CS208 Mid-semester Exam (Spring 2014)

Time: 3 hours **Date: Feb 19, 2014**

- **Be brief, complete, and stick to what has been asked.**
- **You may cite results/proofs covered in the class without reproducing them.**
- **Do not copy solutions form others.**
- **Penalty for copying: FR grade.**

1. [*5*+5+5+5+5 marks] Construct deterministic finite automata accepting each of the following languages (L_1 – L_4):
   (a) $L_1 = \{w \in \{a,b\}^* \ : \$ each a in w is immediately preceded and immediately followed by a $b\}$
   (b) $L_2 = \{w \in \{a,b\}^* \ : \ w$ has $3k+1$ b's for some $k \in \mathbb{N}\}$
   (c) $L_3 = \{w \in \{a,b\}^* \ : \ w$ has neither aa nor bb as a substring$\}$
   (d) $L_4 = \{w \in \{a,b\}^* : w$ has even number of a's and one or two b's$\}$
   (e) Give regular expression corresponding to the language L_2.

2. [*8*+8+8+6 marks] We have shown in the class that regular languages are closed under union, intersection, complementation, concatenation, and Kleene closure. Recall that to prove that regular languages are closed under intersection, we showed that given DFAs $\mathcal{A}_1$ and $\mathcal{A}_2$ how to construct (product construction) a DFA accepting the language $L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$ (where $L(\mathcal{A})$ is the language accepted by the DFA $\mathcal{A}$) and proved the correctness of the construction.

   Prove the following closure properties (i.e. give a formal construction to reduce an arbitrary DFA to the DFA after the operation (50% marks), and prove the correctness of the construction via structural induction (50% marks)):

(a) Given a string $w \in \Sigma^*$ we define mirror of the string $\overleftarrow{w} \in \Sigma^*$ recursively in the following manner:

$$\overleftarrow{w} = \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a\overleftarrow{x} & \text{if } w = xa \text{ where } x \in \Sigma^* \text{ and } a \in \Sigma. \end{cases}$$

Notice, for instance $\overleftarrow{aabbc} = cbbaa$. We can extend the notion of mirror from strings to languages. For a language L we define its mirror language $\overleftarrow{L}$ as the set $\{\overleftarrow{w} : w \in L\}$. Prove that if L is accepted by some finite automaton, then so is $\overleftarrow{L}$.

(b) Given a string $w \in \Sigma^*$ and a letter $a \in \Sigma$ we define $\textsc{drop}(w, a)$ recursively in the following manner:

$$\textsc{drop}(w, a) = \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ \textsc{drop}(x, a).b & \text{if } w = xb \text{ for } x \in \Sigma^*,\ b \in \Sigma \text{ and } a \neq b. \\ \textsc{drop}(x, a) & \text{if } w = xb \text{ for } x \in \Sigma^*,\ b \in \Sigma \text{ and } a = b. \end{cases}$$

For example, $\textsc{drop}(bcabca, c) = baba$. The drop operation can be extended from strings to languages in a straightforward manner. For a language $L \subseteq \Sigma^*$ and a letter $a \in \Sigma$ we define $\textsc{drop}(L, a)$ as the set $\{\textsc{drop}(w, a) : w \in L\}$. Prove that if L is accepted by some finite automaton, then so is $\textsc{drop}(L, a)$.

(c) For a language $L \subseteq \Sigma^*$ and a letter $a \in \Sigma$ we define $\textsc{chop}(L, a)$ as the set

$$\textsc{chop}(L, a) = \{x : xa \in L\}.$$

For example if $L = \{a, bba, aab, ababa\}$ then $\textsc{chop}(L, a) = \{\varepsilon, bb, abab\}$. Prove that if L is accepted by some finite automaton, then so is $\textsc{chop}(L, a)$.

(d) Let $L.a$ be the concatenation of the languages L and $\{a\}$. Which of the following is true? Justify your answer.

- $\textsc{chop}(L, a).a = L$
- $\textsc{chop}(L.a, a) = L$

3. [15 marks] Let $D = \{0, 1\}$ and let $T = D \times D \times D$. A correct addition of two binary numbers can be presented as a string in T^* if we think of symbols in T as vertical columns. For example the binary addition

$$\begin{array}{cccc} & 0 & 1 & 0 & 1 \\ + & 0 & 1 & 1 & 0 \\ \hline & 1 & 0 & 1 & 1 \end{array}$$

can be written as the string $w = (0, 0, 1)(1, 1, 0)(0, 1, 1)(1, 0, 1)$. Show that the set of all strings in T^* that represent correct additions is a regular language. [Hint: Question 2(a).]

4. [5+5+5+10+5 marks] Recall the standard semantics of non-deterministic finite automata (NFA) $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ introduced in the class. For simplicity, we assume that the NFA is ε-free and complete, i.e. for every state $q \in Q$ and letter $a \in \Sigma$ we have that the set $\delta(q, a)$ is not empty. A *run* of the NFA $\mathcal{A}$ over a string $w = a_1 a_2 \dots a_n$ is a sequence of states $(q_0, q_1, \dots, q_{n-1}, q_n)$ such that q_0 is the initial state, and for all $0 \leq i < n$ we have that $q_{i+1} \in \delta(q_i, a_{i+1})$. There may be more than one run of an NFA over a string. We say that a string $w = a_1 a_2, \dots a_n$ is accepted by the NFA $\mathcal{A}$ if there is **at least one run** $(q_0, q_1, \dots, q_n)$ such that $q_n \in F$. We define the language accepted by the NFA as the set of strings accepted by it. We also defined the language accepted by the NFA using *extended transition function* $\hat{\delta}$.

   In this question you are required to develop a different notion of acceptance for NFA such that a string w is accepted if **all the runs** $(q_0, q_1, \dots, q_n)$ of the NFA over the string w are such that $q_n \in F$. We call such an NFA a *universal finite automaton.*

   (a) Formally define a *universal finite automaton* (UFA) and the language defined by UFA using extended transition function $\hat{\delta}$;

   (b) Give the computation tree for the UFA shown in Fig. 1 for strings *ababa* and *abba*, and answer whether the UFA accepts these strings.

   (c) Describe the language accepted by the UFA shown in Fig. 1.

   (d) Prove that every universal finite automaton has an equivalent deterministic finite automaton.

   (e) Using the construction shown in the previous proof, determinize the automaton shown in Figure 1.

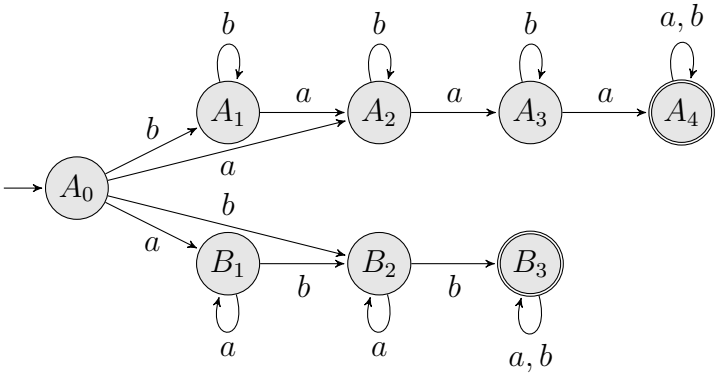

Figure 1: A universal finite automaton