Minimum Cost Topology Construction for Rural Wireless Mesh Networks

Debmalya Panigrahi[†*] Partha Dutta[‡*] Sharad Jaiswal [§] K. V. M. Naidu [§] Rajeev Rastogi [§]

[†]MIT
debmalya@mit.edu

[‡]IBM India Research Lab
parthdut@in.ibm.com

[§]Bell Labs Research India
{jsharad,naidukvm,rastogi}@alcatel-lucent.com

***Abstract*—IEEE 802.11 WiFi equipment based wireless mesh networks have recently been proposed as an inexpensive approach to connect far-flung rural areas. Such networks are built using high-gain directional antennas that can establish long-distance wireless point-to-point links. Some nodes in the network (called gateway nodes) are directly connected to the wired internet, and the remaining nodes connect to the gateway(s) using one or more hops.**

The dominant cost of constructing such a mesh network is the cost of constructing antenna towers at nodes. The cost of a tower depends on its height, which in turn depends on the length of its links and the physical obstructions along those links. We investigate the problem of selecting which links should be established such that all nodes are connected, while the cost of constructing the antenna towers required to establish the selected links is minimized. We show that this problem is NP-hard and that a better than $O(\log n)$ approximation cannot be expected, where n is the number of vertices in the graph. We then present the first algorithm in the literature, for this problem, with provable performance bounds. More precisely, we present a greedy algorithm that is an $O(\log n)$ approximation algorithm for this problem. Finally, through simulations, we compare our approximation algorithm with both the optimal solution, and a naive heuristic.

I. Introduction

Rural areas (especially in developing regions) have populations with very low paying capacities. Hence, a major factor in network deployment is the cost of the infrastructure and the network equipment. In this context, we investigate efficient algorithms for the minimum cost topology construction problem in rural wireless mesh networks.

The cost of laying wire to rural areas is prohibitively expensive. Also, traditional wireless technologies such as cellular data networks (e.g., EV-DO) and upcoming technologies like IEEE 802.16 WiMAX have prohibitively expensive equipment costs. As a result, there has been considerable recent interest [2], [8], [16], [17] in the design of rural mesh networks using IEEE 802.11 (WiFi) equipment. The cost of an 802.11 radio (∼\$50/PCMCIA card) is orders of magnitude less than that of cellular/WiMAX base stations. Thus, this approach is an attractive option for building low cost networks.

Rural mesh networks have two key characteristics, 1) a fixed topology (a node in this network is a village), and 2) long-distance links between the nodes (about 7-8 kms). As depicted in Fig. 1, a typical IEEE 802.11 based rural mesh network

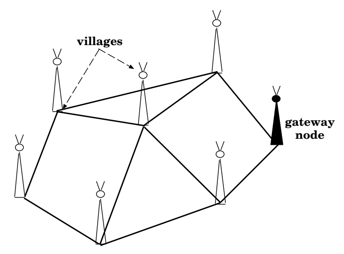

Fig. 1. A rural wireless mesh network.

consists of a cluster of villages connected with each other through point-to-point wireless links. Some special nodes in this mesh, called the *gateway* nodes, are connected to the wired internet. Other mesh nodes connect to the gateway nodes (and thus, to the rest of the internet) through one or more hops in the mesh.

An essential requirement to establish long-distance links is that line-of-sight is maintained between the radio antennas at the end-points. To ensure line-of-sight across such long distances (over obstacles such as trees, buildings and the terrain), would require the antennas to be mounted on tall towers. The required height of the towers depends both on the length of the link, and the height of the obstructions along the link. The cost of the tower depends on its height and the type of material used. For relatively short heights (10-20 meters) antenna masts are sufficient. For greater heights, sturdier and much more expensive antenna towers are required. Fig. 2 (taken from [4]) lists the cost of building towers and masts for various heights.

To cover a distance of 7-8 kms requires the tower height of at least one end-point to be around 30-45 meters [3]. The cost of building such a tower (\$4000 - \$5000) is orders of magnitude greater than the cost of the communication equipment at a node. Given this considerable difference between the cost of towers and other equipment, the principal problem in building rural mesh networks is to construct a topology with the lowest total cost of antenna towers.

A basic requirement of any topology is that it should connect all villages to the gateway node, i.e., we would like to construct a minimum cost spanning subgraph. The cost of the subgraph is the sum of the cost of antenna towers required to establish all the links in the subgraph. In this work, we describe the first algorithms for this topology construction problem with provable performance bounds.

To state the problem more formally, consider a graph

*This work was done when Debmalya Panigrahi and Partha Dutta were at Bell Labs Research India.

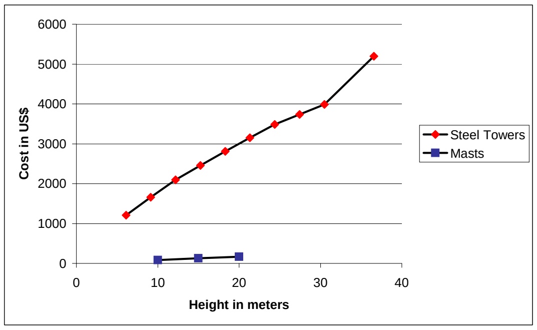

Fig. 2. Cost of steel towers and masts for varying heights

$G = (V, E)$, where V is the set of vertices and E is the set of edges. Let $n = |V|$. A *height (assignment) function* h gives an assignment of tower height to every node of G. (We only consider integer height assignments.) We say that an edge (u, v) is covered by a height function if tower heights $h(u)$ at u and $h(v)$ at v are sufficient to establish (or cover) the edge between u and v. (In Section III, we will describe the requirements on the tower heights to establish a link.) Let $COVER(h)$ denote the set of edges that are covered by the height function h.

Given a tower height, a *cost function*, denoted by c, gives the cost of building a tower of that height. The *total cost* of a height function h is $\sum_{v \in V} c(h(v))$. Among all height functions h such that $COVER(h)$ is a connected spanning subgraph, our goal is to find the height function with the minimum total cost.

This is computationally a very expensive problem. It involves searching over all possible connected spanning subgraphs, and over all possible heights of the towers on each node. A naive (brute force search) approach would have a complexity of $O(h_{max}^n)$, where h_{max} is the maximum possible height of a tower. Clearly, for any real network (with around 50 nodes) computing the optimal solution is not possible - even if it has to be done only once. We have also modeled the problem as an ILP and, as expected, found that the LP solver could return an optimal solution only for small-sized graphs (at most 11 nodes).

