Problem Set 5

> *Which of the following statements cannot be true, given the current state of knowledge?*
>
> - *Some NP-complete problems are polynomial-time solvable, and some NP-complete problems are not polynomial-time solvable.*
> - *There is an NP-complete problem that is polynomial-time solvable.*
> - *There is an NP-complete problem that can be solved in $O(n^{\log n})$ time, where n is the size of the input.*
> - *There is no NP-complete problem that can be solved in $O(n^{\log n})$ time, where n is the size of the input.*

ANSWER: Quoting Wikipedia (https://en.wikipedia.org/wiki/Time_complexity):

> ***polynomial time**: An algorithm is said to be of polynomial time if its running time is upper bounded by a polynomial expression in the size of the input for the algorithm, i.e., $T(n) = O(n^k)$ for some positive constant k.*

> ***exponential time**: An algorithm is said to be exponential time, if $T(n)$ is upper bounded by $2^{poly(n)}$, where $poly(n)$ is some polynomial in n. More formally, an algorithm is exponential time if $T(n)$ is bounded by $O(2^{n^k})$ for some constant k.*

We don't know if P = NP, so option 2 can't be ruled out.

For all we know, the running time required to solve NP-complete problems could be anywhere between polynomial and exponential (note that $n^{\log n}$ is more than polynomial but less than exponential). Thus, options 3 and 4 can't be ruled out.

That only leaves option 1. The claim in option 1 is incorrect because if there's a polynomial-time algorithm for any NP-complete problem X, then every other NP problem has a polynomial-time algorithm by reduction to X.

> *Choose the strongest true statement.*
>
> - *If the minimum-size vertex cover problem can be solved in time $O(T(n))$ in bipartite graphs, then the maximum-size independent set problem can be solved in time $O(T(n))$ in bipartite graphs.*
> - *If the maximum-size vertex cover problem can be solved in time $O(T(n))$ in general graphs, then the minimum-size independent set problem can be solved in time $O(T(n))$ in general graphs.*
> - *If the minimum-size vertex cover problem can be solved in time $O(T(n))$ in general graphs, then the maximum-size independent set problem can be solved in time $O(T(n))$ in general graphs.*
> - *All three of the other assertions are true.*

ANSWER: Option 4 is correct. Vertex covers and independent sets are complements (take the complement of one and you get the other). Thus solving one problem allows you to solve the other with $\Theta(n)$ postprocessing, where n is the number of vertices.

Which of the following statements is true?

- Consider a TSP instance in which every edge cost is either 1 or 2. Then an optimal tour can be computed in polynomial time.
- Consider a TSP instance in which every edge cost is negative. The dynamic programming algorithm covered in the video lectures might not correctly compute the optimal (i.e., minimum sum of edge lengths) tour of this instance.
- Consider a TSP instance in which every edge cost is negative. Deleting a vertex and all of its incident edges cannot increase the cost of the optimal (i.e., minimum sum of edge lengths) tour.
- Consider a TSP instance in which every edge cost is the Euclidean distance between two points in the place (just like in Programming Assignment #5). Deleting a vertex and all of its incident edges cannot increase the cost of the optimal (i.e., minimum sum of edge lengths) tour.

ANSWER: Papadimitriou and Yannakakis (https://www.jstor.org/stable/3690150?seq=1#page_scan_tab_contents) have shown that it is possible to approximate the TSP problem 1 in polynomial time within accuracy $\frac{7}{6}$. This guarantee has been further improved by Bläser and Shankar Ram (https://link.springer.com/chapter/10.1007/11537311_44) to $\frac{65}{56}$. However, no matter how good those results are, they are still approximations, and not an optimal solution. Thus, option 1 is incorrect.

The DP algorithm doesn't make any assumptions on the edge costs, so option 2 is incorrect.

If all edge weights are negative, then deleting a vertex and all of its incident edges can certainly increase the minimum sum because in effect, that edge weight is now added to the previous minimum. Thus, option 3 is incorrect.

Take the optimal tour in the original instance. Now, instead of visiting the deleted vertex v, skip straight from ‘s predecessor to its successor on the tour. Because Euclidean distance satisfies the Triangle Inequality (https://en.wikipedia.org/wiki/Triangle_inequality), this shortcut only decreases the overall distance traveled. The best tour can of course only be better. Thus, option 4 is correct.

