CS/ECE 374 P32

Abhay Varmaraja, Jiawei Tang, Pengxu Zheng

TOTAL POINTS

31.2 / 100

QUESTION 1

1 32.A. 20 / 20

✓ **+ 6 pts** **Correct reduction from a known NP-hard problem**

✓ **+ 3 pts** **Reduces from a known NP-hard problem, but only one direction of the reduction is incorrect (e.g., yes-instances are mapped to yes-instances, but no-instances may be mapped to yes-instances)**

✓ **+ 6 pts** **(Correct reduction) Proves the reduction takes yes-instances to yes-instances**

✓ **+ 6 pts** **(Correct reduction) Proves the reduction takes no-instances to no-instances**

+ 3 pts (Incorrect reduction) Proves correctness of the reduction in one direction

+ 2 pts Mentions the reduction can be implemented in deterministic polynomial time

+ 5 pts IDK

+ 0 pts Automatic zero: reduction goes in the wrong direction (reduces given problem to a known NP-hard problem)

+ 0 pts Incorrect

QUESTION 2

2 32.B. 11.2 / 40

+ 9.6 pts Correct reduction from a known NP-hard problem

+ 4.8 pts Reduces from a known NP-hard problem, but only one direction of the reduction is incorrect (e.g., yes-instances are mapped to yes-instances, but no-instances may be mapped to yes-instances)

+ 9.6 pts (Correct reduction) Proves the reduction takes yes-instances to yes-instances

+ 9.6 pts (Correct reduction) Proves the reduction takes no-instances to no-instances

+ 4.8 pts (Incorrect reduction) Proves correctness of the reduction in one direction

✓ **+ 3.2 pts** **Mentions the reduction can be implemented in deterministic polynomial time**

+ 0 pts Automatic zero for NP-hardness: reduction goes in the wrong direction (reduces given problem to a known NP-hard problem)

✓ **+ 4 pts** **Describes/sketches a correct verification procedure for yes-instances**

✓ **+ 4 pts** **Mentions the verifier can be implemented in polynomial time**

+ 10 pts IDK

+ 0 pts Incorrect

QUESTION 3

3 32.C. 0 / 40

+ 12 pts Correct reduction from a known NP-hard problem

+ 6 pts Reduces from a known NP-hard problem, but only one direction of the reduction is incorrect (e.g., yes-instances are mapped to yes-instances, but no-instances may be mapped to yes-instances)

+ 12 pts (Correct reduction) Proves the reduction takes yes-instances to yes-instances

+ 12 pts (Correct reduction) Proves the reduction takes no-instances to no-instances

+ 6 pts (Incorrect reduction) Proves correctness of the reduction in one direction

+ 4 pts Mentions the reduction can be implemented in deterministic polynomial time

+ 10 pts IDK

+ 0 pts Automatic zero: reduction goes in the wrong direction (reduces given problem to a known NP-hard problem)

✓ **+ 0 pts** **Incorrect**

HW11 Solution

CS/ECE 374: Algorithms & Models of Computation, Spring 2019 Version: **1.0**

Submitted by:

- ≪**Pengxu Zheng**≫: ≪**pzheng5**≫
- ≪**Jiawei Tang**≫: ≪**jiaweit2**≫
- ≪**Abhay Varmaraja**≫: ≪**abhaymv2**≫

(Discussed with Heng's Group)

Solution: 32.A.

We describe this problem as a transformation of the cliques problem. Let G(V, E) be the graph with its vertices representing prisoners and each edge between two vertices represents the connected prisoners are enemies. The goal is equivalent to finding the largest complete sub-graph of G(V, E) such that the size of the subgraph is less than or equal to k.

We claim that there exist one and only one vertex that is in the largest complete subgraph. We prove it using proof by contradiction:

For the sake of contradiction, suppose there exist two prisoners that are in the same block. Since all vertices are connected with each other (assumption from the clique problem), there should be an edge connecting two prisoners in the same block. However, it is given that two prisoners can't co-exist in the same block if there was an edge between them (aka. they are enemies). Thus, the claim holds. Thus, for each of the k blocks, if there is no more than one prisoner in the clique, there should be no more than k vertices in the largest complete subgraph. For runtime, since we only built a graph using the exact number of prisoners and added edges, the transformation should take less than polynomial time. Therefore, we argue that the problem is NP-Hard, based on the transition of the cliques problem.

32.B.

To prove this problem (namaed as Q) is NP-Complete, we prove the two facts:

1. The problem Q $\in$ NP.
2. The Hamiltonian cycle problem HC $\leq_p$ P.

We use a n algorithm described as follows to determine if a path is a HC and the sum of edge weights is 0:

Loop through the path, and check if there are repeating vertices. If repeating vertices are detected, return false. Use a variable to sum the total weights. If the total edge weight is not equal to 0 after looping through all vertices, return false. Use another variable to keep track of how many vertices are visited. If the number visited is less than |V|, return false. Otherwise, return true.

32

1 32.A. **20 / 20**

✓ **+ 6 pts** **Correct reduction from a known NP-hard problem**

✓ **+ 3 pts** **Reduces from a known NP-hard problem, but only one direction of the reduction is incorrect (e.g., yes-instances are mapped to yes-instances, but no-instances may be mapped to yes-instances)**

✓ **+ 6 pts** **(Correct reduction) Proves the reduction takes yes-instances to yes-instances**

✓ **+ 6 pts** **(Correct reduction) Proves the reduction takes no-instances to no-instances**

+ 3 pts (Incorrect reduction) Proves correctness of the reduction in one direction

+ 2 pts Mentions the reduction can be implemented in deterministic polynomial time

+ 5 pts IDK

+ 0 pts Automatic zero: reduction goes in the wrong direction (reduces given problem to a known NP-hard problem)

+ 0 pts Incorrect

2 32.B. **11.2 / 40**

+ **9.6 pts** Correct reduction from a known NP-hard problem

+ **4.8 pts** Reduces from a known NP-hard problem, but only one direction of the reduction is incorrect (e.g., yes-instances are mapped to yes-instances, but no-instances may be mapped to yes-instances)

+ **9.6 pts** (Correct reduction) Proves the reduction takes yes-instances to yes-instances

+ **9.6 pts** (Correct reduction) Proves the reduction takes no-instances to no-instances

+ **4.8 pts** (Incorrect reduction) Proves correctness of the reduction in one direction

✓ **+ 3.2 pts Mentions the reduction can be implemented in deterministic polynomial time**

+ **0 pts** Automatic zero for NP-hardness: reduction goes in the wrong direction (reduces given problem to a known NP-hard problem)

✓ **+ 4 pts Describes/sketches a correct verification procedure for yes-instances**

✓ **+ 4 pts Mentions the verifier can be implemented in polynomial time**

+ **10 pts** IDK

+ **0 pts** Incorrect

3 32.C. **0 / 40**

+ **12 pts** Correct reduction from a known NP-hard problem

+ **6 pts** Reduces from a known NP-hard problem, but only one direction of the reduction is incorrect (e.g., yes-instances are mapped to yes-instances, but no-instances may be mapped to yes-instances)

+ **12 pts** (Correct reduction) Proves the reduction takes yes-instances to yes-instances

+ **12 pts** (Correct reduction) Proves the reduction takes no-instances to no-instances

+ **6 pts** (Incorrect reduction) Proves correctness of the reduction in one direction

+ **4 pts** Mentions the reduction can be implemented in deterministic polynomial time

+ **10 pts** IDK

+ **0 pts** Automatic zero: reduction goes in the wrong direction (reduces given problem to a known NP-hard problem)

✓ **+ 0 pts Incorrect**