In this paper, we make several important contributions towards developing efficient algorithms to solve this problem. First, we describe the requirements to establish a point-to-point 802.11 link between two nodes of a given network graph. We then give the formal definition of the Topology Construction problem (denote by TC). We prove the problem to be NP-hard by a reduction from the set-cover problem. We also show that the reduction is gap-preserving, and hence, one cannot realistically expect a better than $O(\log n)$ approximation. We then describe a greedy algorithm that achieves this $O(\log n)$ lower bound up to a constant factor. Finally, through extensive simulations, we compare our approximation algorithm with both the optimal solution, and a naive heuristic. In our simulations, our algorithm provides improvements of upto 225% over the naive approach.

Organization. The paper is organized as follows. In section II, we discuss related work on topology construction in wireless networks. In section III, we describe how to model antenna heights and costs and give a formal description of the Topology Construction problem. (We also show in the appendix that the TC problem is NP-Hard and has a logarithmic hardness of approximation.) We then present our approximation algorithm for solving this problem in section IV. In section V, we present results from numerical simulations that compare our approximation algorithm with the optimal solution and a naive heuristic. Finally, we conclude with some directions for future work in section VI.

II. Related Work

The Digital Gangetic Plains (DGP) project [6] in and around Kanpur, India, is an operational example of the type of rural mesh network that motivates this work. Nodes communicate in DGP using long-distance point-to-point links that are established using directional antennas.

Cellular networks require efficient schemes to place towers to cover large areas while minimizing costs. However the problem in cellular network deployment is to place the minimum number of towers to cover the maximum possible area. In our problem the location of the towers is fixed (villages), and the goal is instead to select a set of links and tower height assignments that will cover the links with minimum costs.

We now discuss some related work around topology construction in a setting similar to ours - wireless mesh networks.

Topology Construction Problem. There has been some recent work on constructing low-cost wireless mesh networks [19], [8]. The authors in [19] discuss solutions to the topology construction problem in the same context as ours—rural mesh networks where the costs are dominated by the height of antenna towers. They describe a two-pass heuristic for this problem. In the first pass, a spanning tree is determined based on criteria such as a bound on the maximum degree, depth of the tree etc. In the second pass, the optimal height assignment to cover all the edges in the selected spanning tree is determined by solving a linear program.

We improve on [19] in three ways. Firstly, no performance bounds are presented for the heuristic in [19] while our algorithm gives a worst-case logarithmic bound. Secondly, the heuristic considers only a single obstruction between two nodes, while our approach can handle any number of obstructions. Finally, while the heuristic is restricted to using a piecewise linear cost function for the antenna towers, our algorithms can handle a much more general cost function (satisfying some simple properties discussed in the next section).

The authors in [8] also discuss a geometric version of our problem but do not provide any algorithm for solving it. So, to the best of our knowledge, we present the first algorithms for solving the topology construction problem with provable guarantees on the approximation factor.

Power Optimization in Wireless Multi-Hop Networks. A related problem addressed in the literature is that of power optimal topology construction in wireless multi-hop networks [5], [7], [9], [10], [12], [14], [15]. In this class of problems, the transmit power on a node is determined by the longest link incident on the node. In general, the aim is to construct topologies which minimize the maximum power consumption of radio transmitters on network nodes, while

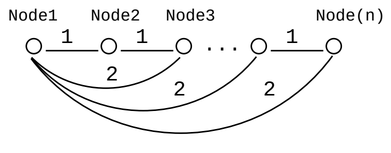

Fig. 3. An example of MST being $\Omega(n)$-factor worse than the optimal

ensuring a desired level of connectivity/fault-tolerance. Similar to our problem, the cost incurred at the node can be amortized over all the links incident on it.

This problem is also NP-hard, but admits constant factor approximation algorithms for selecting k-connected networks, for small values of k. For $k = 1$, [12] and [7] give approximation algorithms with an approximation factor of 2 and 1.69 respectively. For arbitrary k, the authors in [9] give $O(k)$-approximation algorithms for both vertex and edge connectivity. These approximation factors were improved to $O(\log^4 n)$ and $O(\sqrt{n})$ for the vertex and edge connected cases respectively in [10]. The main observation on which these algorithms are based is that the minimum-weight spanning tree (MST), or an approximation to a minimum-weight k-connected subgraph (when $k > 1$), gives a good approximation to the power-optimal solution. Unfortunately, fixing an MST as the underlying topology does not work for our problem, as we show in the following example.

Consider the network graph G in Fig. 3, and assume that the cost function is the identity function. Suppose that in the middle of every edge e in G there is an obstruction of height $h(e)$. Then to cover the edge e, the tower heights at its endpoints should be sufficient to clear the obstruction in the middle. As we will explain in the next section, this requires that the sum of the tower heights at the two endpoints of e should be at least $2h(e)$. Thus, a natural candidate for the weight $w(e)$ is $2h(e)$. Suppose these edge weights for G are the ones depicted in Fig. 3.

Now with the above edge weights, the path formed by all the edges with weight 1 is an MST of G. The minimum cost of constructing towers to cover all the edges of this MST is roughly $n/2$, e.g., by constructing a tower of height 1 on every other node on this path. However, it is easy to see that a tower of height 2 at node 1 is sufficient to cover another spanning tree—the star graph centered at node 1—which is not an MST. Thus in this case, selecting an MST as the set of edges to be covered leads to a cost that is $\Omega(n)$ times worse than the optimal cost. In contrast, in the worst case our algorithm is only $O(\log n)$ worse than the optimal.

Another variant of the power-optimal network construction problem seeks to minimize power consumption while ensuring the desired *directed* connectivity in the network. The 1-connected version of this problem has a logarithmic hardness [15] (while, as we pointed out earlier, the undirected version admits a constant factor approximation algorithm). However, the techniques in [15] are not immediately applicable to our problem for the following two reasons. Firstly, in [15] the input to the problem specifies a weight for each edge. In our problem we may be given multiple obstructions for each edge, which cannot be abstracted by a single fixed edge weight (unlike the simple MST example above, which has just 1 obstruction per edge). Secondly, in [15] a directed

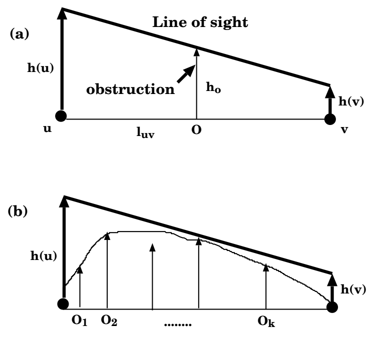

Fig. 4. Computing the height of towers at the end-points of a link

edge is covered only if the power at the *source node of the edge* is more than the weight of the edge. However in our problem, whether an undirected edge is covered depends on the obstacles on that edge, and the endpoints of the edge can "cooperatively" set their tower heights to cover the edge. For instance, as in the MST example above, if there is an obstruction of height $h(e)$ in the middle of an edge e, then to cover the edge we need that the *sum* of the heights at the two endpoints is at least $2h(e)$.

