CS 4820, Spring 2018	**Homework 8, Problem 2**
Name: Rongguang Wang	
NetID: rw564	
Collaborators: Siyuao Liu (sl2928); Yihao Chen (yc2288)	

(2) *(10 = 2+8 points)* In the PAIRED INTERVAL SCHEDULING problem one is given sets $S_1, \ldots, S_n \subset \mathbb{R}$, each of which is a union of two disjoint closed intervals with distinct integer endpoints. In other words, $S_i = [a_i, b_i] \cup [c_i, d_i]$ where $a_i < b_i < c_i < d_i$ are all integers. One is also given a positive integer $k \leq n$. The problem is to decide whether there exist k pairwise disjoint sets among the collection $\{S_1, \ldots, S_n\}$.

(a) *(2 points)* What is wrong with the following incorrect proof that PAIRED INTERVAL SCHEDULING is NP-complete?

> Faced with an instance of the PAIRED INTERVAL SCHEDULING problem, we can form a *conflict graph* whose vertex set is $\{1, \ldots, n\}$, with an edge joining i to j if S_i and S_j intersect. The problem asks whether this graph contains an independent set of size k. The INDEPENDENT SET problem is NP-Complete, so the PAIRED INTERVAL SCHEDULING problem is also NP-Complete.

Solution:

The proof reduces the Paired Interval Scheduling (PIS) problem to Independent Set problem and claim that PIS problem is NP-Complete since Independent Set problem is NP-Complete. The reduction direction was wrong. The PIS problem should reduce from the Independent Set problem which shows that PIS problem is even harder than Independent Set problem, thereby, PIS problem is definitely NP-Complete.

(b) *(8 points)* Prove that PAIRED INTERVAL SCHEDULING is NP-complete.

Solution:

The problem is aiming to proof that the Paired Interval Scheduling (PIS) problem is NP-Complete. In PIS problem, there are n sets S_i and each of them is a union of two disjoint closed intervals with distinct integer endpoints like $S_i = [a_i, b_i] \cup [c_i, d_i]$ where $a_i < b_i < c_i < d_i$ and they are all integers. The problem is to decide whether there exists k pairwise disjoint sets in the collection $\{S_1, \ldots, S_n\}$ with $k \leq n$.

First, a polynomial-time verifier for PIS problem works as follows. Given an instance of PIS problem specified by a collection of sets S_i and a parameter k, the verifier first sorts the intervals in the collection of sets in increasing order in $O(\log n)$ time and numbering the sets from left to right in ascending order. Then, starting from the first set S_1 of disjoint intervals $[a_1, b_1] \cup [c_1, d_1]$, the verifier record the index of the sets that not conflict with set S_1 into the hash map in $O(n)$ time. The verifier repeat the process form the second set to the last set iteratively in $O(n)$ time, and in total is $O(n^2)$ time. Next, the verifier count the index numbers in the hash map. If the number of the count is equals to or larger than k, the instance is a valid solution to PIS problem. Thus, the verification time is in polynomial time which means PIS problem is in NP.

Next, to prove that PIS problem is NP-Complete, we reduce from 3SAT. Given an instance of 3SAT with n variables and m clauses, we construct an instance of the PIS problem with n variable intervals

and m clause intervals. First, we construct variable gadgets by initiate n pairs of variables x_i and $\bar{x_i}$ and each contains two disjoint intervals. Variables x_i and $\bar{x_i}$ has one conflicting interval $[a_i, b_i]$ and one non-conflicting interval $[c_i, d_i]$ (in x_i) and $[\bar{c_i}, \bar{d_i}]$ (in $\bar{x_i}$). The truth assignment is that variables x_i is true and $\bar{x_i}$ is false. Each pair of the variables in the variable gadget are non-conflicting with each other which means that each pair of the variables are globally unique. As for the clause gadget, there are also n pairs of variables available. Each pair of variables x_j and $\bar{x_j}$ in clause gadget should not conflict with the corresponding variables x_j and $\bar{x_j}$ in variable gadget but they should conflict with the corresponding negated variables $\bar{x_j}$ and x_j in variable gadget. Similar to the variable gadget, each pair of variables x_j and $\bar{x_j}$ in clause gadget share one conflict interval $[a_j, b_j]$ and each has one non-conflicting interval $[c_j, d_j]$ (in x_j) and $[\bar{c_j}, \bar{d_j}]$ (in $\bar{x_j}$). It is noticeable that the shared interval $[a_j, b_j]$ is globally unique. At the same time, interval $[c_j, d_j]$ in variable x_j from clause gadget should be exactly the same as interval $[\bar{c_j}, \bar{d_j}]$ in variable $\bar{x_j}$ from variable gadget. Similarly, interval $[\bar{c_j}, \bar{d_j}]$ in variable $\bar{x_j}$ from clause gadget should be exactly the same as interval $[c_j, d_j]$ in variable x_j from variable gadget. We also define that if either one variable x_k or $\bar{x_k}$ in the same pair appeared more than once in different clauses in the 3SAT, the shared interval $[a_k, b_k]$ should be globally unique to any other intervals, and even different from other x_k or $\bar{x_k}$ pair appeared in other clauses. For each clause j containing three literals, the corresponding clause gadget only need to contain one non-conflicting interval pairs. In another word, the non-conflicting interval pair requirement $n(j) = 1$ for each clause j and the size if each clause is 1. Generating the set of variable interval pairs and set of clause interval pairs takes $O(m + n)$ time, so this is a polynomial-time reduction. To prove the correctness of the reduction, we must show two things.

1. If the 3SAT instance is satisfiable, then there is $m + n$ pairs of intervals that satisfies all of the problem constraints. Given a satisfying truth assignment of the 3SAT formula, there is one pair of intervals do not conflict with any other intervals in the collection if x_i is true or x_i is false for both variable intervals and clause intervals. This means that x_i and $\bar{x_i}$ cannot exist at the same time, but the variable interval and clause interval of x_i or $\bar{x_i}$ can exist at the same time. Furthermore, by our assumption that the truth assignment satisfies at least one literal in each clause, and by our construction of the variable intervals constraint, the scheduling contains n pairs of intervals from the variable gadget and m pairs of intervals from the clause gadget.

2. If there is $m+n$ pairs of intervals that satisfies all of the problem constraints, then there is a satisfying truth assignment of the 3SAT formula. Given a scheduling of intervals, assign x_i true and $\bar{x_i}$ false since both the interval pairs in variable gadget and clause gadget are conflict with each other, respectively. By our construction of the non-conflicting constraint of clause j, we know that if the scheduling includes a pair of intervals in each clauses from clause gadget and n pair of intervals from variable gadget, then the corresponding truth assignment satisfies that there is at least $m + n$ pair of intervals in the collection. Thus, we have constructed a satisfying truth assignment of the 3SAT formula, as desired.