

CSE 5311 Homework #3 Solutions

1 Rectangle-Fit Problem is NP-Complete

This problem can be reduced from the bin-packing problem. The rectangle-fit problem is obviously in NP because it can easily be verified in polynomial time whether a given set of rectangles fits within another given rectangle with dimensions x and y .

To prove that it is NP-hard, we reduce it from the bin-packing problem in the following manner. The bin-packing problem says that given a set of objects, each with size ≥ 1 , then k is the minimum number of unit-sized bins needed to pack all the objects. There is a direct relationship between the rectangle-fit problem and the bin-packing problem as can be seen from the diagram given in Figure 1.

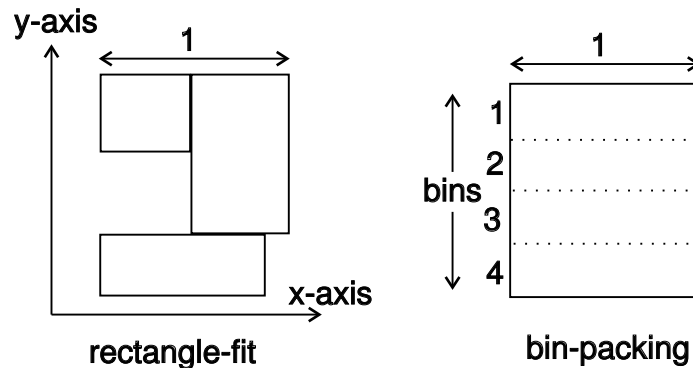


Figure 1: Rectangle Fit and Bin Packing

The reduction is obvious from Figure 1. We set dimension x of the minimum sized rectangle needed to fit all the other rectangles equal to unit size. Dimension y is given by the number (k) of bins required to cover the total width of the set of rectangles.

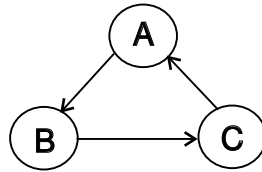
Thus, the rectangle-fit problem will have a covering rectangle of dimensions x and y if and only if the bin-packing problem packs all the object into k bins of size x such that $kw = y$, where w is the width of each bin. Thus, the rectangle-fit problem is NP-complete.

2 Graph Transitivity Problem is NP-Complete

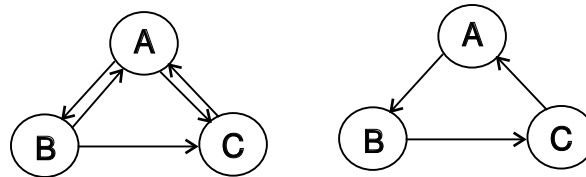
To prove that the problem is NP-complete we prove that it is in NP and that it is NP-hard by reducing from another NP-complete problem.

The problem is obviously in NP because it can be easily verified in polynomial time whether all the directed paths are preserved after removing the given k edges.

To prove that the problem is NP-hard, it is reduced from the Directed Hamiltonian Circuit problem. The Directed Hamiltonian Circuit problem says that given a directed graph G , the graph has a simple directed cycle containing each vertex in G .



As can be seen from the preceding diagram of a directed Hamiltonian circuit, the number of edges in such a cycle is equal to the number of vertices in the graph. Hence the reduction is as follows. The given problem can have k out of $|E|$ edges removed while still preserving all directed paths if and only if the corresponding Directed Hamiltonian Circuit has a simple cycle of size $|V|$, where $k = |E| - |V|$. So, for the example given below, $k = 2$ is the optimal (maximum) number of edges that can be removed while still preserving all directed paths from the original graph. Thus, this problem is NP-complete.



3 Problem 36-1, p. 961

a) INDEPENDENT-SET = $\{ \langle G, K \rangle \mid \text{graph } G \text{ has an independent set of size } k \}$.

To prove this problem is NP-complete we first show that the independent-set problem is in NP. Suppose we are given a graph $G = (V, E)$ and an integer k . The certificate we choose is the independent set $V' \subseteq V$ itself. The verification algorithm affirms that $|V'| = k$ and then for each edge $(u, v) \in E$ it checks that at most one of u and v is in V' . This verification can be performed straight-forwardly in polynomial time.

We prove that the independent-set problem is NP-hard by showing that CLIQUE \propto INDEPENDENT-SET. This reduction is based on the notion of the “complement”

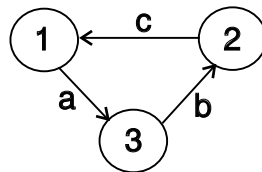
of a graph. Given an undirected graph $G = (V, E)$, we define the complement of G as $\overline{G} = (V, \overline{E})$, where $\overline{E} = \{(u, v) \mid (u, v) \notin E\}$.