Greedy Algorithms. Our topology construction algorithm is a greedy algorithm. Greedy algorithms have been developed for some network construction problems—most notably by [13] for the node-weighted Steiner tree problem. Their algorithms proceed by greedily adding subgraph structures called *spiders* to the set of selected edges. However, while node weights are fixed in their case, we need to also find the correct tower heights (which correspond to weights) at the nodes.

III. Network Model and Problem Statement

In this section we will describe the requirements on the height of the towers at the end-points to establish a link between any two nodes in the network graph. We also discuss the cost function that we use in this paper. We then give a formal problem statement for this work and show that the problem is NP-Hard.

A. Computing tower heights at the end-points of a link

Consider two nodes, u and v that are separated by a distance l_{uv}. The edge (u, v) is considered to be *covered* if an 802.11 based point-to-point communication link can be established between u and v. Assume that the transmit powers and the gains of the antennas at both ends are sufficient to overcome the free-space path loss between the two points. The first basic requirement to cover the edge between u and v is that there be a clear *visual line-of-sight* between the antennas at the end-points (as shown in Fig. 4a). In other words, the line joining the antennas mounted on the towers should clear any obstructions along the path. Secondly, it is also required that *RF line-of-sight* is maintained between the two points. This is determined by an elliptical area between u and v termed the first *Fresnel zone*. To establish RF line-of-sight, a significant area of the Fresnel zone ($> 60\%$ of the radius of the Fresnel zone at the location of the obstruction [1]) should also clear all obstructions between u and v. However, this can be simply modeled by extending the height of the obstruction to include the radius of the Fresnel zone that has to be in the clear.[1]

In reality, there can be multiple obstructions between u and v. As in Figure 4b, consider multiple obstructions, $O_1, O_2, \ldots, O_k$ between u and v. Now, let $h(u)$ and $h(v)$ represent the tower heights at the nodes of u and v. Covering edge (u, v) requires a visual and RF line-of-sight connection between the towers at its two terminal nodes. This would imply that the straight line f_{uv} joining the top of the two towers (of heights $h(u)$ at u and $h(v)$ at v) should clear every obstruction in (u, v).

Hence, we also note that given a particular pair of tower heights at u and v, deciding whether these heights covers edge (u, v) can be done in time linear in the number of obstructions on that edge.[2]

B. Modeling tower costs

An important component in this problem is the nature of the cost function that maps tower heights to the cost of building the tower. As shown in Fig. 2, in our setting, there are two types of antenna towers that are used. For heights less than 20 meters, one can use the cheaper masts. For greater heights, one has to use the more expensive steel towers. Further, there is an order of magnitude difference between the cost of the cheaper masts and that of the steel towers. Thus, roughly speaking, the cost function is constant as long as the cheaper masts can be used and becomes linear in height once the steel towers are needed, with a jump in cost when we switch from masts to steel towers. Let us denote the height at which the material of the tower has to be switched as h_{min}. Further, there is a physical restriction on the maximum possible height of a tower, denoted by h_{max}. Thus, the cost function c can be formally defined as

$$c(h) = \begin{cases} K & \text{if } 0 \le h \le h_{min} \\ Ah + B & \text{if } h_{min} < h \le h_{max} \end{cases}$$

where A, B and K are constants and $Ah_{min} + B >> K$.

Although, in practice, the cost function can be modeled as discussed above, our algorithm works with a much more general cost functions. Specifically, we only require the cost function c to satisfy the following two natural properties **C1** and **C2**.

C1 Given the tower costs at two neighboring nodes u and v, it can be determined (in polynomial time) whether the corresponding tower heights cover the edge (u, v). This simply requires that the corresponding tower height can be computed (in polynomial time) given the tower cost.[3]
As mentioned earlier, determining whether the height of the towers is sufficient to cover an edge can be done in polynomial time.

C2 The cost function is monotonically increasing with height, i.e., $h_1 \ge h_2 \Rightarrow c(h_1) \ge c(h_2)$ for any values of h_1 and h_2.

It is easy to verify that the cost function c defined earlier in this section satisfies both of the above properties. In the remainder of this paper, when the height function h is unambiguous, we will often denote the cost of the tower at a node v as $c(v)$ rather than $c(h(v))$.

C. Problem statement and Hardness

Before we formally define the Topology Construction (TC) problem, we need a few more definitions. A height function h is said to be *valid* if $h(v) \le h_{max}$, for each vertex v. Now, for any height function h, let $COVER(h)$ be the set of edges that are covered by h.

Input. An undirected graph $G = (V, E)$, with obstruction locations and heights on each edge $e \in E$ and a cost function $c : \mathbb{R}^+ \to \mathbb{R}^+$ which satisfies the properties **C1** and **C2** mentioned above.

Output. A valid height function h such that the subgraph induced by edges in $COVER(h)$ is a connected, spanning subgraph[4] of G such that the total cost of the towers $\sum_{v \in V} c(v)$ is minimized.

We prove the following theorem in the appendix that shows that the TC problem is NP-hard, and coupled with results in [18], it also implies that it is NP-hard to approximate the TC problem to a factor which is asymptotically better than $O(\log n)$.

Theorem 1: There is a gap-preserving reduction from the minimum set cover problem to the TC problem.

IV. Approximation Algorithm

We give a greedy algorithm TC-ALGO for the TC problem and show that it achieves an approximation ratio of $4(1 + \log n)$. Since achieving an approximation ratio that is better than $O(\log n)$ is NP-hard, our algorithm is optimal (subject to constant factors). In the next two subsections, we first describe the algorithm, and then give an analysis for bounding the approximation factor.

We need a few more definitions before describing our algorithm. Given a set of edges $E' \subseteq E$, we define the number of components of E' as the number of components in the graph (V, E').[5] For any height function h, $COMP(h)$ denotes the number of components of $COVER(h)$.

A. Description

TC-ALGO is presented in Algorithm 1. It uses a subroutine STAR-TC-ALGO which is presented in Algorithm 2. In TC-ALGO, the height function is initialized to 0 at all nodes. Thus at the beginning, $COVER(h) = \emptyset$ and $COMP(h) = n$. Our goal is to obtain a least cost valid height function h such that

[1]Consider an obstruction O of height h_o at a distance d_1 from u and d_2 from v. The radius of the first Fresnel zone at this point is defined as $r_f = \sqrt{\frac{\lambda d_1 d_2}{(d_1+d_2)}}$, where λ is the wavelength of an 802.11b signal. To establish RF line-of-sight, it is required that at least 60% of r_f be clear of any obstacles at O [1]. In other words, the line joining the antennas at u and v should be at a height $> h_o + 0.6r_f$ at point O.