> *Let TSP1 denote the following problem: given a TSP instance in which all edge costs are positive integers, compute the value of an optimal TSP tour. Let TSP2 denote: given a TSP instance in which all edge costs are positive integers, and a positive integer T, decide whether or not there is a TSP tour with total length at most T. Let HAM1 denote: given an undirected graph, either return the edges of a Hamiltonian cycle (a cycle that visits every vertex exactly once), or correctly decide that the graph has no such cycle. Let HAM2 denote: given an undirected graph, decide whether or not the graph contains at least one Hamiltonian cycle.*
>
> - *If TSP2 is polynomial-time solvable, then so is TSP1. If HAM2 is polynomial-time solvable, then so is HAM1.*
> - *Polynomial-time solvability of TSP2 does not necessarily imply polynomial-time solvability of TSP1. Polynomial-time solvability of HAM2 does not necessarily imply polynomial-time solvability of HAM1.*
> - *If TSP2 is polynomial-time solvable, then so is TSP1. But, polynomial-time solvability of HAM2 does not necessarily imply polynomial-time solvability of HAM1.*
> - *Polynomial-time solvability of TSP2 does not necessarily imply polynomial-time solvability of TSP1. But, if HAM2 is polynomial-time solvable, then so is HAM1.*

ANSWER: The decision problems are at least no harder than the corresponding optimization problems. For TSP2, we can increment T by one and check whether it’s “Yes” or “No” using the algorithm for TSP2. Another method is to double T every iteration initially and then do binary search once a boundary is found.

HAM2 can be reduced to TSA2 as follows: Construct a complete graph $G'(V', E')$ s.t. $V' = V$ and $e \in E'$ as follows:

$$cost(e) = \begin{cases} 1 \text{ if } (u, v) \in E \\ 2 \text{ otherwise} \end{cases}$$

Note that construction of the graph G' can be done in polynomial-time.

G has a Hamiltonian Cycle if and only if there exists a cycle in G' passing through all the vertices exactly once, and that has a length $\leq n$ (i.e. has a solution for TSP2 where $T = n$).

Proof: If there is a cycle that passes through all the vertices exactly once, and has length $\leq n$ in G', the cycle contains only edges that were originally present in G (the new edges in G' have cost 2 and hence cannot be part of a cycle of length $\leq n$). Hence there exists a Hamiltonian Cycle in G.

Conversely, if there exists a Hamiltonian cycle in G, it forms a cycle in G' with length n, since the cost of all the edges is n. Hence there exists a solution for TSP2 in G' with length $\leq n$.

Thus, option 1 is correct.

> *Assume that $P \neq NP$. Consider undirected graphs with nonnegative edge lengths. Which of the following problems can be solved in polynomial time?*
>
> *Hint: The Hamiltonian path problem is: given an undirected graph with n vertices, decide whether or not there is a (cycle-free) path with $n - 1$ edges that visits every vertex exactly once. You can use the fact that the Hamiltonian path problem is NP-complete. There are relatively simple reductions from the Hamiltonian path problem to 3 of the 4 problems below.*
>
> - *For a given source s and destination t, compute the length of a shortest $s - t$ path that has exactly $n - 1$ edges (or $+\infty$, if no such path exists). The path is allowed to contain cycles.*
> - *Amongst all spanning trees of the graph, compute one with the smallest-possible number of leaves.*
> - *Amongst all spanning trees of the graph, compute one with the minimum-possible maximum degree. (Recall the degree of a vertex is the number of incident edges.)*
> - *For a given source s and destination t, compute the length of a shortest $s - t$ path that has exactly $n - 1$ edges (or $+\infty$, if no such path exists). The path is not allowed to contain cycles.*

ANSWER: Notice that a Hamiltonian path is a spanning tree of a graph and only has two leaf nodes, and that any spanning tree of a graph with exactly two leaf nodes must be a Hamiltonian path. That means that the NP-hard problem of determining whether a Hamiltonian path exists in a graph can be solved by finding the minimum-leaf spanning tree of the graph: the path exists if and only if the minimum-leaf spanning tree has exactly two leaves.

Problem 3 is NP-Hard; see this paper.

If we could solve problem 4, we could solve the Hamiltonian path problem by invoking this algorithm for all $\Theta(n^2)$ choices of s and t.

Problem 1 can be solved by a Bellman-Ford type recurrence in polynomial time. The basic intuition is that it reduces the amount of state we need to track. With cycles allowed we just need to build up a table of the shortest length-k path from s to each other vertex. With cycles disallowed we need to track all of the intermediate vertices, thus creating an exponential number of subproblems. Also see this (https://cs.stackexchange.com/q/102558/95996) discussion.