Homework No. 6

CSCI 4470/6470 Algorithms, CS@UGA, Spring 2018

Due Friday April 27, 2018

The answers must be word-processed or typed. You may substitute formulae and figures with hand-writings. Your submitted algorithms should be in the pseudo-code, not in any specific programming language. Answers deviating from these requirements will be returned without grading.

The answers must be the student’s own work. Idea sharing and referencing to others’ work are not allowed. Plagiarism and other forms of academic dishonesty will be handled within the guidelines of the Student Handbook and reported to the University.

There are 100 points in total.

1. (20 points) Consider the 2-SAT problem:

2-SAT

**INPUT:** a boolean formula \( f(x_1, \ldots, x_n) \) in conjunctive form
with exact two literals in every clause;

**OUTPUT:** “yes” if and only if \( f(x_1, \ldots, x_n) \) is satisfiable.

2-SAT is a restricted version of the SAT problem. It has formula instances with exactly two literals in every clause (here a literal is either a boolean variable or its negative form). For example,

\[
f(x_1, \ldots, x_n) = (\neg x_1 \lor x_3) \land (x_2 \lor \neg x_4) \land (x_1 \lor x_2) \land \ldots
\]

Unlike SAT, which is NP-complete, 2-SAT has polynomial-time algorithms. But this question asks you to show to use the **search tree method** (which we used in solving problem MAX INDEPENDENT SET) to design an algorithm for 2-SAT. Such an algorithm can have time upper bound \( T(n) = O(1.62^n) \), where \( n \) is the number of variables in the input formula \( f \).
You are only required to write your algorithm in pseudo-code (no usage of a programming language is allowed) and derive a recurrence for the time complexity $T(n)$ that can lead to complexity $T(n) = O(1.62^n)$.

2. (20 points) Consider the following decision version of MST problem:

**MST Decision:**

**Input:** a graph $G = (V, E)$ with integer edge weights $w$,
  two vertices $s, t \in V$, and integer $k$;

**Output:** “yes” if and only if there exists a path from $s$ to $t$
  with total weight $\leq k$;

(1) Explain why **MST Decision** is in the class $\mathcal{P}$.

(2) Let $A$ be an algorithm that solves problem **MST Decision**.

   Explain how to use $A$ as a subroutine to solve the original
   MST problem, without using any existing algorithm for MST.

3. (20 points) Using the definition of class $\mathcal{NP}$ to prove that problem **MST Decision** is in the class $\mathcal{NP}$.

4. (20 points) Use the reduction from problem **Independent Set** to **Clique** to show the result of reduction for input instance $(G, k)$, where $G$ is shown in the following Figure and $k = 3$:

   ![](graph.png)

5. (20 points) Let $\Pi_1$ and $\Pi_2$ be two decision problems and $\Pi_1 \leq_p \Pi_2$. Explain why the following statements are true or not true:

   (1) either $\Pi_1 \in \mathcal{NP}$ or $\Pi_2 \in \mathcal{NP}$;

   (2) If $\Pi_2$ is polynomial time solvable, so is $\Pi_1$;

   (3) If $\Pi_2$ is $\mathcal{NP}$-complete, then $\Pi_1$ is $\mathcal{NP}$-complete.