HW 10: Extra problems

Instructor: Sariel Har-Peled

CS/ECE 374: Algorithms & Models of Computation, Fall 2017 Version: **1.0**

1 A subset S of vertices in an undirected graph G is called ***almost independent*** if at most 374 edges in G have both endpoints in S. Prove that finding the size of the largest almost-independent set of vertices in a given undirected graph is NP-hard.

2 A subset S of vertices in an undirected graph G is called ***triangle-free*** if, for every triple of vertices $u, v, w \in S$, at least one of the three edges uv, uw, vw is *absent* from G. Prove that finding the size of the largest triangle-free subset of vertices in a given undirected graph is NP-hard.

3 Charon needs to ferry n recently deceased people across the river Acheron into Hades. Certain pairs of these people are sworn enemies, who cannot be together on either side of the river unless Charon is also present. (If two enemies are left alone, one will steal the obol from the others mouth, leaving them to wander the banks of the Acheron as a ghost for all eternity. Lets just say this is a Very Bad Thing.) The ferry can hold at most k passengers at a time, including Charon, and only Charon can pilot the ferry.

Prove that it is NP-hard to decide whether Charon can ferry all n people across the Acheron unharmed.[1] The input for Charons problem consists of the integers k and n and an n-vertex graph G describing the pairs of enemies. The output is either TRUE or FALSE.

This problem is a generalization of the following extremely well-known puzzle, whose first known appearance is in the treatise *Propositiones ad Acuendos Juvenes* [*Problems to Sharpen the Young*] by the 8th-century English scholar Alcuin of York.[2]

> XVIII. PROPOSITIO DE HOMINE ET CAPRA ET LVPO.
>
> Homo quidam debebat ultra fluuium transferre lupum, capram, et fasciculum cauli. Et non potuit aliam nauem inuenire, nisi quae duos tantum ex ipsis ferre ualebat. Praeceptum itaque ei fuerat, ut omnia haec ultra illaesa omnino transferret. Dicat, qui potest, quomodo eis illaesis transire potuit?
>
> **Solutio.** Simili namque tenore ducerem prius capram et dimitterem foris lupum et caulum. Tum deinde uenirem, lupumque transferrem: lupoque foris misso capram naui receptam ultra reducerem; capramque foris missam caulum transueherem ultra; atque iterum remigassem, capramque assumptam ultra duxissem. Sicque faciendo facta erit remigatio salubris, absque uoragine lacerationis.

In case your classical Latin is rusty, here is an English translation:

> XVIII. THE PROBLEM OF THE MAN, THE GOAT, AND THE WOLF.
>
> A man needed to transfer a wolf, a goat, and a bundle of cabbage across a river. However, he found that his boat could only bear the weight of two [objects at a time, including the man]. And he had to get everything across unharmed. Tell me if you can: How they were able to cross unharmed?
>
> **Solution.** In a similar fashion [as an earlier problem], I would first take the goat across and leave the wolf and cabbage on the opposite bank. Then I would take the wolf across; leaving the wolf on shore, I would retrieve the goat and bring it back again. Then I would leave the goat and take the cabbage across. And then I would row across again and get the goat. In this way the crossing would go well, without any threat of slaughter.

Please do not write your solution to problem 3 in classical Latin.

[1] Aside from being, you know, dead.

[2] At least, we *think* thats who wrote it; the evidence for his authorship is rather circumstantial, although we do know from his correspondence with Charlemagne that he sent the emperor some simple arithmetical problems for fun. Most scholars believe that even if Alcuin is the actual author of the *Propositiones*, he didnt come up with the problems himself, but just collected his problems from other sources. Some things never change.

4 Consider an instance of the Satisfiability Problem, specified by clauses $C_1, \ldots, C_k$ over a set of Boolean variables $x_1, \ldots, x_n$. We say that the instance is *monotone* if each term in each clause consists of a nonnegated variable; that is each term is equal to x_i, for some i, rather than $\bar{x}_i$. Monotone instance of Satisfiability are very easy to solve: They are always satisfiable, by setting each variable equal to 1.

