A 2.5 Factor Approximation Algorithm for the k-MST Problem

Sunil Arya* H. Ramesh†

Abstract

The k-MST problem requires finding that subset of k vertices of a given graph whose Minimum Spanning Tree has least weight amongst all subsets of k vertices. There has been much work on this problem recently, culminating in an approximation algorithm by Garg [G], which finds a subset of k vertices whose MST has weight at most 3 times the optimal. Garg also argued that a factor of 3 cannot be improved unless lower bounds different from his are used. We use a pruning technique on top of Garg's algorithm to achieve an approximation factor of 2.5. Note that Garg's algorithm is based upon the Goemans-Williamson [GW] clustering method, using which it seems hard to obtain any approximation factor better than 2.

1 Introduction

The k-MST problem has received much attention in recent years. The first constant factor approximation algorithm for this problem on general graphs with non-negative edge weights was given by Blum, Chalasani and Vempala [BCV]. The constant in the approximation factor was around 17. Subsequently, Garg [G] explored and exploited structural properties in their algorithm and gave a 3-factor approximation algorithm. Garg also showed that the factor of 3 was impossible to beat using only his lower bounds. We use a pruning technique on top of Garg's algorithm to give a 2.5-factor approximation. The Goemans-Williamson clustering approach [GW] on which all the above algorithms are based seems to have a factor of 2 inherent in it. It remains open whether one can actually obtain a 2-factor approximation algorithm for this problem.

2 Outline of Garg's Algorithm

Given a (multi)graph G with non-negative edge weights, a specified root vertex r, and a number k, Garg's algorithm determines two disjoint sets of vertices V_1 and V_2 with the following properties:

*Hong Kong University of Science and Technology.

†Indian Institute of Science, Bangalore. Work done partly at Hong Kong University of Science and Technology.

1. $|V_1| + |V_2| = k$.

2. V_1 contains r.

3. The sum of the weights of the MSTs of V_1 and V_2 is at most $2L$, where L is the weight of the optimal k-MST containing r.

Garg's approximation factor of 3 results as follows. The two sets V_1 and V_2 are connected together using the shortest path from the root r to any vertex in V_2. Garg ensures that this shortest path has length at most L[1] by removing in advance all vertices whose distance[2] to r is more than L; since the edge weights are non-negative, none of the vertices removed can be in the optimal k-MST containing r. This results in a tree rooted at r with at least k vertices and having weight at most $3L$. Again, since edge weights are non-negative, this tree can be easily pruned to yield a rooted tree with exactly k vertices and having weight at most $3L$. Repeating this with each vertex in G as root gives a 3-factor approximation to the k-MST problem.

3 Our Pruning Procedure

We consider each edge e of graph G in turn and perform the following procedure. We obtain a multigraph G' from G by contracting the edge e so that both endpoints of this edge are now in a single vertex $r \in G'$. Note that there is a one-to-one map from edges in $G - e$ to edges in G'. We then prune away all vertices in G' whose distance from r exceeds $L/2$ to get a multigraph G''. Here, L[3] is the weight of the $k-1$-MST in G' containing vertex r. Next, we use Garg's algorithm on G'' with root r and value $k - 1$ to obtain sets V_1 and V_2 with the above properties. We then take the MSTs of these two sets and connect them using the shortest path from r to any vertex in V_2. Finally, we add the edge e to this tree to get the result tree. The least weight tree so obtained over all edges e is the output of our algorithm.

Lemma 3.1 *The above algorithm gives a 2.5 factor approximation for the k-MST problem.*

Proof. Consider the optimal k-MST T in G, and let L be its weight. Let v, w be two leaves such that the length of the path from v to w in T is the maximum over all pairs of leaves. This length is at most L. Consider the *central edge* e on the path from v to w in T, i.e., an edge such that v and w are both distance at most $L/2$ from one of its endpoints. Note that such an edge always exists.

Now consider our algorithm with this edge e contracted. Let T' be the tree T with the edge e contracted. Clearly, all the vertices in T' lie within a distance of $L/2$ from the root r in G', representing the endpoints of e. Thus every vertex in T' is also in G''. Garg's

[1] Even though L is not known in advance, it can be approximated using binary search to an inverse exponential accuracy.

[2] Distance between two vertices is the weight of the shortest path between them.

[3] As before, we can assume that L is known to an inverse exponential accuracy.

algorithm on G'' with value $k-1$ will yield two disjoint subsets of total size $k-1$, whose MSTs have weights summing to at most $2(L - w(e))$, where $w(e)$ is the weight of edge e. Connecting them will require a path of length at most $L/2$. The total weight of the tree so obtained is at most $2.5L - w(e)$. □

References

[BCV] A. Blum, P. Chalasani, S. Vempala. A constant factor approximation algorithm for the k-MST problem. Proceedings of *STOC 96*.

[G] N. Garg. A 3 factor approximation algorithm for the k-MST problem. Proceedings of *FOCS 96*.

[GW] M. Goemans, D. Williamson. An approach for solving network design problems. Proceedings of *SODA 92*.