

Power Efficient Range Assignment in Ad-hoc Wireless Networks

E. Althaus G. Călinescu* I.I. Măndoiu† S. Prasad‡ N. Tchernvenski* A. Zelikovsky‡

Abstract—We study the problem of assigning transmission ranges to the nodes of ad hoc wireless networks so that to minimize power consumption while ensuring network connectivity. We give (1) an exact branch and cut algorithm based on a new integer linear program formulation solving instances with up to 35-40 nodes in 1 hour; (2) a minimum spanning tree (MST) based 2-approximation algorithm for the case when power requirements are asymmetric; (3) an improved analysis for two approximation algorithms recently proposed by Călinescu et al. (TCS’02), decreasing the currently best factor to $5/3 + \epsilon$; (4) a comprehensive experimental study comparing new and previously proposed heuristics with the above exact and approximation algorithms.

INTRODUCTION

Ad hoc wireless networks have received significant attention in recent years due to their potential applications in battlefield, emergency disaster relief, and other application scenarios (see, e.g., [1], [3], [2], [4], [6], [7], [9], [13], [16], [15]). Unlike wired networks or cellular networks, no wired backbone infrastructure is installed in ad hoc wireless networks. A communication session is achieved either through single-hop transmission if the recipient is within the transmission range of the source node, or by relaying through intermediate nodes otherwise. When a transmission is made by a node it can be received by all nodes within its transmission range. This feature is extremely useful for energy-efficient multicast and broadcast communications.

For the purpose of energy conservation, each node can (possibly dynamically) adjust its transmitting power, based on the distance to the receiving node and the background noise. In the most commonly used power-attenuation model [10], the signal power falls as $\frac{1}{r^\kappa}$ where r is the distance from the transmitter antenna and κ is a real *constant* dependent on the wireless environment, typically between 2 and 4. Another common assumption is that all receivers have the same power threshold for signal detection, typically normalized to one. With this assumption, the power requirement for supporting a link between nodes u and v separated by a distance r is

$$p(u, v) = p(v, u) = r^\kappa \quad (1)$$

In practice power requirements may be asymmetric, i.e., $p(u, v) \neq p(v, u)$, due to non-uniform receiver sensitivity

Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, D-66123 Saarbrücken, Germany. E-mail: althaus@mpi-sb.mpg.de.

* Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616. E-mail: {calinesc,tchenic}@cs.iit.edu.

† Department of Computer Science & Engineering, UC San Diego, La Jolla, CA 92093. E-mail: mandoiu@cs.ucsd.edu.

‡ Department of Computer Science, Georgia State University, Atlanta, GA 30303. E-mail: {sprasad,alexz}@cs.gsu.edu.

and/or environmental conditions. Unless explicitly stated otherwise, in this paper we assume symmetric power requirements.

Having every link established in both directions simplifies one-hop transmission protocols by allowing acknowledgement messages to be sent back for every packet (see, for example [14]). This motivates the study of the MIN-POWER SYMMETRIC CONNECTIVITY problem, where a link is established only if both nodes have transmission range at least as big as the distance between them, and the goal is to ensure that the network is connected [1], [5].

Formally, let V be a set representing network nodes. A *power assignment* is a function $r : V \rightarrow R_+$. We say that a *unidirectional link* from node u to node v is established under the power assignment r if $r(u) \geq p(u, v)$. Similarly, a *bidirectional link* uv is established under the range assignment r if $r(u) \geq p(u, v)$ and $r(v) \geq p(v, u)$. Let $B(r)$ denote the set of all bidirectional links established between pairs of nodes in V under the transmission range r .

MIN-POWER SYMMETRIC CONNECTIVITY: Given a set of nodes V and symmetric power requirements $p(u, v)$, $(u, v) \in V \times V$, find a power assignment $r : V \rightarrow R_+$ minimizing $\sum_{v \in V} r(v)$ subject to the constraint that the graph $(V, B(r))$ is connected.