[2]In practice, links are likely to have a large number of obstructions, which would make this task time-consuming. To overcome this problem, one would only consider those obstructions which are above some threshold height with the assumption that clearing these obstructions would ensure that all other obstructions are also cleared.

[3]If the tower cost remains constant over a range of heights, then we use the maximum height in that range.

[4]$S = (V_S, E_S)$ is a *spanning subgraph* of $G = (V, E)$ if (1) $V_S = V$ and (2) $E_S \subseteq E$. A graph is said to be *connected* if any pair of nodes have a path connecting them.

[5]A component of a graph is a maximal connected subgraph.

$COMP(h) = 1$, i.e., h covers a connected spanning subgraph of G.

TC-ALGO proceeds in phases. In each phase, we select a height increment $incr_{best}$ for all nodes[6] which is then added to the current height function. This is repeated until we obtain a height function h which covers a connected spanning subgraph (i.e., $COMP(h) = 1$).

We now describe how $incr_{best}$ is selected in a phase. Every height increment has a cost associated with it, namely, the increase in the total cost of towers because of applying the increment. Roughly speaking, in a phase, we would like to select a height increment that requires minimum increase in cost for the maximum reduction in the number of components. We call this metric, the cost-to-benefit ratio. However, finding the best such height increment over the whole graph is a difficult problem. We therefore impose the following two constraints:

1. *Increments are local.* Any height increment $incr$ done in a phase is such that, there is a node v such that $incr$ only increases heights at v and its neighbors. Node v is called the *central node* of $incr$. This restriction on height increment helps us to localize the search for the best height increment.

2. *Only edges incident on the central node add to benefit.* Let h be the height function at the beginning of a phase. In any height increment $incr$, say with central node v, some new edges are covered, say set E_v. Let E'_v be the subset of E_v that contains only the edges that are incident on v.[7] Then the *benefit of* $incr$ in this phase is the difference between the number of components of $COVER(h)$ and $COVER(h) \bigcup E'_v$. This definition of benefit helps us to simplify the analysis of the approximation guarantee. (In the rest of this section, all height increments are local and benefit is always calculated as defined above.)

Now in each phase, we want to select the height increment $incr_{best}$ with the best cost-to-benefit ratio. To this end, in TC-ALGO we search over all nodes (line 6) to find a possible central node for the best height increment. Furthermore, for each candidate v for the central node, we do a doubling search over possible *cost increments* corresponding to the height increments at v. The values of resulting heights at v that are considered are the following: (1) the current height at v, $h(v)$, for which the cost increment is 0, (2) the maximum possible height, h_{max}, for which the cost increment is maximum, and (3) heights that are less than h_{max} and correspond to cost increments $\{1, 2, 4, 8, \ldots\}$. (This set of heights are denoted by $\mathcal{H}(v)$ in line 7.)

But, even though we search over every candidate central node v and its possible height increments, we still need to search over the possible height increments at neighbors of v. Thus, we are left with a subproblem, that we call STAR-TC: given the current height function h, a central node v and its height increment δ, find a height increment at neighbors of v that has the lowest cost-to-benefit ratio.

Algorithm 2 gives our algorithm STAR-TC-ALGO for the STAR-TC problem. We first note that increasing the height at nodes that are in the same component as v does not reduce the number of components. Thus, we restrict the height increments only to those neighbors of v that are in a component different from v in $COVER(h)$. We call this set of nodes nbr. Now for every node u in nbr we find the smallest height increment $h^+(u)$ at u such that the edge (u, v) is covered. (This can be done using a binary search over the possible height increments at u, and using property C1 from Section III-B.) Note that as the cost function is monotonic, height increment $h^+(u)$ also gives the lowest tower cost increment $c^+(u)$ at u to cover the edge (u, v). Now, we list the nodes in nbr in increasing order of their c^+ values. We call this list L. Next we observe that, once an edge from v to a node in some component is covered, covering an edge from v to another node in the same component does not reduce the number of components. Thus, for nodes in the same component, we keep the lowest incremental cost (c^+) node in L, and remove all other nodes from L.

We now consider the following $|L|$ height increments. Each increment increases heights only at v and the nodes that form a prefix of L. More precisely, the i^{th} height increment increases the height of v by δ, heights of the first i nodes in L by their respective $h^+()$, and all other nodes by 0. Note that the benefit of such an increment is exactly k. Then, among all these height increments, we return the one that has the lowest cost-to-benefit ratio.

Algorithm 1 TC-ALGO(G, c)

```
1: Input: graph $G = (V, E)$, cost function $c$
2: Output: height function $h$
3: for each $v \in V$ do $h(v) := 0$;
4: while $COMP(h) > 1$ do
5:     $r_{best} := \infty$;
6:     for each $v \in V$ do
7:         $\mathcal{H}(v) := \{h(v), h_{max}\} \cup$
           $\{c^{-1}(c(h(v)) + 2^i) : c(h(v)) + 2^i < c(h_{max})$ and $i = 0, 1, 2, 3, \ldots\}$
8:         for $\alpha \in \mathcal{H}(v)$ do
9:             $(r_{tmp}, incr_{tmp}) :=$
                   STAR-TC-ALGO$(G, h, v, \alpha - h(v))$;
10:            if $r_{tmp} < r_{best}$ then
11:                $r_{best} := r_{tmp}$; $incr_{best} := incr_{tmp}$;
12:    for each $v \in V$ do $h(v) := h(v) + incr_{best}(v)$;
13: return $h$;
```

B. Analysis

In this section we will prove that TC-ALGO has an approximation factor of $4(1 + \log n)$. We start with the analysis of STAR-TC-ALGO which we show to be an optimal algorithm for the STAR-TC problem. Then we show that a phase of TC-ALGO chooses a height increment such that the corresponding cost increment is within twice of the best height increment at that phase. Finally, we use these two results to show the approximation factor of the entire TC-ALGO.

1) STAR-TC-ALGO: We now show that STAR-TC-ALGO finds an optimal solution for the STAR-TC problem.

[6]Depending on the context, a height increment refers to the increase in tower height at a node, or at a set of nodes.

[7]E_v may contain an edge that is not incident on v but is incident on a neighbor u of v—this edge gets covered when the height at u is increased.

