Lab assignment solutions

Problem 1:

Expected to prove: TSP is NP-complete problem

Assumption: Hamiltonian Cycle problem is an NP-complete problem.

1. **Any HC problem can be reduced to TSP problem**. Proof: Let G be a graph with n vertices, and m edges. The Hamiltonian problem is to find a Hamiltonian Cycle of G. To do this, we create a complete graph of n vertices K_n, where the vertices are those from G, then assign a weight of 0 if an edge in K_n is also in G, and 1 otherwise. Now we have transformed the problem to TSP problem, for the complete graph K_n and k=0. The solution to our instance of TSP problem will also give solution to the previous Hamiltonian Cycle problem because, the cycle found in TSP problem is a Hamiltonian Cycle and will traverse only those edges that have a weight of 0 (otherwise the total cost will be more than k=0).
2. **Any NP-problem R can be reduced to HC**. Proof: This follows from the assumption that HC is a NP-complete and thus NP-hard. And by definition, any NP-problem can be reduced to NP-hard problem in polynomial time.
3. **Any NP-problem R can be reduced to TSP**. Proof: This is true because of the transitive property of reducibility. R is reducible to HC by (2) above, and HC is reducible to TSP by (1), which follows R is reducible to TSP.

Therefore, any NP-problem R can be reduced to TSP (i.e., TSP is NP-hard), and TSP is NP problem (this was proved during the lecture), which concludes the proof that TSP is NP-complete problem.

Problem 2:

a. False. for B to be NPH, any NP problem should be reducible to B, not just one.
b. False. We can find a counter example. Let A be finding a Hamiltonian cycle in a graph of size 3, and B be finding a Hamiltonian cycle in any graph. A is reducible to B, but B is not.
c. True. Assume X is NP-complete and has a polynomial time algorithm solution. Then any NP problem Y, can be reduced to X (because X is NP hard), and the same algorithm used in X can be used to find a solution for Y.
d. False. This is not generally true. For it to be true, B must be an NP problem.

Problem 3:

A graph with only one edge, and two vertices is an example. The Vertex Cover Approximation will give a set with two vertices, but the minimum number of vertices necessary to cover the graph is 1. (2 = 2*1)

Problem 4:

To show that the verification algorithm $B((G, k), V_c)$, runs in polynomial time:

1. $V_c \subseteq \mathrm{V}$, and takes $O(n)$, to show each vertex in the cover is also in the graph
2. $|V_c| \leq k$ takes $O(1)$ to get the size of the vertex cover
3. For each edge e in the graph, one of the vertices of e is in the vertex cover set. Takes $O(2m)$.

So, in total the verification process takes $O(n+m)$ and for the worst case, the graph may have $m = \frac{n(n-1)}{2}$, which makes the asymptotic runtime $O(n^2)$, polynomial time bound.