

1. A small (worst case) improvement in the factor of two approximation to weighted node cover

We now examine a modification of the node cover approximation method developed in homework 5. This modification improves slightly the worst case approximation bound, but might be much more effective in practice.

First observe that we don't need to double the occurrence of each edge of G . That is, for each edge (i, j) of G , put into B either the edge (i, j') or the edge (j, i') , but not both.

1a. Explain why, with this sparser graph B , a node cover of B still defines a node cover $S(G)$ of G such that $w(S(G))/w(S^*(G)) \leq 2$, where the total weight of a node cover X is denote $w(X)$.

Answer: A node cover of G still gives a node cover of B with the same set of nodes in B as in the original construction. That is clear because the edges in the new B are a subset of the edges of the original B , and hence a node cover of the original B is a node cover of the new B . And since the node cover of B is the same as before, it has weight at most twice that of the optimal node cover in G .

This seems intuitively better than the original construction because the number of edges has been reduced by half, so intuitively the optimal node cover of B should be smaller. However, every node of G still "appears" twice in B , and the optimal node cover in G is still doubled in B , in the above argument. This doubling can be reduced with the following rule.

First, pick any edge (i, j) in G and place i (but not i' in B). Then for every neighbor k of i in G (including j), place k' (but not k) in B , and put the edge (i, k') into B . Similarly, for every neighbor k (including i) of j in G , put k (but not k') in B , and put the edge (k, j') into B . Then neither i' nor j appears in B . After putting in these nodes and edges, add all nodes which are not neighbors of i or j , and for each edge (u, v) of G which does not yet appear in B , place either edge (u, v') or (u', v) into B .

Theorem Letting $S(G)$ be the node cover of G defined by the node cover $S^*(B)$, and assuming all nodes have positive weight, $w(S(G)) < 2w(S^*(G))$.

1b. Prove (that is, explain completely) the above Theorem.

Answer: First note that by this construction, for every edge (i, k') in B ,

edge (i, k) is in G , and for every edge (k, j') , edge (k, j) is in G , therefore a node cover S in G , which must contain either i or j or both nodes, can be converted to a node cover in B as follows: if node i is in S , then choose i for the node cover of B ; if j is in S , then choose j' for the node cover in B ; for any other node k in S , choose both k and k' for the node cover in B . The result is a node cover in B in which at least one node of S (either i or j) is *not* duplicated in the node cover of B . Therefore the optimal node cover of B has weight strictly less than twice the optimal node cover of G . Conversely, as before a node cover of B can be converted to a node cover of G at the same or smaller weight.

The best edge to pick (to get the best guaranteed approximation result) is the edge (i, j) in G where $\min[w(i), w(j)]$ is maximized. The heuristic can be improved by iterating, as long as every edge of G ends up in B at least once. It is an interesting open question how nodes and edges should be picked in order to optimize the effectiveness of this heuristic. If you have an idea, explore it.

Answer: I don't have a good idea for this one.

2. Suppose you have a set P of n points in three dimensional space. You want to pick a single point, call it p^* , which need not be in P , that minimizes the average Euclidian distance of the points in P to p^* . For any point p in the space, let $D(p)$ denote the average distance of the points in P to p . Give a polynomial-time algorithm to find a point p' such that $D(p')/D(p^*) < 2$.

Answer: For each point $s \in P$ compute the average distance between s and the points in P . Let p' be the point in P with smallest such average distance. Then adapt the Steiner string argument to cover this problem, noting that the key change is the use of Euclidian distance instead of edit distance, and that the key needed fact of edit distance is that it obeys triangle inequality, which of course Euclidian distance does also.

3. Consider the factor of two approximation method for unweighted node cover discussed in class on Nov. 14, based on a maximal matching. In class we asked if there is a small modification of the method that produces a node cover whose size is guaranteed to be strictly less than twice the size of the optimal. I believe the answer is that there is such a small modification. Explore this question and see if you also find a small modification that achieves this result. Of course, the term "small" is subjective and it is hard to specify in advance exactly what is small, but I think we all know it when we see it.

Answer: The node cover obtained by taking both ends of the edges in a maximal matching is called the heuristic node cover. Let $e = (u,v)$ be an edge in the maximal matching. Now any node cover must contain either u or v , and since no edges in a matching share any nodes, any node cover must have at least half the nodes that the heuristic cover has. That is, not only must every node cover have at least half as many nodes as does the heuristic cover have, every node cover must have at least half of the specific nodes that the heuristic node cover has. Therefore if the optimal node cover has any node outside of the heuristic cover, the heuristic node cover will have size less than twice that of the optimal cover. Hence, if the heuristic solution has exactly twice as many nodes as the optimal, it must be that optimal node cover is a strict subset of the heuristic node cover.

This leads to an approximation algorithm for node cover that is strictly within a factor of two of the optimal: For each node v in the heuristic node cover, determine if the heuristic node cover would still be a node cover when v is removed. If such a v is found, return the reduced node cover. If no such v is found, return the heuristic node cover.

Why is the returned node cover strictly smaller than twice the size of the optimal node cover? Clearly, if any such v is found, then the resulting node cover has size strictly less than twice that of the optimal, since it is smaller than the heuristic node cover, which was at most twice the size of the optimal. Conversely, as discussed above, if the heuristic node cover has size exactly twice that of the optimal, then there must be such a v , because the optimal cover must be a strict subset of the heuristic cover. Therefore, if no such v is found, it must be because the heuristic node cover is already strictly smaller than twice the size of the optimal node cover.

4a. In the approximation for the Steiner string problem, we said that the (unweighted) edit distance has the property that $D(\bar{S}, S_i) \leq D(\bar{S}, S^*) + D(S^*, S_i)$ for any S_i . Give a detailed explanation of that fact.

Answer: Case analysis. (this answer would get few points).

4b. Now suppose that we have a function f that maps any two strings S_i, S_j to a non-negative number $f(S_i, S_j)$. Suppose further, that f satisfies the triangle inequality. That is, for any three strings S_i, S_j, S $f(S_i, S_j) \leq f(S_i, S) + f(S, S_j)$, and $f(S_i, S_i) = 0$ for any S_i .

Given a set of strings \mathcal{S} , we define the f -consensus error of a string S relative to \mathcal{S} as $Ef(S) = \sum_{S_i \in \mathcal{S}} f(S, S_i)$. Note that S need not be from \mathcal{S} .

Define S^* as a string that minimizes the f -consensus error over all possible strings.

Problem: Is there a polynomial-time algorithm to find a string S such that $Ef(S)/Ef(S^*) < 2$? Briefly explain your answer.

Answer: The essential technical point in the original Steiner String analysis is that edit distance obeys triangle inequality. Everything goes through when f replaces edit distance, since f also obeys triangle inequality.

In a way, question 4b and question 2 were making the same point, but in different specific ways.