Implicit in the work of Clementi, Penna, and Silvestri [4] is a proof that MIN-POWER SYMMETRIC CONNECTIVITY in E^2 is NP-Hard. Therefore, we study approximation algorithms and heuristics for the problem. In this paper we make the following contributions:

- We give an exact branch and cut algorithm based on a new integer linear program formulation for the problem. Experimental results show that the branch and cut algorithm solves instances with 25 nodes in less than one minute and instances with up to 35-40 nodes in 1 hour. (Section I.)
- We give a minimum spanning tree (MST) based 2-approximation algorithm for MIN-POWER SYMMETRIC CONNECTIVITY WITH ASYMMETRIC POWER REQUIREMENTS, to the best of our knowledge this version of the problem has not been considered before. (Section II.)
- We give an improved analysis for two approximation algorithms from [5], decreasing the currently best approximation factor to $5/3 + \epsilon$.¹ (Appendix.)
- We present a comprehensive experimental study comparing new and previously proposed heuristics with the above

¹The *approximation factor* of an algorithm A for a minimization problem is the supremum, over all possible instances I , of the ratio between the cost of the output of A when running on I and the cost of an optimal solution for I (the smaller the performance ratio, the better). We say that A is an α -*approximation algorithm* if its approximation ratio is at most α .

exact and approximation algorithms. Experimental results show an average of 5-6% reduction in power consumption compared to the MST based solution. (Section III.)

A. Related Work

Kirousis, Kranakis, Krizanc, and Pelc [6] give a minimum spanning tree (MST) based 2-approximation algorithm for MIN-POWER SYMMETRIC CONNECTIVITY (their algorithm is actually designed for the COMPLETE RANGE ASSIGNMENT problem discussed below). Călinescu et al. [5] improve the approximation ratio under 2 by exploiting similarities between MIN-POWER SYMMETRIC CONNECTIVITY and the classic STEINER TREE problem. In particular, [5] gives a fully polynomial $1 + \ln 2$ approximation scheme² based on [18] and a more practical $15/8$ approximation algorithm based on [17].

The objective of minimizing the total power has been also addressed under the specific power requirements given by (1) and the related *asymmetric* connectivity model, in which unidirectional links give raise to a directed graph on V . Four problems have been studied under this model.

1. ASYMMETRIC UNICAST, which requires establishing a minimum power directed path from a source s to a destination t . ASYMMETRIC UNICAST is easily solved in polynomial time by shortest-path algorithms. The related MIN-POWER SYMMETRIC UNICAST problem can also be solved efficiently by a shortest-path computation in an appropriately constructed graph [5].

2. ASYMMETRIC BROADCAST, which requires establishing a minimum power arborescence rooted at a given vertex s [13], [16]. Clementi et al. [3] prove that ASYMMETRIC BROADCAST is NP-Hard when the nodes are in E^2 . The best known approximation algorithm for ASYMMETRIC BROADCAST [15], based on computing a minimum spanning tree, has performance ratio of at most 12 when the nodes are in E^2 . Chu and Nikolaidis [2] give an experimental study of asymmetric broadcast algorithms for mobile ad hoc networks. As noted in [5], the related SYMMETRIC BROADCAST problem is identical to MIN-POWER SYMMETRIC CONNECTIVITY.

3. ASYMMETRIC MULTICAST, in which one is given a root s and a set of terminals T , and the goal is to establish a minimum-power branching rooted at s which reaches all vertices of T . As a generalization of ASYMMETRIC BROADCAST, ASYMMETRIC MULTICAST is also NP-Hard, and based on the work of [15], it is immediate that a minimum Steiner tree would give an approximation ratio of 12ρ , where ρ is the approximation for Steiner tree in graphs (the best result known at this moment, given in [11], is $\rho = 1 + \frac{1}{2} \ln 3 + \varepsilon$).

4. COMPLETE RANGE ASSIGNMENT, in which the objective is establishing a strongly connected subgraph of V . Kirousis, Kranakis, Krizanc, and Pelc [6] prove that COMPLETE RANGE ASSIGNMENT in E^3 is NP-Hard and, based on the minimum spanning tree, give a 2-approximation algorithm. As opposed to the ASYMMETRIC BROADCAST approximation of [15], the COMPLETE RANGE ASSIGNMENT approxi-

²A fully polynomial α -approximation scheme is a family of algorithms A_ε such that, for every $\varepsilon > 0$, $A_\varepsilon(1)$ has performance ratio at most $\alpha + \varepsilon$, and (2) runs in time polynomial in the size of the instance and $1/\varepsilon$.

mation of [6] is valid in arbitrary graphs (that is, the distance between two points could be arbitrary, not necessarily Euclidean). Clementi, Penna, and Silvestri [4] give an elaborate reduction proving that COMPLETE RANGE ASSIGNMENT in E^2 is also NP-Hard. As shown in [5], the power for the asymmetric COMPLETE RANGE ASSIGNMENT can be twice less than the power for MIN-POWER SYMMETRIC CONNECTIVITY.