The reduction algorithm takes as input an instance $\langle G, K \rangle$ of the clique problem. It computes the complement \overline{G} , which is easily obtained in polynomial time. The output of the reduction algorithm is the instance $\langle \overline{G}, K \rangle$ of the independent-set problem with the same k vertices in the independent set as in the clique. The proof is very similar to the vertex-cover problem proof on p. 950. Using the same example as given in the book (figure 36.13 on p. 949) we have G yielding the clique $V' = \{u, v, x, y\}$ and \overline{G} yielding the independent set $V' = \{u, v, x, y\}$.

- b) Not done.
- c) It can be observed that when each vertex in G has degree 2, the graph is just a simple cycle. In this case, an independent set of maximum size can be obtained by starting at any vertex and picking each alternate vertex on the cycle until the size of the independent set is $\lfloor |V|/2 \rfloor$. The running time of this algorithm is obviously $O(|V|)$ (or $O(|E|)$ since $|V| = |E|$ in this case).
- d) Using figure 27.8 on p. 601, it can be seen that when the graph G is bipartite, then the independent-set (maximum size) is the side with the larger number of vertices (the set of vertices in L). The running time will be $O(|V|)$.

4 Problem 37.2-1, p. 973

For each combination of 3 vertices $\binom{n}{3}$, calculate $x = c - a - b$ where c, a, b are the cost of the edges.



If $x < 0$, ignore. Otherwise keep track of $(x)_{\max}$. Add $(x)_{\max}$ to the cost of each edge. The resulting traveling-salesperson problem (TSP) satisfies the triangle inequality i.e., $c(u, w) \leq c(u, v) + c(v, w)$. Using the example from p. 959 (reproduced here in figure 2) we find that $(x)_{\max} = 5 - 1 - 2 = 2$. Adding 2 to each edge yields the second graph in figure 2 which satisfies the triangle inequality. Since the same number is added to each edge, the two instances will have the same set of optimal tours. This transformation takes polynomial time because $\binom{n}{3} = O(n^3)$.

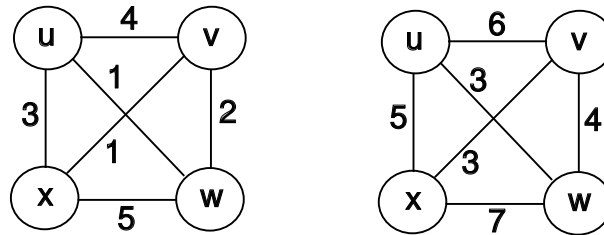


Figure 2: Triangular Inequalities

Now we explain why such a polynomial-time transformation does not contradict theorem 37.3 (p. 972), assuming that $P \neq NP$. The TSP with triangle inequality has a polynomial-time approximation algorithm with a ratio bound $\rho \geq 1$ i.e., $\text{cost}(\Delta) \leq \rho \cdot \text{opt}(\Delta)$, where $\text{opt}(\Delta)$ denotes the cost of an optimal tour for the TSP with triangle inequality. So for the original TSP without triangle inequality (denoted by Δ) we can write $\text{cost}(\Delta) \leq \rho \cdot \text{opt}(\Delta)$ or $\text{cost}(\Delta) - X|V| \leq \rho(\text{opt}(\Delta) - X|V|)$, where X is the constant added to each edge to convert to triangle inequality and hence $X|V|$ is the total increase in cost for the triangle inequality TSP. The above equation can be rewritten as $\text{cost}(\Delta) \leq \rho \cdot \text{opt}(\Delta) - \underbrace{(\rho - 1)X|V|}_{\text{positive for } \rho > 1}$, which may or may not be true. Hence the

transformation does not contradict theorem 37.3.

5 Problem 37.2-2, p. 973

Figure 3 shows how the closest-point heuristic works on the graph used in the example on p. 971.

From the manner in which nodes are added to the cycle in this heuristic, it can be observed (comparing with figure 37.2 on p. 971) that the final cycle has an inherent MST within it and that its length will be less than the length of the full walk which traverses every edge of the MST twice. Hence, from the analysis in theorem 37.2 (p. 970) we have $c(\text{closest-point heuristic}) \leq c(w)$ and hence $c(\text{closest-point heuristic}) \leq 2c(H^*)$ where $c(H^*)$ is the cost of the optimal tour. Hence the closest-point heuristic has a ration bound of 2.

6 Problem 37.2-4, p. 974

We are given that the vertices of the TSP are points in the plane and the cost $c(u, v)$ is the euclidean distance between point u and v . Figure 4 gives an example where the solid lines indicate a TSP optimal tour. Using triangle inequality, $z \leq x + y$ and $c \leq a + b$. Hence replacing $x + b$ and $a + y$ with c and z gives us a shorter tour which contradicts

