**Carsten Hood** UIN: 922009787

CSCE 411-200 Fall 2016

Homework 6

Due Wed, Dec 7, 11:30 AM

—————————————

**1. Draw the decision tree for bubble-sort with n = 3. Use the code for bubble-sort on page 40 of the textbook.**

—————————————

**2. Consider the problem of finding the median of three integers {a,b,c}.**

**(a) Describe a comparison-based algorithm for solving this problem.**

Compare any two of the integers and determine which is lesser. Then compare this integer to the third. If it is greater than the third, it is the median. Otherwise (when it is less than the other two integers), compare the other two integers to determine which is lesser and hence the median.

if a < b

if a < c

if b < c then return b, else return c

else return a

else

if b < c

if a < c then return a, else return c

else return b

**(b) Draw the decision tree for your algorithm.**

**(c) What is the worst-case number of comparisons taken by your algorithm?**

3 comparisons

—————————————

**3. Exercise 34.1-1.**

Assume LONGEST-PATH is solvable in polynomial time by an algorithm A. Then we can solve the LONGEST-PATH-LENGTH problem in polynomial time by repeatedly running A on the input (G, u, v, k) for increasing values of k (1, 2, … ) until A returns no/false. The smallest k value that returns yes/true constitutes the solution to the LONGEST-PATH-PROBLEM. This process’s complexity is bounded by the number of edges and hence the runtime remains polynomial.

Now assume that LONGEST-PATH-LENGTH is solvable in polynomial time by algorithm B.

To solve an arbitrary instance (G, u, v, k) of LONGEST-PATH, compare the output K of running B on the input (G, u, v) with the k parameter. If k <= K, then the LONGEST-PATH output must be yes/true. Otherwise, it must be false. This process only adds a constant-time comparison to LONGEST-PATH-LENGTH’s polynomial runtime and hence is also solvable in polynomial time.

—————————————

**4. Suppose L1 and L2 are any two languages in P. (Remember that a language is a set of strings over some finite alphabet; suppose L1 and L2 are defined with respect to the same alphabet.) Prove the following:**

Let TM1 and TM2 be TMs that solve L1 and L2, respectively.

**(a) The union of L1 and L2 is also in P.**

To show the union is also in P, we construct a TM, TM3, that decides the union: process the input with TM1 and TM2 step by step. If both reject the input, then return no/false. If either one accepts the input then continue and return true/yes. This also runs in polynomial time.

**(b) The intersection of L1 and L2 is also in P.**

To make TM3 that determines the intersection of L1 and L2: run TM1 on the input. If it accepts the input, run TM2 on the input as well. If TM2 also accepts the input, return yes. Otherwise, if either TM1 or TM2 reject the input, return false.

**(c) The complement of L1 is also in P.**

To make TM3 that determines the complement of L1, run TM1 on the input and return the opposite outcome. Hence, if TM1 validates the input then return NO, and if TM1 rejects the input then return YES. The outcome constitutes the complement and also runs in polynomial time.

—————————————

**5. Given an undirected graph G = (V,E), an *independent set* is a subset V' of V such that there is no edge in E connecting any pair of vertices in V'. The *independent set decision problem* is, given a graph G and an integer k, to determine whether G has an independent set of size at least k. Prove that the independent set decision problem (denoted IS) is NP-complete. For the known NP-complete problem, you may use any one of SAT, 3SAT, HC, TSP, VC, or CLIQUE.**

—————————————

**6. Prove that the 3-COLOR decision problem is NP-complete. Refer to Problem 34-3 for more information about this problem and a description of the reduction to use. You do not need to structure your proof according to parts (d) through (f) if you don't find that helpful. (Skip parts (a) through (c).)**

—————————————

**7. Problem 35-1**