G22.3520: Honors Analysis of Algorithms

Problem Set 6+7

Due on Mon Dec 14, after the class

Collaboration is allowed, but you must write your own solutions. **Absolutely no extensions.**

For all problems, the alphabet $\Sigma = \{0, 1\}$.

Problem 1

Solve Problem 4 from: http://www.cs.nyu.edu/courses/fall07/G22.3520-001/ps2.pdf

Note: This is the problem about the algorithm $A(S)$.

Problem 2 (Pumping Lemma for regular languages)

Consider a DFA with n states and suppose x is a string accepted by the DFA with $|x| > n$. Show that x can be written as $x = uvw$ (i.e. concatenation of three strings u, v, w) such that $|v| > 0$ and uv^iw is accepted by the DFA for every integer $i \geq 0$ (i.e. concatenation of u, v repeated i times, and w).

Using this lemma, prove that $\{0^k1^k \mid k \geq 1\}$ is not a regular language.

Problem 3

The standard procedure for converting an NFA to an equivalent DFA yields an exponential blowup in the number of states. That is, if the original NFA has n states, then the resulting DFA has 2^n states. In this problem, you will show that such an exponential blowup is necessary in the worst case.

Define $L_n = \{w :$ The nth symbol from the right is $1\}$.

1. Give an NFA with $n + 1$ states that recognizes L_n.
2. Prove that any DFA with fewer than 2^n states cannot recognize L_n. *Hint: Let M be any DFA with fewer than 2^n states. Start by showing that there exist two different strings of length n that drive M to the same state.*

Problem 4

If A is a language, let $A_{-\frac{1}{2}}$ be the set of all first halves of strings in A, i.e.

$$A_{-\frac{1}{2}} = \{x \mid \text{for some } y, |x| = |y|, \text{and } xy \in A\}.$$

Show that if A is regular, so is $A_{-\frac{1}{2}}$. *Hint: Run two DFAs "in parallel", one forward and the other backward.*

Problem 5

For a language A, let SUFFIX(A) denote the set of all suffixes of strings in A, i.e.

$$\text{SUFFIX}(A) = \{v \mid uv \in A \text{ for some string } u\}.$$

Show that if A is a context-free language, so is SUFFIX(A). *Hint: Design a PDA.*

Problem 6

Show that the following languages are decidable:

1. $\text{INFINITE}_{\text{DFA}} = \{\langle D\rangle \mid D \text{ is a DFA such that } L(D) \text{ is infinite}\}$.
2. $L = \{\langle R, S\rangle \mid R \text{ and } S \text{ are regular expressions such that } L(R) \subseteq L(S)\}$.

Problem 7

In this problem, we explore the notion of *oracle reducibility.* If A is a language, then a *Turing machine with oracle* A is a Turing machine with a "magical" subroutine that decides membership in A. In other words, the subroutine, when given a string w, tells the machine whether or not $w \in A$. Let

$$\text{HALT}_{\text{TM}} = \{\langle M, x\rangle \mid M \text{ is a Turing machine that halts on } x\}.$$

Show that there is a Turing machine with oracle HALT_{TM} that decides the following problem with only *two* questions to the oracle: Given three (machine, input) pairs $\langle M_1, x_1\rangle, \langle M_2, x_2\rangle, \langle M_3, x_3\rangle$, decide for each pair whether the Turing machine halts on the corresponding input.

Note: This is trivial if one allows *three* questions. Just ask the oracle whether $\langle M_i, x_i\rangle \in \text{HALT}_{\text{TM}}$ for $i = 1, 2, 3$.

Problem 8

Show that the collection of Turing-recognizable languages is closed under the operation of (a) union (b) concatenation (c) star and (d) intersection. What about complementation operation ?

Problem 9

Show that every context free language is in P. In other words, for any fixed grammar G, design a polynomial time algorithm to test whether a given string w is generated by the grammar. The algorithm should run in time polynomial in $|w|$. The size of the grammar itself is thought of as a constant. *Hint: Use dynamic programming. Assume w.l.o.g. that the grammar is in Chomsky normal form.*

Problem 10

Show that P is closed under the star operation (*Hint: Use dynamic programming.*) Recall that for a language L,

$$L^* = \{x_1 x_2 \dots x_k \mid k \geq 0,\ x_i \in L\ \forall 1 \leq i \leq k\}$$

Problem 11 (Do not submit)

Show that these problems are NP-complete (you can assume that SAT and CLIQUE are NP-complete).

1. DOUBLE-SAT $= \{\langle\phi\rangle \mid \phi$ is a boolean formula that has at least two satisfying assignments$\}$.
2. HALF-CLIQUE $= \{\langle G\rangle \mid G$ is a graph with a clique of size at least $|G|/2\}$.

Problem 12

Let $\{x_1, x_2, \dots, x_n\}$ be boolean variables. A literal is either a variable or its negation, i.e. x_i or $\overline{x}_i$. A clause is logical OR of one or more distinct literals. The size of a clause is the number of literals in it. A 2CNF formula ϕ is a collection of m clauses (possibly with repetition),

$$\phi = (C_1, C_2, \dots, C_m)$$

where each C_i is of size at most two. Let MAX-2SAT be the following decision problem: Given a pair (ϕ, k) where ϕ is a 2CNF formula with n variables and m clauses, and k is a positive integer such that $k \leq m$, decide whether there exists an assignment to the n boolean variables that satisfies at least k clauses. Show that MAX-2SAT is NP-complete, by giving a polynomial time reduction from VERTEX COVER.

Recall that VERTEX COVER is a problem where, given an undirected graph $G = (V, E)$, with $|V| = n$, and given a positive integer $\ell \leq n$, one needs to decide whether G has a vertex cover of size at most ℓ. A vertex cover is a subset $V' \subseteq V$ such that for every edge in E, at least one of its endpoints is included in V'.

Hint: To every vertex in the graph, assign a boolean variable which is intended to be TRUE if and only if the vertex is included in the vertex cover. Add clauses of size two corresponding to the edges, and clauses of size one corresponding to the vertices. The clauses corresponding to edges may need to be repeated a number of times.