I. INTEGER LINEAR PROGRAM FORMULATION

In this section we give an integer linear program (ILP) formulation for MIN-POWER SYMMETRIC CONNECTIVITY and describe a branch and cut algorithm based on it. The results in Section III show that the algorithm is practical for instances with up to 35-40 nodes.

We begin by reformulating MIN-POWER SYMMETRIC CONNECTIVITY in graph theoretical terms. Let $G = (V, E, c)$ be an edge-weighted graph and uv denote the undirected edge between nodes u and v . The cost $c(uv)$ of an edge $uv \in E$ corresponds to the (symmetric) power requirement $p(u, v) = p(v, u)$. For a node $u \in V$ and a spanning tree T of G , let uu_T be the maximum cost edge incident to u in T , i.e., $uu_T \in T$ and $c(uu_T) \geq c(uv)$ for all $uv \in T$. The *power cost* of a spanning tree T is

$$p(T) = \sum_{u \in V} c(uu_T)$$

Since any connected graph contains a spanning tree, an equivalent formulation of MIN-POWER SYMMETRIC CONNECTIVITY is to ask for a spanning tree with minimum power-cost in the complete graph on V with edge costs given by $c(uv) = \|uv\|^\kappa$. Thus, MIN-POWER SYMMETRIC CONNECTIVITY can be reformulated as follows:

MINIMUM POWER-COST SPANNING TREE: Given a connected edge-weighted graph $G = (V, E, c)$, find a spanning tree T of G with minimum power-cost.

To formulate MINIMUM POWER-COST SPANNING TREE as a linear integer program we use two types of binary decision variables:

- x_{uv} for all $uv \in E$; x_{uv} is set to 1 if uv belongs to the selected spanning tree T and to 0 otherwise. We call these variables the *tree variables*.
- $y_{\overline{uv}}$ for all $\overline{uv} \in \overline{E} := \{\overline{uv}, \overline{vu} \mid uv \in E\}$; $y_{\overline{uv}}$ is set to 1 if $u_T = v$ (i.e., if $uv \in T$ and $c(uv) \geq c(uw)$ for all $uw \in T$). We call these variables the *range variables*.

Note that there are $|E|$ tree variables and $|\overline{E}| = 2|E|$ range variables. Let ST be set of the incidence vectors of all spanning trees of G (viewed as subsets of E). Our ILP formulation is as follows.

$$\begin{aligned} \min \quad & \sum_{\overline{uv} \in \overline{E}} c(uv) y_{\overline{uv}} \\ \text{s.t.} \quad & \sum_{v \in V \mid \overline{uv} \in \overline{E}} y_{\overline{uv}} = 1, \quad \forall u \in V \quad (2) \end{aligned}$$

$$x_{uv} \leq \sum_{\overline{uv} \in \overline{E} \mid c(uv) \geq c(uw)} y_{\overline{uv}}, \quad \forall \overline{uv} \in \overline{E} \quad (3)$$

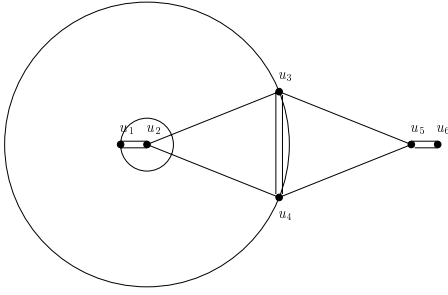


Fig. 1. Let $x_e = 1/2$ for all edges in the picture ($x_e = 1$, if there are two parallel edges). Let range variables $y_{\overline{uv}}$ be equal to $1/2$ for $v = u_1, u_3$, and to 0 otherwise. Then constraints of type (2) and (3), are satisfied, but the constraint (5) is violated for $S = \{u_1, u_2\}$.

$$\begin{aligned} x &\in \text{conv}(ST) \\ y &\in \{0, 1\}^{|\overline{E}|} \\ x &\in \{0, 1\}^{|E|} \end{aligned} \quad (4)$$

The constraints (2) enforce that we select exactly one range variable for every node $v \in V$, i.e., we properly define the range of each node. The constraints (3) enforce that an edge uv is included in the tree only if the range of each endpoint is at least the cost of the edge. The constraints (4) enforce that the tree variables indeed form a spanning tree. There are several well known linear descriptions for $x \in \text{conv}(ST)$. We use the following, most famous formulation: $x \in \text{conv}(ST) \Leftrightarrow x \geq 0$, $\sum_{e \in E} x_e = |V| - 1$ and $\sum_{e \in \gamma(S)} x_e \leq |S| - 1$ for all $S \subseteq E$.