For example, suppose we have the three clauses

$$(x_1 \vee x_2), (x_1 \vee x_3), (x_2 \vee x_3)$$

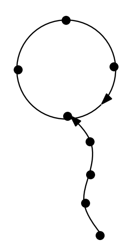
This is monotone, and indeed the assignment that sets all three variables to 1 satisfies all the clauses. But we can observe that this is not the only satisfying assignment; we could also have set x_1 and x_2 to 1 and x_3 to 0. Indeed, for any monotone instance, it is natural to ask how few variables we need to set to 1 in order to satisfy it.

Given a monotone instance of Satisfiability, together with a number k, the problem of *Monotone Satisfiability with Few True Variables* asks: Is there a satisfying assignment for the instance in which at most k variables are set to 1? Prove that this problem is NP-Complete. *Hint:* Reduce from Vertex Cover.

5 Given an undirected graph $G = (V, E)$, a partition of V into $V_1, V_2, \ldots, V_k$ is said to be a clique cover of size k if each V_i is a clique in G. Prove that the problem of deciding whether G has a clique cover of size at most k is NP-Complete. *Hint:* Consider the complement of G.

6 Given an undirected graph $G = (V, E)$ a *matching* in G is a set of edges $M \subseteq E$ such that no two edges in M share a node. A matching M is *perfect* if $2|M| = |V|$, in other words if every node is incident to some edge of M. PerfectMatching is the following decision problem: does a given graph G have a perfect matching? Describe a polynomial-time reduction from PerfectMatching to SAT. Does this problem that PerfectMatching is a difficult problem?

7 A balloon is a directed graph on an even number of nodes, say $2n$, in which n of the nodes form a directed cycle and the remaining n vertices are connected in a "tail" that consists of a directed path joined to one of the nodes in the cycle. See figure below for a balloon with 8 nodes.



Given a directed graph G and an integer k, the BALLOON problem asks whether or not there exists a subgraph which is a baloon that contains $2k$ nodes. Prove that BALLOON is NP-Complete.

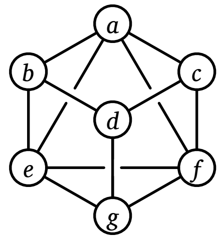
8 Consider the following problem. You are managing a communication network, modeled by a directed graph $G = (V, E)$. There are c *users* who are interested in making use of this network. User i (for each $i = 1, 2, \ldots, c$) issues a *request* to reserve a specific path P_i in G on which to transmit data.

You are interested in accepting as many of these path requests as possible, subject to the following restriction: if you accept both P_i and P_j, then P_i and P_j can not share any modes.

Thus the *Path Selection Problem* asks: Given a directed graph $G = (V, E)$, a set of requests $P_1, \ldots, P_c$-each of which must be a path in G- and a number k, is it possible to select at least k of the paths so that no two of the selected paths share any nodes?

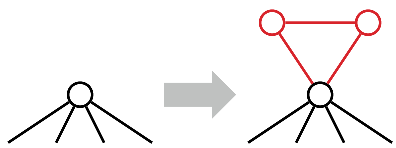
Prove that the Path Selection is NP-Complete.

9 A *double-Hamiltonian tour* in an undirected graph G is a closed walk that visits every vertex in G exactly twice. Prove that it is NP-hard to decide whether a given graph G has a double-Hamiltonian tour.



This graph contains the double-Hamiltonian tour $a \to b \to d \to g \to e \to b \to d \to c \to f \to a \to c \to f \to g \to e \to a$.

Solution: We prove the problem is NP-hard with a reduction from the standard Hamiltonian cycle problem. Let G be an arbitrary undirected graph. We construct a new graph H by attaching a small gadget to every vertex of G. Specifically, for each vertex v, we add two vertices $v^\sharp$ and $v^\flat$, along with three edges $vv^\flat$, $vv^\sharp$, and $v^\flat v^\sharp$.



