Automata Qual Exam (Spring 2013)
Answer ALL questions (Closed Book Exam)

Question 1 (15 + 15 points)

Let $A_{\text{TM}} = \{<M, w> \mid M \text{ is a TM and } M \text{ accepts } w\}$. You can assume without proof that A_{TM} is not Turing-decidable (recursive).

For this question, recall that the input to a Turing machine is located in the leftmost part of the tape, while the unbounded number of cells to the right of the last input symbol are all initially blanks. That is, the tape is only infinite to the right direction, while there is a left end where the input begins.

(a) Is it decidable that, given M and w, to determine whether the Turing machine M reads the last symbol in its input w?

Hint: If the Turing machine never reads the last input symbol, the machine will not read any symbol to the right hand side of the last input symbol.

Answers: Decidable. Suppose there are n input symbols. If M does not reach the last input symbol, it can only move around among the first $n-1$ symbols. In that case, the number of possible configurations is at most $(n-1)|Q||\Gamma|^{n-1}$ where Q is the set of states and Γ is the tape alphabet. So, if the Turing machine does not read the last input symbol within $(n-1)|Q||\Gamma|^{n-1}$ steps, the machine will never read the last input symbol.

(b) Is it decidable that, given M and w, to determine whether the Turing machine M changes the last symbol in its input w?

Answers: Undecidable. We can reduce A_{TM} to our problem. Given an input $<M, w>$ to A_{TM}, we extend w by one extra symbol # (where # does not appear in w) at the right end of w. Next, we modify M to M' so that the machine always skips over #. It is only when the machine is about to accept that it will go back to change the # symbol. We can make use of a solver for our problem to $<M', w\#>$ for deciding if $<M, w> \in A_{\text{TM}}$.

Question 2 (5 points + 15 points + 5 points + 5 points)

(a) Consider the grammar $S \longrightarrow AS \mid \epsilon$, $A \longrightarrow a \mid bAA$. Is the grammar ambiguous? Justify your answer. (Note: a careful mathematically sound argument is expected.)

Answers: The grammar is unambiguous as the leftmost derivation sequence is forced for any input. In the leftmost derivation, if the leftmost nondeterminal of a sentential form is S and

if the input is not yet exhaused, we must expand S by AS; if the leftmost nondeterminal is S and the input is exhausted, we expand S by ϵ; if the leftmost nondeterminal is A, depending on whether the input is a or b, there is only one rule that can be applied to rewrite A.

(b) Prove by induction that $L(A) \subseteq \{w \mid w \text{ has more } a\text{'s than } b\text{'s}\}$.

Answers: By induction on the number of derivation steps. Base case: In one step, A can only generate a which has more a's than b's. Hypothesis: Given that $A \stackrel{\leq k}{\Longrightarrow} w$, we have $|w|_a > |w|_b$. Induction step: Consider $A \stackrel{k+1}{\Longrightarrow} w$. The first derivation must be $A \Longrightarrow bAA$. Therefore, $A \Longrightarrow bAA \stackrel{\leq k_1}{\Longrightarrow} bw_1A \stackrel{\leq k_2}{\Longrightarrow} bw_1w_2 = w$ where $k_1 < k$ and $k_2 < k$. By induction hypothesis, $|w_1|_a > |w_1|_b$ and $|w_2|_a > |w_2|_b$, which implies that $|w|_a > |w|_b$ as $w = bw_1w_2$.

(c) Is $L(S) \subseteq \{w \mid w = \epsilon \text{ or } w \text{ has more } a\text{'s than } b\text{'s}\}$? Justify your answer.

Answers: Yes. From $S \longrightarrow AS \mid \epsilon$, we have $S \Longrightarrow \epsilon$ or $S \stackrel{*}{\Longrightarrow} AA\ldots A$. Since $L(A) \subseteq \{w \mid w$ has more a's than b's$\}$, we conclude that $L(S) \subseteq \{w \mid w = \epsilon$ or w has more a's than b's$\}$.

(d) Is $L(S) = \{w \mid w = \epsilon \text{ or } w \text{ has more } a\text{'s than } b\text{'s}\}$? Justify your answer.

Answers: No. S cannot generate the string aab which has more a's than b's.

Question 3 (10 points + 10 points + 10 points + 10 points)

Our aim in this question is to construct a finite automaton that accepts the set of nonempty strings over Σ that have the same value when evaluated left to right as right to left by multiplying according to a given multiplication table. An example multiplication table where $\Sigma = \{a, b, c\}$ is given below:

$*$	a	b	c
a	a	a	c
b	c	a	b
c	b	c	a

With respect to the above multiplication table, *baba* is accepted since $(((ba)b)a) = ((cb)a) = (ca) = b$ and $(b(a(ba))) = (b(ac)) = (bc) = b$, while *bbac* is not accepted since $(((bb)a)c) = ((aa)c) = (ac) = c$ and $(b(b(ac))) = (b(bc)) = (bb) = a$

Your answers should refer to the multiplication by its name $*$. In other words, the multiplication function $* : \Sigma \times \Sigma \to \Sigma$ given may not necessarily correspond to the above example multiplication table. Moreover, Σ may not necessarily be a set of three letters.

Your answers to the following questions will be graded not only by its correctness, but also by the correct and concise use of mathematical notations.

Also, the following questions are formulated independently. Even if you cannot work out the answer to an earlier question, you still have a chance to solve the next question.

(a) Design a DFA M_a for accepting nonempty strings that are evaluated to a from left to right.

Answers: $M_a = (\{q_0\} \cup \{q_\alpha \mid \alpha \in \Sigma\}, \Sigma, \delta_a, q_0, \{q_a\})$ *where* $\delta_a(q_0, \alpha) = q_\alpha$, $\delta_a(q_\alpha, \beta) = q_{\alpha * \beta}$.

(b) Design an NFA M'_a for accepting nonempty strings that are evaluated to a from right to left.

Answers: $M'_a = (\Sigma \cup \{accept\}, \Sigma, \{(\alpha, \beta, \gamma) \mid \alpha = \beta * \gamma,\ \alpha, \beta, \gamma \in \Sigma\} \cup \{(\alpha, \alpha, accept) \mid \alpha \in \Sigma\}, \{a\}, \{accept\})$

(c) Explain using mathematical notations how to combine DFA $M_a = (Q, \Sigma, \delta, q_0, F)$ and NFA $M'_a = (Q', \Sigma, \delta', q'_0, F')$ to construct an NFA M''_a that accepts nonempty strings that are evaluated to a from left to right and right to left.

Answers:
$M''_a = (Q \times Q', \Sigma, \delta'', (q_0, q'_0), F \times F')$ *where* $\delta''((q, q'), \alpha) = \{(\delta(q, \alpha), q'') \mid q'' \in \delta'(q', \alpha)\}$.

(d) Suppose $\Sigma = \{a, b, c\}$. Let the NFA constructed in part (c) be denoted as $M''_a = (Q_a, \Sigma, \delta_a, q^a_0, F_a)$. Suppose we have similarly constructed NFAs $M''_b = (Q_b, \Sigma, \delta_b, q^b_0, F_b)$ and $M''_c = (Q_c, \Sigma, \delta_c, q^c_0, F_c)$. Explain using mathematical notations how to combine M''_a, M''_b and M''_c to give an NFA M that solves the original goal of the problem.

Answers:
$M = (Q_a \cup Q_b \cup Q_c, \Sigma, \delta, \{q^a_0, q^b_0, q^c_0\}, F_a \cup F_b \cup F_c)$ *where* $\delta(q, \alpha) = \delta_x(q, \alpha)$ *for* $q \in Q_x$.