11-711: Algorithms for NLP

Homework Assignment #1: Formal Language Theory

Out: September 10, 2009
Due: September 24, 2009

Problem 1

Prove that, for any *deterministic* FSA $A = (Q, \Sigma, \delta, q_0, F)$,

$$\hat{\delta}(q, xy) = \hat{\delta}\left(\hat{\delta}(q, x), y\right)$$

for $x, y \in \Sigma^*$. Use the definition of $\hat{\delta}$ provided in lecture:

$$\begin{array}{ll} (1) & \hat{\delta}(q, \epsilon) = q \\ (2) & \hat{\delta}(q, x\sigma) = \delta\left(\hat{\delta}(q, x), \sigma\right) \end{array}$$

where ϵ is the empty string, $x \in \Sigma^*$, and $\sigma \in \Sigma$.

Problem 2

Give *deterministic* FSAs accepting the following languages:

1. The set of strings over $\{a, b, c\}$ in which all the as precede the bs, which in turn precede the cs. It is possible that there are no as, bs, or cs. (Sudkamp Problem 6.5)
2. The set of strings over $\{a, b, c\}$ in which every b is immediately followed by at least one c. (Sudkamp Problem 6.10)
3. The set of strings over $\{0, 1\}$ such that the third symbol from the right end is the same as the last symbol.

Problem 3

Give *non-deterministic* FSAs (possibly with ϵ moves) accepting the following languages:

1. The set of strings over $\{a, b\}$ whose third and third-to-last symbols are both b. For example, *aababaa*, *abbbbbbbb*, and *abba* are in the language. (Sudkamp Problem 6.27)
2. The set of strings over $\{0, 1\}$ such that the third symbol from the right end is the same as the last symbol. How does your NDFSA differ from the DFSA you constructed in Problem 2?

Problem 4

Give regular expressions for each of the following languages:

1. The set of strings over $\{a, b, c\}$ in which every b is immediately followed by at least one c. (Sudkamp Problem 2.25)
2. The set of strings over $\{0, 1\}$ in which all occurences of the substring 00 appear before any occurence of the substring 11. (HU Problem 2.10(b))

Problem 5

Let us define a new model of FSA called a two-initial-state FSA (TISFSA). The machine has *two* initial states, and is otherwise identical to a DFSA. In other words, a TISFSA is a deterministic finite state automaton with a single non-deterministic feature: once the machine is reset, it can be in one of two initial states. Once the initial state is chosen (non-deterministically), computation procedes in a deterministic fashion.

Formally, a TISFSA is defined as $A = (Q, \Sigma, \delta, q_0^1, q_0^2, F)$. The input alphabet Σ is the same for both controls. Q, δ, q_0^1, q_0^2, and F are the set of states, the transition function, the two initial states, and the set of final states. The transition function δ is a completely defined function, defined on $Q \times \Sigma \to Q$. The language accepted by such a machine is defined as:

$$L(A) = \{w \in \Sigma^* \mid \delta(q_0^1, w) \in F \text{ or } \delta(q_0^2, w) \in F\}$$

1. Show that the set of languages accepted by a TISFSA is regular. *Note: You must show a construction appropriate and specific for TISFSAs. You may not rely on the "power set" construction for NDFSAs.* (Hint: Construct an equivalent DFSA A' that accepts the same language as the TISFSA A. Then show that $w \in L(A)$ *if and only if* $w \in L(A')$.)
2. Let $A_1 = (Q, \Sigma, \delta, q_0^1, F)$ and $A_2 = (Q, \Sigma, \delta, q_0^2, F)$ be standard DFSAs, and let $A = (Q, \Sigma, \delta, q_0^1, q_0^2, F)$ be a TISFSA. Prove that $L(A) = L(A_1) \cup L(A_2)$.