Automata and Formal Languages Comprehensive Exam (60 Points)

Fall 2005

Problem 1 (10 points)

(a) Show that if L is a regular language, then so is

$$min(L) = \{w : w \text{ is in } L, \text{ but no proper prefix of } w \text{ is in } L\}.$$

(b) Show that if L is a regular language, then so is

$$init(L) = \{w : \text{there exists an } x \text{ such that } wx \text{ is in } L\}.$$

Hint: For each part, start with a DFA for L and modify it. Include *brief* arguments justifying your constructions.

Solution:

(a) Let M be a DFA accepting L (such a machine exists because L is regular). Obtain a new DFA M' from M as follows. Add an extra "null" state z which is non-accepting. All transitions from z lead back to z. To every accepting state q of M, add transitions (for all input symbols) from q to z. If $w \in L$ and no prefix of w is in L, then M will reach an accepting state for the first time once w is fully consumed. Thus w will lead M' to an accepting state. Conversely, if some prefix of w is in L, then M will first reach an accepting state before w is fully consumed. Thus on input w, the DFA M' will conclude in the null state. Thus M' accepts precisely the language $min(L)$.

(b) Let M be a DFA accepting L (such a machine exists because L is regular). Obtain a new DFA M' from M as follows. For every state q of M for which there is a (possibly empty) sequence of transitions from q to an accepting state of M, make q accepting in M'. (Other states remain non-accepting in M'.) Then an input w is accepted by M' if and only if there is a string x such wx is accepted by M. Thus M' accepts precisely the language $init(L)$.

Problem 2 (15 points)

Decide whether the following statements are TRUE or FALSE. *You will receive 3 points for each correct answer and -2 points for each incorrect answer.*

(a) There is a language L such that neither L nor its complement are recursively enumerable.

(b) Suppose there is a polynomial-time reduction from the language L_1 to the language L_2. If L_1 is NP-hard, then L_2 must be NP-complete.

(c) Suppose there is a polynomial-time reduction from the language L_1 to the language L_2. If L_1 is NP-complete, then L_2 must be NP-hard.

(d) The following language is recursively enumerable: encodings of Turing machines that accept at least 154 different inputs.

(e) The following language is recursively enumerable: encodings of Turing machines that accept at most 154 different inputs.

Solution:

(a) TRUE

(b) FALSE

(c) TRUE

(d) TRUE

(e) FALSE

Problem 3 (15 points)

Classify each of the following languages as being in one of the following classes of languages: *empty, finite, regular, context-free, recursive, recursively enumerable, all languages*. You must give the *smallest* class that contains *every possible language* fitting the following definitions. For example, the language of a DFA could be empty or finite, and must always be context-free, but the smallest class that contains all such languages is that of the *regular* languages. *You will receive 3 points for each correct answer and -2 points for each incorrect answer.*

(a) A subset of a regular language.

(b) The concatentation of two recursively enumerable languages. (Recall that the concatenation of languages L_1 and L_2 is $L_1L_2 = \{wx \mid w \in L_1, x \in L_2\}$.)

(c) The concatentation of two recursively enumerable languages, one of which is the complement of the other.

(d) An NP-complete language.

(e) An NP-hard language.

Solution:

(a) All languages

(b) Recursively enumerable

(c) Recursive

(d) Recursive

(e) All languages

Problem 4 (20 points)

An instance of the INTEGER LINEAR PROGRAMMING PROBLEM is the following: given a set of linear constraints of the form $\sum_{i=1}^{n} a_i x_i \leq c$ or $\sum_{i=1}^{n} a_i x_i \geq c$, where the a's and c's are integer constants and x_1, x_2, ..., x_n are variables, does there exist an assignment of integers to each of the variables that makes all of the the constraints true? Prove that the INTEGER LINEAR PROGRAMMING PROBLEM is NP-hard.

Solution: Recall the NP-complete 3-SAT problem: given a set of Boolean variables and a set of disjunctions of 3 literals each, is there a truth assignment that simultaneously satisfies all of the clauses? We can establish NP-hardness of ILP via a polynomial-time reduction from 3-SAT.

The reduction starts with a 3-SAT instance, given by variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$, where C_i has the form $l_1 \vee l_2 \vee l_3$, where l_1, l_2, l_3 are literals (each of the form x_j or $\neg x_j$ for some j). It then constructs an instance of ILP as follows. For each Boolean variable x_j, there is an integer variable y_j with

the interpretation that if $y_j = 1$ then x_j should be set to "true" and if $y_j = 0$ then x_j should be set to "false". For every j, add the constraints $x_j \geq 0$ and $x_j \leq 1$. For every j, introduce the variable z_j and the constraint $z_j = 1 - y_j$ to model the value of $\neg x_j$. (Strictly speaking, we accomplish this by adding the inequalities $z_j \geq 1 - y_j$ and $z_j \leq 1 - y_j$.) Finally, for every clause of the form $l_i \vee l_j \vee l_k$ we construct a constraint $(y_i, z_i) + (y_j, z_j) + (y_k, z_k) \geq 1$, where by (e.g.) (y_i, z_i) we mean y_i if $l_i = x_i$ and z_i if $l_i = \neg x_i$. This reduction runs in linear time, but it remains to verify its correctness.

First suppose that there is a satisfying truth assignment to the given 3-SAT instance. For each Boolean variable x_i, set the corresponding integer variable y_i to 1 if x_i is set to true in the satisfying assignment and to 0 otherwise. Set $z_i = 1 - x_i$ for each i. Each (y_i, z_i) equals 1 if and only if the corresponding literal is satisfied by the truth assignment. Since the truth assignment satisfies all of the 3-SAT clauses, the left-hand side $(y_i, z_i) + (y_j, z_j) + (y_k, z_k)$ of every constraint is at least 1. Thus the assignment to the the integer variables is also satisfying.

Conversely, suppose there is an assignment to the integral variables of the ILP instance that satisfies all of the linear constraints. Set x_i to true (false) if $y = 1$ ($y = 0$). Reversing the argument in the previous paragraph shows that this a satisfying truth assignment to the given 3-SAT instance.