A vertex in G, and the corresponding vertex gadget in H.

I claim that G has a Hamiltonian cycle if and only if H has a double-Hamiltonian tour.

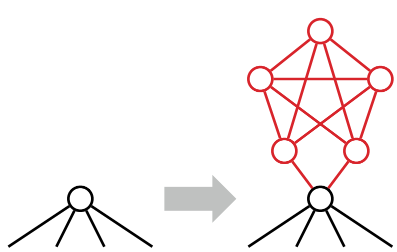
$\Longrightarrow$ Suppose G has a Hamiltonian cycle $v_1 \to v_2 \to \cdots \to v_n \to v_1$. We can construct a double-Hamiltonian tour of H by replacing each vertex v_i with the following walk:

$$\cdots \to v_i \to v_i^\flat \to v_i^\sharp \to v_i^\flat \to v_i^\sharp \to v_i \to \cdots$$

$\Longleftarrow$ Conversely, suppose H has a double-Hamiltonian tour D. Consider any vertex v in the original graph G; the tour D must visit v exactly twice. Those two visits split D into two closed walks, each of which visits v exactly once. Any walk from $v^\flat$ or $v^\sharp$ to any other vertex in H must pass through v. Thus, one of the two closed walks visits only the vertices v, $v^\flat$, and $v^\sharp$. Thus, if we simply remove the vertices in $H \setminus G$ from D, we obtain a closed walk in G that visits every vertex in G once.

Given any graph G, we can clearly construct the corresponding graph H in polynomial time.

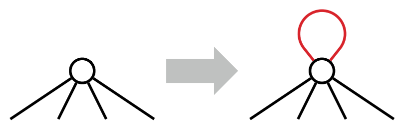
With more effort, we can construct a graph H that contains a double-Hamiltonian tour ***that traverses each edge of H at most once*** if and only if G contains a Hamiltonian cycle. For each vertex v in G we attach a more complex gadget containing five vertices and eleven edges, as shown on the next page.



A vertex in G, and the corresponding modified vertex gadget in H.

Solution: **Bad and incorrect solution!!!**

We attempt to prove the problem is NP-hard with a reduction from the Hamiltonian cycle problem. Let G be an arbitrary undirected graph. We construct a new graph H by attaching a self-loop every vertex of G. Given any graph G, we can clearly construct the corresponding graph H in polynomial time.

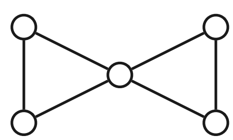


An incorrect vertex gadget.

Suppose G has a Hamiltonian cycle $v_1 \to v_2 \to \cdots \to v_n \to v_1$. We can construct a double-Hamiltonian tour of H by alternating between edges of the Hamiltonian cycle and self-loops:

$$v_1 \to v_1 \to v_2 \to v_2 \to v_3 \to \cdots \to v_n \to v_n \to v_1.$$

On the other hand, if H has a double-Hamiltonian tour, we *cannot* conclude that G has a Hamiltonian cycle, because we cannot guarantee that a double-Hamiltonian tour in H uses *any* self-loops. The graph G shown below is a counterexample; it has a double-Hamiltonian tour (even before adding self-loops) but no Hamiltonian cycle.



This graph has a double-Hamiltonian tour.

Rubric:[for all polynomial-time reductions] 10 points =

+ 3 points for the reduction itself
  - For an NP-hardness proof, the reduction must be from a known NP-hard problem. You can use any of the NP-hard problems listed in the lecture notes (except the one you are trying to prove NP-hard, of course).
+ 3 points for the if proof of correctness
+ 3 points for the only if proof of correctness
+ 1 point for writing polynomial time

- An incorrect polynomial-time reduction that still satisfies half of the correctness proof is worth at most 4/10.
- A reduction in the wrong direction is worth 0/10.