To solve the ILP we use branch and cut, i.e., we drop the integrality constraints and solve the corresponding LP relaxation. If the solution of the LP is integral, we found the optimal solution, otherwise we pick a variable with a fractional value and split the problem into two subproblems by setting the variable to 0 and 1 in the subproblems. We solve the subproblems recursively and disregard a subproblem if its LP bound is worse than the best known solution.

Since there are an exponential number of inequalities in this formulation of spanning trees, we can not solve the LP directly. Instead, we start with a small subset of these inequalities and algorithmically test whether the LP solution violates an inequality which is not in the current LP. If so, we add the inequality to the LP, otherwise we found the solution of the LP with the exponential number of inequalities. The inequalities added to the LP if needed are called *cutting planes*, algorithms that find violated cutting planes are called *separation algorithms*.

In our case, the initial LP consists of the constraints (2) and (3), the constraint $\sum_{e \in E} x_e = |V| - 1$, and the bound constraints, i.e., the constraints $0 \leq x \leq 1$ and $0 \leq y \leq 1$. The only constraints added on demand are the constraints $\sum_{e \in \gamma(S)} x_e \leq |S| - 1$ for all $S \subseteq E$. A separation algorithm for these inequalities is due to Padberg and Wolsey [8].

The running time of a branch and cut algorithm can be improved by tightening the LP relaxation, i.e., by finding additional inequalities which are valid for all integer points, but may be violated by solutions to the LP relaxation (Figure 1 shows an example). We use the following class of valid inequalities. Let $S \subset V$. For every $u \in S$ let $u_S \in V \setminus S$ so that $c(uu_S) \leq c(uv)$

for all $v \in V \setminus S$. The inequality

$$\sum_{u \in S} \sum_{v \in V | c(uv) \geq c(uu_S)} y_{\overline{uv}} \geq 1 \quad (5)$$

is valid for the problem above. We can argue as follows. There is at least one edge in the spanning tree T crossing the cut S . Let uv be such an edge and $u \in S$. Then $c(uv) \geq c(uu_S)$ and the range of u is at least $c(uv)$. Thus $\sum_{v \in V | c(uv) \geq c(uu_S)} y_{\overline{uv}}$ is one and the inequality is valid.

Since we do not have a separation algorithm for these inequalities, we use the following heuristic to separate some of them. We chose an arbitrary node u . For every node $v \in V \setminus \{u\}$, we compute the minimal directed cut from u to v and from v to u , where the capacity of an edge xy is given by $\sum_{xw | c(xw) \geq c(xy)} y_{xw}$. For all computed cuts, we test whether the corresponding inequality is violated.

II. APPROXIMATION ALGORITHMS

In this section we give a 2-approximation algorithm for MIN-POWER SYMMETRIC CONNECTIVITY WITH ASYMMETRIC POWER REQUIREMENTS, and give an improved analysis for two approximation algorithms proposed in [5] for MIN-POWER SYMMETRIC CONNECTIVITY.

Theorem 1: The minimum spanning tree with respect to edge weights given by $p(u, v) + p(v, u)$ has a power cost within twice of optimum for MIN-POWER SYMMETRIC CONNECTIVITY WITH ASYMMETRIC POWER REQUIREMENTS.

Proof: Let MST be the minimum spanning tree with respect to edge weights given by $p(u, v) + p(v, u)$ and let OPT be the tree with minimum power cost. Then the power cost of MST is

$$\begin{aligned} p(MST) &= \sum_{v \in V} \max_{u | uv \in MST} p(u, v) \\ &\leq \sum_{v \in V} \sum_{u | uv \in MST} p(u, v) = c(MST) \leq c(OPT) \end{aligned}$$

where $c(T) = \sum_{uv \in T} (p(u, v) + p(v, u))$. The theorem follows by observing that, for every tree T , $c(T) \leq 2p(T)$. Indeed, let r be an arbitrary node of T , and let T_{in} and T_{out} be the two arborescences obtained from T by directing all edges towards, respectively away from r . Since the power range of each node u of T must exceed the power requirement of the unique edge leaving u in T_{in} , respectively entering u in T_{out} , it follows that $p(T) \geq \sum_{\overline{uv} \in T_{in}} p(u, v)$ and $p(T) \geq \sum_{\overline{uv} \in T_{out}} p(u, v)$. Hence,

$$2p(T) \geq \sum_{uv \in T} (p(u, v) + p(v, u)) = c(T)$$

Remark. The following example from [5] shows that the factor of 2 in Theorem 1 cannot be improved, even in the case of symmetric power requirements. Consider $2n$ points located on a single line such that the distance between consecutive points alternates between 1 and $\varepsilon < 1$ (see Figure 2) and let $\kappa = 2$. Then the minimum spanning tree MST connects consecutive neighbors and has power-cost $p(MST) = 2n$. On the other hand,

