Problem Set 2 — Due January 18, 2005

Problem 1. Give DFAs for the following languages. Assume an alphabet of $\Sigma = \{0, 1\}$.

(a) The set of all strings with 010 as a substring.

(b) The set of all strings which do not have 010 as a substring.

(c) The set of all strings which have an even number of 0's or an even number of 1's.

(d) The complement of $\{1, 10\}^*$.

(e) The binary encodings of numbers divisible by 3: $\{0\}^* \circ \{\varepsilon, 11, 110, 1001, 1100, 1111, \ldots\}$.

Problem 2 State whether the following propositions are true or false, proving each answer.

Part A. Every DFA-acceptable language can be accepted by a DFA with an even number of states.

Part B. Every DFA-acceptable language can be accepted by a DFA whose start state is never visited twice.

Part C. Every DFA-acceptable language can be accepted by a DFA no state of which is ever visited more than once.

Part D. Every infinite DFA-acceptable language can be accepted by a DFA that, for some string $x \in L$, visits the start state twice on input x.

Part E. Every DFA-acceptable language can be accepted by a DFA with only a single final state.

Problem 3. Let $\mathit{Extend}(L) = \{x \in L : \text{ there exists a } y \in \Sigma^+ \text{ for which } xy \in L\}$. ($\Sigma^+ = \Sigma\Sigma^*$.)

Part A. What is $\mathit{Extend}(\{\mathtt{0}, \mathtt{1}\}^*)$? What is $\mathit{Extend}(\{\varepsilon, 0, 1, 00, 01, 111, 1110, 1111\})$?

Part B. Prove that if L is DFA-acceptable then $\mathit{Extend}(L)$ is DFA-acceptable.

Problem 4.

Part A. Recall the Diophantine Equation problem: given a multivariate polynomial P with integer coefficients (e.g., $P(x, y, z) = x^2 - 5xy + 3yz^2 + xyz$), decide whether or not P has an integer root[1]. I claimed without proof (and you may use this for part B) that there is no algorithm to answer this question. But suppose I provide you with a "magic box" that answers the question. In a single computational step, it says *yes* or *no* according to whether or not P has a root. Given such a magic box, describe an algorithm that *finds* an integer root of any multivariate polynomial that has one (and the algorithm answers *No Root* if the polynomial provided doesn't have an integer root). (In short, your are showing that for Diophantine Equation, a solution for the decision problem implies a solution to the search problem.)

Part B. Let $s(n)$ be the maximum number of computational steps that your algorithm takes to run (on some fixed, magicbox-containing computer) when the polynomial P contains at most n variables and each coefficient is at most n. Explain why there is no algorithm to compute $S(n)$ for any function S such that $S(n) \geq s(n)$ for all n. (In short, you are showing that some functions are just too darn big to be computed.)

[1] I.e., a way of setting the variables to integer values so that P evaluates to 0.