Algorithm 2 STAR-TC-ALGO(G, h, v, δ)

```
1: Input: graph $G = (V, E)$, height function $h$, node $v$, height increment $\delta$ at $v$
2: Output: cost-to-benefit ratio $r'_{best}$, height increment function $incr$
3: $c^+(v) := c(h(v) + \delta) - c(h(v))$;
4: $nbr :=$ set of neighbors of $v$ that are not in the same component as $v$ in $COVER(h)$;
5: for each node $u \in nbr$ do
6:     $h^+(u) :=$ smallest $\beta$ s.t. heights $(h(v)+\delta)$ at $v$ and $(h(u)+\beta)$ at $u$ cover edge $(u, v)$;
7:     $c^+(u) := c(h(u) + h^+(u)) - c(h(u))$;
8: $L :=$ list of nodes in $nbr$ in ascending order of $c^+$;
9: for each component $D$ in $COVER(h)$ do
10:    remove from $L$ all nodes $u \in D$ except the one with lowest $c^+$;
11: $r'_{best} := \infty; k_{best} := 0$;
12: for $1 \le k \le |L|$ do
13:    $r'_{tmp} := (c^+(v) + \sum_{1 \le i \le k} c^+(L[i]))/k$;
14:    if $r'_{tmp} < r'_{best}$ then
15:        $k_{best} := k; r'_{best} := r'_{tmp}$;
16: for each $u \in V$ do $incr(u) := 0$;
17: for each $u \in L[1 \ldots k_{best}]$ do $incr(u) := h^+(u)$;
18: $incr(v) := \delta$;
19: return $(r'_{best}, incr)$;
```

Lemma 2: STAR-TC-ALGO returns an optimal solution to the STAR-TC problem.

Proof: Consider the optimal height increment $incr_{opt}$ for an instance of the STAR-TC problem (G, h, v, δ). We will do two transformations to $incr_{opt}$ to obtain a height increment $incr_{algo}$ that is considered by STAR-TC-ALGO. During both transformations, the benefit remains the same and the cost increment either remains the same or decreases.

Since all allowed height increments are local, $incr_{opt}$ increases the height of v and some of its neighbors. We modify $incr_{opt}$ to obtain another height increment $incr'_{opt}$, as follows (all components are components of $COVER(h)$):

- For every node (except v) in the component containing v, change the height increment to 0. Node v retains its height increment δ.
- Consider each component D which becomes connected to the component containing v, due to the height increment $incr_{opt}$. Among nodes in D, select a node u such that (u, v) can be covered with the lowest increase in cost at u. Accordingly, set the height increment at u. For all other nodes in D set the height increment to 0. Let J be the set of selected nodes.
- Consider each component D which does not become connected to the component connecting v, due to the height increment $incr_{opt}$. Set the height increment at all nodes in D to 0.

Note that $incr_{opt}$ and $incr'_{opt}$ both have the same benefit, $|J|$. It is also easy to see that the cost of $incr'_{opt}$ is at most that of $incr_{opt}$. Therefore, as $incr_{opt}$ is optimal, $incr'_{opt}$ is also optimal.

It follows from STAR-TC-ALGO that J is a subset of the list L at the beginning of the **for** loop in line 12.[8] Also, for any node $u \in J$, $incr'_{opt}(u)$ is equal to the $h^+(u)$ selected in the algorithm. Now consider a height increment $incr_{algo}$ where the first $|J|$ elements of L have height increment given by h^+, v has increment δ, and all other nodes have increment 0. Since L is arranged in ascending order of additional cost, and J is a subset of L, cost of $incr'_{opt}$ is greater than or equal to the cost of $incr_{algo}$. Clearly, the benefit of $incr_{algo}$ is $|J|$. Thus $incr_{algo}$ also has the optimal cost-to-benefit ratio. As $incr_{algo}$ is one of the height increments considered by the algorithm while selecting the prefix of L with the lowest cost-to-benefit ratio (line 13), the height increment returned by the algorithm is optimal. ■

2) A phase of TC-ALGO*:* The lemma below shows that given a height function, a single phase of TC-ALGO finds a 2-approximation of the optimal height increment. Roughly speaking, the 2-approximation in a phase results from the doubling search over the cost of height increments at a node.

Lemma 3: Let h be the height function at the beginning of a phase in TC-ALGO. Then the cost-to-benefit ratio of the height increment chosen in that phase of TC-ALGO is at most twice the cost-to-benefit ratio of any height increment from h.

Proof: Let h^+_{opt} be a height increment that has the lowest cost-to-benefit ratio starting from h. Suppose h^+_{opt} is centered at node v, and let c^+_{opt} be the corresponding cost increment.

Consider the set of all cost increments corresponding to heights of v in $\mathcal{H}(v)$, namely $\{0, c(h_{max}) - c(h(v))\} \cup \{1, 2, 4, \ldots\}$. Since $0 \le c^+_{opt}(v) \le c(h_{max}) - c(h(v))$, there is one cost increment γ in this cost increment set such that $c^+_{opt}(v) \le \gamma \le 2c^+_{opt}(v)$.

Now consider a height increment h^+_{tmp} such that the height increment at all nodes $u \neq v$ is given by h^+_{opt}, and at node v is $c^{-1}(c(h(v)) + \gamma) - h(v)$. Clearly, as $\gamma \le 2c^+_{opt}(v)$, total cost increment due to h^+_{tmp} is at most twice that of h^+_{opt}. Notice that, $c(h(v)) + \gamma \ge c(h(v)) + c^+_{opt}(v) = c(h(v) + h^+_{opt}(v))$. Therefore, as the cost function is monotonically increasing, $h^+_{tmp}(v) \ge h^+_{opt}(v)$. Thus at all nodes $h^+_{tmp} \ge h^+_{opt}$, and hence, the benefit of h^+_{tmp} is greater than or equal to h^+_{opt}. It follows that the cost-to-benefit ratio of h^+_{tmp} is at most twice that of the optimal.

By construction, the height increment at v due to h^+_{tmp} (the height increment corresponding to the cost increment γ) is one of the height increments at v considered in TC-ALGO. Since STAR-TC-ALGO is an optimal algorithm for the STAR-TC problem, the cost-to-benefit ratio of the height increment selected by TC-ALGO in this phase is less than or equal to that of h^+_{tmp}, which is within twice of the optimal. ■

3) TC-ALGO*:* Suppose the input graph $G = (V, E)$ for the TC problem has n nodes and m edges. Let OPT be the total cost of the optimal height function. We now show that TC-ALGO has an approximation factor of $4(1 + \log n)$.

[8]Here we are assuming that any tie between two nodes in the same component, for the node with the lowest cost increment in that component, is resolved in the same way in STAR-TC-ALGO and $incr'_{opt}$, e.g., by choosing the node with the smallest identifier.

