, if $\delta(q, \mathtt{a}) = q_1 \vee (q_2 \wedge q_3)$ in a CFA M, then at state q, reading $\mathtt{a}$, M nondeterministically choose to switch to q_1, or simulate the behavior of M from *both* q_2 and q_3 on the rest of the word. Intuitively, this transition says that M accepts the word aw from q if M accepts the word w from q_1, or, M accepts the word w from both q_2 and q_3.

   Note that every NFA can be viewed as a CFA, since each transition $\delta(q, \mathtt{a}) = P$, where P is a subset of states, can be viewed as $\delta(q, \mathtt{a}) = \bigvee_{p \in P} p$.

   Formally, a CFA is a tuple $M = (Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ is a finite alphabet, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is a set of final or accepting states, and $\delta : Q \times \Sigma \to B(Q)$ where $B(Q)$ is the set of all Boolean formulas over Q formed using conjunction and disjunction.

   The notion of when M accepts a word is as follows:

   - M accepts ϵ from state $q \in Q$ iff $q \in F$
   - M accepts aw from state q (where $w \in \Sigma^*$ and $a \in \Sigma$) iff there is a set of states $Q' \subseteq Q$ such that the valuation that sets Q' to true and $Q \setminus Q'$ to false satisfies the Boolean formula $\delta(q, a)$, and M accepts w from each of the states $q' \in Q'$.

   (a) Consider a CFA $M = (Q, \Sigma, \delta, q_0, F)$ where

   $Q = \{q_0, q_1, q_2, q_3, q_4\}$;
   $\Sigma = \{\mathtt{a}\}$;
   δ is defined as follows:

δ	a
q_0	$(q_1 \vee q_2) \wedge (q_3 \vee q_4)$
q_1	q_3
q_2	q_4
q_3	q_1
q_4	q_2

$F = \{q_1, q_3\}$.

Convert M to an equivalent NFA. (10 Points)

(b) Given $\Sigma = \{0, 1, 2\}$ and $k > 0$, let $L_k = \{ww \mid |w| = k\}$. Construct a CFA for L_k with $O(k)$ states, for every k. (10 Points)