Assignment 5

8-1

(a) Yes. Because the greedy algorithm has used to perform the calculation of *Interval Scheduling* problem such as O(nlogn). Since *Interval Scheduling* can be solved in polynomial time, and so it can also be solved in polynomial time with access to a black box for *Vertex Cover*.

(b) Unknown, because it is equivalent to whether P = NP. If P = NP, then *Independent Set* can be solved in polynomial time, and so *Independent Set* $\leq_P$ *Interval Scheduling*. Conversely, if *Independent Set* $\leq_P$ *Interval Scheduling*, then since *Interval Scheduling* can be sovled in polynomial time, so could *Independent Set*. But *Independent Set* is NP-complete, so solving it in polynomial time would imply P = NP.

8-3

Firstly, for each of the n sports, we can check whether it has at least one counselor skilled for it. So the problem is a NP problem.

Then, we will prove that problem is NP-complete, by a reduction from the *Set Cover*. We can construct an instance of *Efficient Recruiting Problem* as follows: For set U, create a set contains n sports. For each of the m subsets, create a counselor be skilled at the sports that are items of this subset. Obviously, this reduction takes polynomial time. This problem equals to if there are k subsets, whose union is U.

Hence, *Set Cover* $\leq_P$ *Efficient Recruiting Problem*. Therefore, *Efficient Recruiting Problem* is an NP-Complete problem.

8-21

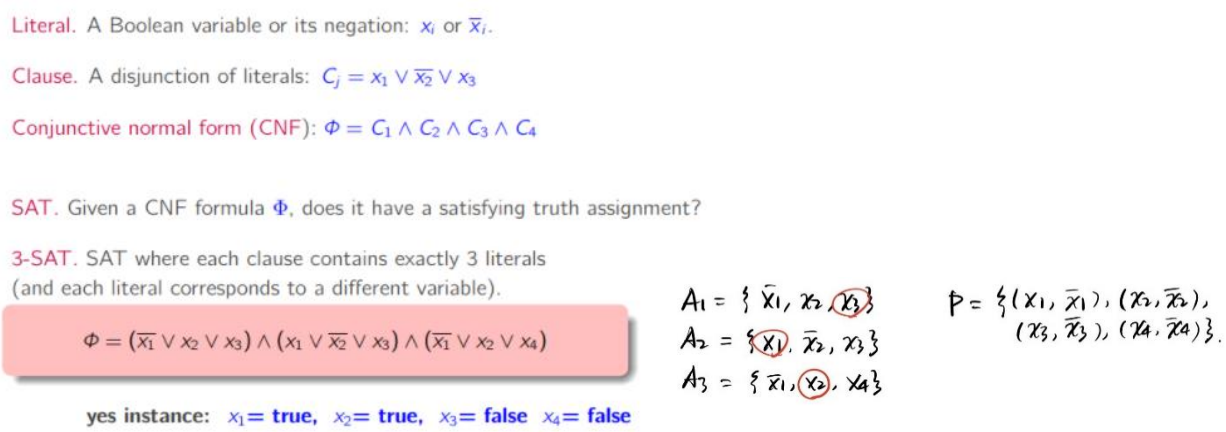
Firstly, it is a NP problem. Because for each incompatible pair in set P, we can check whether elements in this pair both appears in this choice. If true, this choice is not satisfied.

Then, we will prove that problem is NP-complete by a reduction from the *3-SAT*. Given an instance of *3-SAT*, we can construct a set of options for each clause, and each literal in the clause is an option. For incompatible pairs set, we construct each literal and its negation as a pair.

If there is a solution can select options without incompatibilities, then there is a corresponding selection of literal from clauses so that we never choose a variable and its negation. Obviously, this reduction takes polynomial time.

Hence, *3-SAT3* $\leq_P$ *FCC*. Therefore, *FCC* is an NP-Complete problem.

(The graph shows how to construct a *FCC* problem from a *3-SAT* problem)



8-27

It is a NP problem. Because given k sets, we can add the number in each set and sum the square of each set.

To prove this problem is NP-complete, we introduce *Number Partitioning* problem which has been proved to be NP-complete in 8-26.

Number Partitioning Problem: Given a set of positive numbers$\{x_1, x_2, \dots, x_n\}$, and we need to check if they can be divided into two parts, each of which has the same sum.

Then, we will prove this problem is NP-complete by a reduction from the *Number Partitioning*. Given a number set $\{x_1, x_2, \dots, x_n\}$, let $T = \sum_{i=1}^{n} x_i$ be the total sun the all numbers. We can construct an instance of this problem: k = 2, and $B = \frac{1}{2}T^2$.

Suppose it is possible to partition the numbers into two sets such that ${T_1}^2 + {T_2}^2 \leq \frac{1}{2}T^2$.But note that we $T_1 + T_2 = T$, and it is easy to know that ${T_1}^2 + {T_2}^2 \geq \frac{1}{2}T^2$.

The only solution to this $T_1 = T_2 = \frac{1}{2}T$, so these two sets form a solution to the instance of *Number Partitioning.*

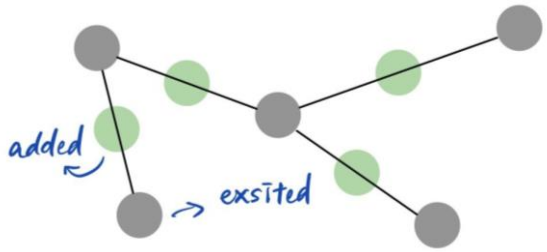
Therefore, this problem NP-Complete.

8-28

The problem is NP since we can exhibit a set of k nodes and check that the distance between all pairs is at least 3.

Then, we will prove that problem is NP-complete by a reduction from *Independent Set*. Let G, k be an input to the independent set problem. We map it to an input to the *Strongly Independent Set* problem. For every edge in G, we subdivide it by adding a new vertex v_e, if $e = (u, v)$ is an edge, then we replace it by two edges (u, v_e), (v_e, v). Further, we form a clique over all the new vertices v_e. Call this new graph G'. First observe that $k > 1$. Now S cannot contain any of the new vertices v_e (because any other vertex in G' is reachable from v_e by a path of length 2).

Therefore, S contains vertices which correspond to those in G. Now check these vertices form an independent set in S.
Hence, *Independent Set* $\leq_P$ *Strongly Independent Set*. Therefore, *Strongly Independent Set* is NP-Complete.



8-31
To begin with, this problem is NP: a solution needs to show a set of vertices S. The verifying algorithm first removes S, and then checks that the resulting graph does not have a cycle (can be done by any graph traversal algorithm).
We reduce from *Vertex Cover*. The Vertex Covering problem is choosing a set of vertices with minimal size in a graph which could cover all the edges in the graph. We only need to construct a two extra vertices for each edge in the Vertex Covering problem and join each of the extra vertex to the original vertices incident to this edge separately, thus each uncovered edge in the original graph would cause a cycle in this new one.
If the oracle for *Vertex Cover* gives a vertex not in the original graph, we could just change it to any of the adjective vertex to shift it to a vertex in the original one. Then we only need to call polynomial times of the oracle for *Vertex Cover* to solve the *Undirected Feedback Set*, with a binary-search on all possible answer for Vertex Covering or just enumerate it.
Hence, *Vertex Cover* $\leq_P$ *Undirected Feedback Set*. Therefore, *Undirected Feedback Set* is NP-Complete.