Lemma 4: Consider an intermediate phase in an execution of TC-ALGO. Suppose the height function at the beginning of this phase is h_β, and let x be $COMP(h_\beta) - 1$. Then, the height increment selected by this phase has a cost-to-benefit ratio of at most $4OPT/x$.

Proof: Let us denote the optimal height function as h_{opt}. Then, $COVER(h_{opt})$ is the set of edges in E which are covered by the optimal height function and $COMP(h_{opt}) = 1$. Now, let $G_\beta = (V, COVER(h_\beta))$ be the graph formed by the edges covered by h_β. Since $COMP(h_\beta) = x+1$, G_β has $x+1$ components, which we denote by $D_\beta = \{D_\beta^1, D_\beta^2, \ldots, D_\beta^{x+1}\}$. Now, let E_β^{opt} be the set of edges in $COVER(h_{opt})$ which connect different components in D_β. Formally, $E_\beta^{opt} = \{(u,v) \in COVER(h_{opt}) \mid u \in D_\beta^i, v \in D_\beta^j, i \neq j\}$. Since $COVER(h_{opt})$ has only a single component, E_β^{opt} connects all components in D_β. Let T_β^{opt} be an *acyclic* subset of E_β^{opt} which connects all the components in D_β. Clearly, $COVER(h_\beta) \cup T_\beta^{opt}$ connects all the vertices in V. ($COVER(h_\beta)$ connects vertices internally in each component of D_β and T_β^{opt} connects all components of D_β.) For an example, refer to Figure 5(a).

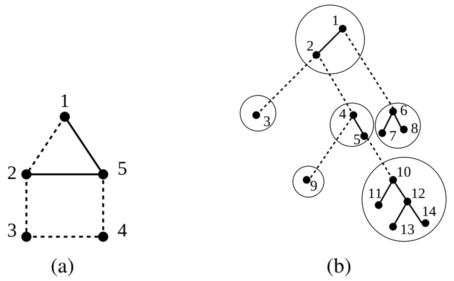

Fig. 5. (a) In this example, dotted edges are in $COVER(h_{opt})$ and solid edges in $COVER(h_\beta)$, $D_\beta^1 = \{1,2,5\}$, $D_\beta^2 = \{3\}$, $D_\beta^3 = \{4\}$, $E_\beta^{opt} = \{(2,3),(3,4),(4,5)\}$ and T_β^{opt} is any one of $\{(2,3),(3,4)\}$, $\{(3,4),(4,5)\}$ and $\{(4,5),(2,3)\}$. (b) In this example, dotted edges are in T_β^{opt} and solid edges in $COVER(h_\beta)$, $D_\beta^r = \{1,2\}$, h_1^+ increases the heights of 1 and 6, h_2^+ of 2, 3 and 4, h_3^+ of 3 alone and so on.

Now consider the tree $\mathcal{T}$ formed by the edges in T_β^{opt} on the vertex sets in D_β. We set an arbitrary vertex set D_β^r as the root of the tree $\mathcal{T}$. For each node v, let $d(v)$ denote the depth in tree $\mathcal{T}$, of the vertex set D_β^j that contains v.

For each node $v_i \in V$, consider a height increment h_i^+ centered at v_i, applied to h_β. (Note that each of these n increments are directly applied to h_β. They are *not* applied one after the other.) The height increment h_i^+: (1) increases the height of v_i to $\max(h_{opt}(v_i), h_\beta(v_i))$, (2) for every neighbor v_j of v_i such that $d(v_j) = d(v_i)+1$, increases the height of v_j to $\max(h_{opt}(v_j), h_\beta(v_j))$, and (3) does not increase the height of any other node. For an example, refer to Figure 5(b).

Let the benefit of height increment h_i^+ be denoted by b_i. Also, let the cost of h_i^+ be denoted by c_i. We now prove the following two lemmas about these above n height increments,

Lemma 5: The sum of benefits $\sum_{i=1}^{n} b_i \geq x$.

Proof: Consider any edge $e \in T_\beta^{opt}$. Let $e = (v_i, v_j)$. Without loss of generality, we assume that $d(v_j) = d(v_i)+1$. Since $T_\beta^{opt} \subseteq COVER(h_{opt})$, $e \in COVER(h_{opt})$; therefore, height of $h_{opt}(v_i)$ at v_i and height of $h_{opt}(v_j)$ at v_j, cover e. Since the height increment h_i^+ increases heights of v_i and v_j to at least their optimal values, the height increment h_i^+ covers edge e. Thus, each edge in T_β^{opt} is covered by some height increment in $\{h_1^+, \ldots, h_n^+\}$. Thus, the sum of the benefits of all these height increments is greater than or equal to the number of edges in T_β^{opt}. Now T_β^{opt} is a tree on $x+1$ vertices (where each vertex corresponds to a component of $COVER(h_\beta)$), and therefore has x edges. Thus, $\sum_{i=1}^{n} b_i \geq x$. ■

Lemma 6: The sum of the costs $\sum_{i=1}^{n} c_i \leq 2\,OPT$, where OPT is the total cost of the optimal solution.

Proof: To prove this lemma, we calculate the number of different height increments in $\{h_1^+, \ldots, h_n^+\}$, where the height of a particular node v_j changes. There are possibly two height increments where the height of a node v_j changes: (1) the height increment h_j^+ where v_j is the central node, and (2) if there is a node v_i such that the edge $(v_i, v_j) \in T_\beta^{opt}$ and $d(v_j) = d(v_i)+1$, then in the height increment h_i^+. It is important to note that, since the component containing v_j can have at most one parent in the tree T_β^{opt}, there is at most one node which satisfies criterion (2) for v_j. Thus, each vertex undergoes a change in height in at most two of the height increments $\{h_1^+, \ldots, h_n^+\}$.

Now observe that in any of the n height increments that we consider, the cost increment corresponding to the height increment at node v_j is either 0 (when its height remains $h_\beta(j)$) or $c(h_{opt}(v_j)) - c(h_\beta(v_j))$ (when, $h_{opt}(v_j) > h_\beta(v_j)$, and the height at v_j is changed to $h_{opt}(v_j)$). Thus the cost increment at a node due to a single height increment is at most $c(h_{opt}(v_j))$. Since, each vertex undergoes a change in height in at most two of the height increments, the sum of the cost of all n height increments $\sum_{i=1}^{n} c_i$ is at most $2\sum_{i=1}^{n} c(h_{opt}(v_j)) = 2OPT$. ■

We now need the following property. (The proof is omitted due to space constraints.)