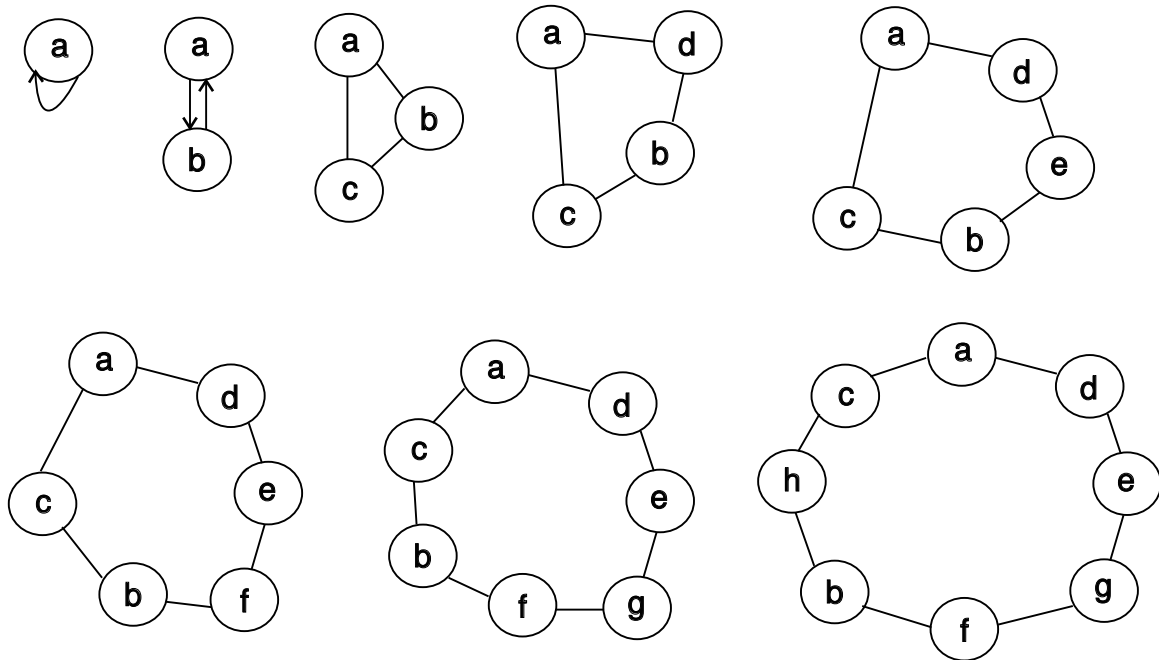


Figure 3: Closest-Point Heuristic

the assumption that the original tour was an optimal tour. Hence an optimal tour will never cross itself.

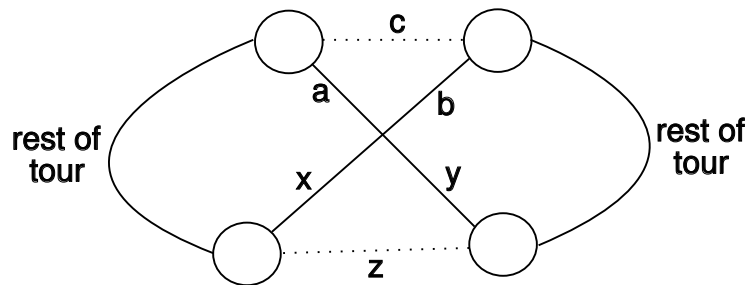


Figure 4: Travelling Salesperson Problem

7 Problem 37.3-1, p. 978

Here $X = \{a, d, e, h, i, l, n, o, r, s, t, u\}$ (see p. 974–975) and each of the given words is a subset of X . The cover produced by GREEDY-SET-COVER when ties are broken in favor of the word appears first in the dictionary is: thread, lost, drain, shun.

8 Problem 37.3-2, p. 978

The decision version of the set-covering problem is:

$$\text{SET-COVER} = \{ \langle X, F, k \rangle \mid \text{set } X \text{ has a set cover of size } k \}$$

where X and F are as defined on p. 974.

To show that this problem is NP-complete, we first show that SET-COVER \in NP. Suppose we are given a set X , a family F of subsets of X , and an integer k . The certificate we choose is the set cover $C' \subseteq F$ itself. The verification algorithm affirms that $|C'| = k$ (i.e., the number of subsets in $C' = k$) and then it checks whether each element of X is in at least one of the subset of C' . This verification can be performed in polynomial time (with a proof very similar to the vertex-cover problem proof on p. 950).

We now prove that the set-covering problem is NP-hard by showing that VERTEX-COVER \propto SET-COVER. The reduction is a direct mapping from VERTEX-COVER to SET-COVER.

$X = E$ i.e., the elements of X in SET-COVER are the edges $(u, v) \in E$ in VERTEX-COVER.

$|F| = |V|$ i.e. the number of subsets in the family F of subsets in SET-COVER is equal to the number of vertices in VERTEX-COVER, where each subset contains the edges from one vertex in VERTEX-COVER.

Thus, it is obvious that a vertex cover of size k in VERTEX-COVER is equivalent to a set cover of size k in SET-COVER where the subsets in the set cover correspond to the set of edges of the vertices in the vertex cover. In other words, the graph $G = (V, E)$ has a vertex cover of size k if and only if the set X has a set cover of size k . The proof starting from either side is obvious.