Lemma 7: If $p_1, p_2, \ldots, p_k$ and $q_1, q_2, \ldots, q_k$ are two sequences of k positive real numbers, then $\min_i \left(\frac{p_i}{q_i}\right) \leq \frac{\sum_{i=1}^{k} p_i}{\sum_{i=1}^{k} q_i}$.

Proof of Lemma 4 (continued). Using the above lemma, we see that $\min_i \left(\frac{c_i}{b_i}\right) \leq \frac{\sum_{i=1}^{n} c_i}{\sum_{i=1}^{n} b_i}$. Now, since $\sum_{i=1}^{n} c_i \leq 2\,OPT$ and $\sum_{i=1}^{n} b_i \geq x$, $\min_i \left(\frac{c_i}{b_i}\right) \leq \frac{2\,OPT}{x}$. In other words, the minimum cost-to-benefit ratio among the n height increments is at most $2\,OPT/x$. Now, note that each of these n increments is a possible height increment for this phase of TC-ALGO. Thus, from Lemma 3, the minimum cost-to-benefit ratio for height increment selected by TC-ALGO in this phase is at most $4\,OPT/x$. ■

As an easy consequence of Lemma 4, we can prove the bound on approximation factor of TC-ALGO.

Theorem 8: The total cost of the height function returned by the greedy algorithm is within a factor of $4(1 + \log n)$ of the total cost (OPT) of the optimal height function.

Proof: Consider a phase where the number of components in the set of covered edges decrease from $x + 1$ to $x - b + 1$. The cost-to-benefit ratio for this phase is $\leq 4\ OPT/x$, from Lemma 4. From our definition of benefit, the benefit of the height increment chosen in this phase is at most b. Thus the cost increment in this phase is $\leq b \times 4\ OPT/x \leq \sum_{i=0}^{b-1} 4\ OPT/(x - i)$. Summing over all the phases, the total cost of the solution that is returned by the greedy algorithm is $\leq \sum_{i=1}^{n} 4\ OPT/i \leq 4(1 + \log n)OPT$. ■

V. NUMERICAL SIMULATIONS

In this section, we carry out extensive numerical simulations to evaluate our approximation algorithm with the optimal solution and also a naive heuristic. For our simulations, we generate synthetic topologies that aim to match the geographical structure of village clusters. We now describe our simulation setup in more detail.

Generating synthetic graph topologies. We consider a circular plane with a radius of 25Kms. We place nodes at random locations on this plane. We consider a link (u, v) between any two nodes, u and v, and for these simulations, assume just one obstacle, o_{uv}, located on the middle of this link. The height of the obstruction (h_o) is selected randomly with a maximum value of 20 meters - the typical height of trees and small houses in a rural setting. We then assign a weight w_{uv}, to the link equal to twice the *effective* height of the obstruction on this link.[9] As described earlier, the effective height of an obstruction, is the sum of the physical height (h_o) and 60% of r_f, the radius of the fresnel zone (computed using the formula from Section III).

Naive heuristic. In order to compare with our approximation algorithms, we describe a naive heuristic for selecting connected subgraphs and assigning heights to the nodes. As a first step, to select a connected subgraph of an input graph G, the heuristic computes the minimum spanning tree (MST), T of G (using the link weights computed as described above). Next, the heuristic has to assign heights to the nodes in G, so as to cover all the edges in T while minimizing the total cost. Given a set of links to be covered, we then formulate the height assignment problem as a simple LP, and compute the heights required on every node.

Comparing with the naive heuristic. We now compare the naive heuristic described above with our approximation algorithm. We consider graphs with number of nodes $n = 10, 15, 20, ..., 50$. For each value of n we generate 50 graph instances. For each graph, we compute C_{naive} the cost of the solution produced by the naive heuristic, and C_{approx}, cost of the approximation algorithm. In Fig. 6 we plot the mean and standard deviation over all graphs of $R_{naive} = \frac{C_{naive} - C_{approx}}{C_{approx}}$, for different values of n. We observe the approximation algorithm performs substantially better than the naive heuristic. On average, the solutions returned by the naive heuristic range from 60% (for $n = 10$) to as much as 225% more expensive (for $n = 50$) compared to the solution returned by the naive algorithm.

[9]While we assign weights to links between every pair of nodes (complete graph), we put a cap on the maximum height assigned to the tower at any node (h_{max} = 50meters). Therefore, if the tower height required to cover a link exceeds h_{max}, the link will not be selected by any of the algorithms.

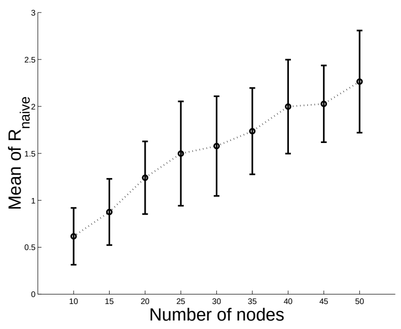

Fig. 6. Approximation algorithm vs. naive heuristic.

n	Mean (std. dev.) of R_{opt}
8	0.45 (0.30)
9	0.44 (0.25)
10	0.42 (0.23)
11	0.40 (0.25)

Fig. 7. Approximation algorithm vs. optimal solution.

Comparing with the optimal solution. We compute the optimal solution by solving an ILP that models the topology construction problem. We use the CPLEX LP-solver [11] to solve this ILP. This approach is, however, computationally very expensive, and the LP-solver could return solutions for graphs with at most 11 nodes. We compare the solution returned by our approximation algorithm with the optimal solution for graphs with number of nodes $n = 8, 9, 10, 11$. For each value of n we generate 50 graphs. For each graph we compute C_{approx}, the cost of the solution returned by our approximation algorithm, and C_{opt}, the cost of the optimal solution. We then compute the mean and standard deviation of $R_{opt} = \frac{C_{approx} - C_{opt}}{C_{opt}}$, over all graphs for different values of n.

The results presented in Figure 7 show that our approximation algorithm gives solutions that are $40 - 45\%$ more expensive than the optimal solution (for small values of n). Thus, our approximation algorithm performs much better than the worst case guarantee of $O(\log n)$ on the approximation factor. While this gap between our approximation algorithm and the optimal solution is not small, we wish to reiterate that computing the optimal solution (even if it has to be done only once) is practically infeasible for real-life networks. Moreover, our algorithm performs substantially better in practice than the naive heuristic.

To summarize, our numerical experiments demonstrate that our approximation algorithm performs well within its worst case performance bounds, and outperforms the naive heuristic by a substantial margin.

VI. CONCLUSIONS AND FUTURE WORK

In this work we have presented efficient approximation algorithms for the topology construction problem in rural mesh networks.

Our work introduces a number of open research problems. One immediate problem is to consider the case of $k \geq 2$ vertex or edge connectivity, similar to the power optimal network construction for k-connectivity [9], [10]. Another important research direction is the geometric version of this problem. In practice, all nodes within a certain distance of each other can establish a link. It is not clear whether the Topology Construction problem is NP-hard for this class of graphs (called *disk graphs*).

In this paper, we assumed that the location of the towers is fixed (within a village). A variant of the problem would make the location of the tower to be a variable (and maybe lower the cost of constructing a connected topology). We did not study this version of the problem because we find that, in practice, the inter-village distance dominates intra-village distance. Hence, this added flexibility in design would not result in any substantial difference in the cost, compared to our proposed algorithms for the existing model.

As a next step, we would also like to evaluate our algorithms over a deployed rural mesh network and evaluate the benefits over existing design approaches.

VII. Acknowledgements

We thank Ramesh Hariharan, Tsan-Sheng Hsu and Ajay Todimala for helpful discussions.

References

[1] Fresnel zone. http://en.wikipedia.org/wiki/Fresnel_zone.
[2] http://airjaldi.com.
[3] Digital gangetic plains: 802.11-based low-cost networking for rural areas, 2001-2004: A report. Technical report, IIT Kanpur, 2004.
[4] anw series heavy duty self supporting towers. http://www.anwireless.com/price.html.
[5] M. Bahramgiri, M. T. Hajiaghayi, and V. S. Mirrokni. Fault-tolerant and 3-dimensional distributed topology control algorithms in wireless multi-hop networks. *Wireless Networks*, 12(2), 2006.
[6] P. Bhagwat, B. Raman, and D. Sanghi. Turning 802.11 inside-out. In *HotNets -II*, 2003.
[7] G. Calinescu, I. Mandoiu, and A. Zelikovsky. Symmetric connectivity with minimum power consumption in radio networks. In *TCS*, 2002.
[8] P. Dutta, S. Jaiswal, K. Naidu, D. Panigrahi, R. Rastogi, and A. Todimala. Villagenet: A low-cost, 802.11-based mesh network for rural regions. In *WIreless Systems: Advanced Research and Development Workshop (WISARD)*, 2007.
[9] M. T. Hajiaghayi, N. Immorlica, and V. S. Mirrokni. Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks. In *MOBICOM*, 2003.
[10] M. T. Hajiaghayi, G. Kortsarz, V. S. Mirrokni, and Z. Nutov. Power optimization for connectivity problems. In *IPCO*, 2005.
[11] ilog cplex. http://www.ilog.com/products/cplex/.
[12] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theor. Comput. Sci.*, 243(1-2), 2000.
[13] P. N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted steiner trees. *J. Algorithms*, 19(1), 1995.
[14] E. L. Li, J. Y. Halpern, P. Bahl, Y.-M. Wang, and R. Wattenhofer. Analysis of a cone-based distributed topology control algorithm for wireless multi-hop networks. In *PODC*, 2001.
[15] Z. Nutov. Approximating minimum power covers of intersecting families and directed connectivity problems. In *APPROX-RANDOM*, 2006.
[16] R. Patra, S. Nedevschi, S. Surana, A. Sheth, L. Subramanian, and E. Brewer. WiLDNet: Design and implementation of high performance wifi based long distance networks. In *NSDI*, 2007.
[17] B. Raman and K. Chebrolu. Design and evaluation of a new MAC for long distance 802.11 mesh networks. In *Mobicom*, 2005.
[18] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability pcp characterization of np. In *STOC*, 1997.
[19] S. Sen and B. Raman. Long distance wireless mesh network planning: Problem formulation and solution. In *WWW*, 2007.
[20] V. Vazirani. *Approximation Algorithms*. Addison-Wesley, 2001.

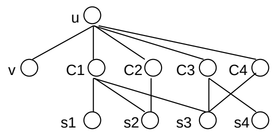

Fig. 8. The graph G in the NP-hardness reduction where the ground set $S = \{s1, s2, s3, s4\}$ and $\mathcal{C} = \{C1, C2, C3, C4\}$ with $C1 = \{s1, s2, s3\}$, $C2 = \{s2\}$, $C3 = \{s3, s4\}$ and $C4 = \{s3\}$.

Appendix

In this section, we will give a sketch of the NP-hardness reduction. The decision version of this problem asks the following question - given an input setting as described in section III and a bound B, does there exist a height function h such that the total cost of the towers is bounded by B and $COVER(h)$ is a connected spanning subgraph of the input graph?

Theorem 9: The decision version of the Topology Construction problem is NP-hard.

Proof: (*Sketch.*) To show that the TC problem is NP-hard, we give a reduction from any instance of the *set cover* problem [20] to an instance of the TC problem.

Given a collection $\mathcal{C}$ of subsets of a ground set S and a positive integer b, the minimum set cover problem is to determine if there exists a collection of subsets $\mathcal{C}' \subseteq \mathcal{C}$ such that $\mathcal{C}'$ is a cover for S and $|\mathcal{C}'| \leq b$.

The instance of the TC problem that we construct from this instance of the set cover problem has the following characteristics:

(1) The undirected graph $G = (V, E)$ is the incidence graph of $\mathcal{C}$ with some augmentation (see Figure 8). In particular, it contains set of nodes $V = S \cup \mathcal{C} \cup \{u, v\}$, where u and v are auxiliary nodes, and set of edges E comprising edges from u to each node $C \in \mathcal{C}$, edges between an element $s \in S$ and a subset $C \in \mathcal{C}$ if and only if $s \in C$, and an edge between u and v. (2) On any edge $e = (u, v) \in E$, there is a single obstruction of height H equidistant from u and v. (3) $h_{max} = 2H$. (4) Cost function c is defined as $c(h) = h, \forall h$. (5) Bound $B = 2H(b+1)$, where b is the bound on the number of subsets selected in the set cover problem.

We now state the following lemma (whose proof is omitted due to space constraints) which establishes the equivalence of the two problems and completes the proof.

■

Lemma 10: There exists a set cover of size at most b if and only if the corresponding instance of the TC problem has a solution h of total cost at most B.

We can also show that this reduction is "gap-preserving",[10] thus proving Theorem 1. Since it is NP-hard to approximate the minimum set cover problem to a factor of $o(\log n)$ [18], it is also NP-hard to obtain an $o(\log n)$ approximation to the TC problem. So, the best that we can realistically hope for is an $O(\log n)$ approximation algorithm. As shown in the paper, we meet this bound.

[10]A *gap preserving* reduction of problem A to problem B is one where the hardness of approximation results for problem A are retained by problem B. For a formal definition and details, refer to [20].