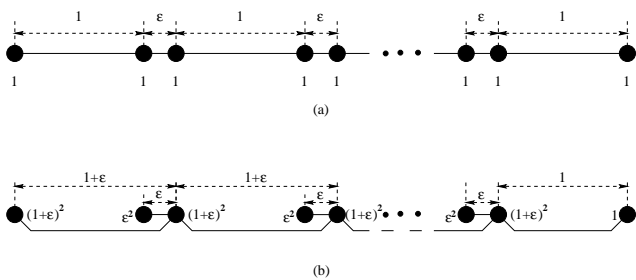


Fig. 2. Tight example for the performance ratio of the MST algorithm ($\kappa = 2$). (a) The MST-based range assignment needs total power $2n$. (b) Optimum range assignment has total power $n(1 + \varepsilon)^2 + (n - 1)\varepsilon^2 + 1 \rightarrow n + 1$.

Input: Edge-weighted graph $G = (V, E, c)$

Output: Spanning tree of G

$T \leftarrow MST(G)$

$H \leftarrow \emptyset$

Repeat forever

Find a fork K with the maximum $g = gain_T(K)$

If $g \leq 0$ **then exit repeat**

$H \leftarrow H \cup K$

$G \leftarrow G/K$

$T \leftarrow T/K$

Output $T \cup H$

Fig. 3. The greedy fork-contraction algorithm.

the tree T with edges connecting each other node (see Figure 2(b)) has power-cost equal $p(T) = n(1 + \varepsilon)^2 + (n - 1)\varepsilon^2 + 1$. When $n \rightarrow \infty$ and $\varepsilon \rightarrow 0$, we obtain that $p(MST)/p(T) \rightarrow 2$.

Recently, [5] proposed an algorithm with approximation ratio of $7/4 + \varepsilon$ based on polynomial time approximation scheme for computing the minimum-weight spanning cactus in a 3-uniform hypergraph, and a more practical greedy algorithm with approximation ratio of $15/8$. In the appendix we improve the approximation factors of these two algorithms to $5/3 + \varepsilon$ and $11/6$, respectively.

The greedy algorithm in [5] (see Figure 3) iteratively improves the MST by inserting the best *fork*, i.e., pair of edges of G sharing a node. Note that the power cost of fork $K = \{e_1, e_2\}$ is $p(K) = 2 \max\{c(e_1), c(e_2)\} + \min\{c(e_1), c(e_2)\}$. The edges of fork K added to the MST replace the highest cost edges in the two created cycles. As a result the power cost may decrease. The *gain* of fork K is defined by

$$gain(K) = 2mst(G) - 2mst(G/K) - p(K)$$

where $mst(G)$ the minimum cost of a spanning tree of G , G/K is the graph obtained from G by collapsing all nodes of K into a single node.

III. EXPERIMENTAL STUDY

We have implemented the exact branch and cut algorithm described in Section I (OPT), the greedy fork-contraction algorithm of [5] (GFC), and three new heuristics:

- A simple edge-switching (ES) heuristic that starts from the MST, and repeatedly replaces a tree edge with a non-tree

edge re-establishing connectivity. At every step, the algorithm chooses the pair of edges that results in the largest reduction in power cost; the process is repeated as long as improvement is still possible. We simulated a distributed implementation of the algorithm in which only non-tree edges that connect nodes within a small number of tree-hops from each other can be considered.

- A heuristic performing both edge and fork switching (EFS). At every step the algorithm chooses the edge or the fork whose addition to the tree leads to the largest reduction in power cost. Unlike GFC, forks are not contracted, which means that an edge in an added fork can later be removed from the tree by other edge or fork switches.
- A Kruskal-like heuristic (KR) that starts with isolated nodes and iteratively adds an edge connecting two different components with *minimum increase* in power cost. A similar heuristic (called incremental search) was studied by Chu and Nikolaidis for computing low-power ASYMMETRIC BROADCAST trees in a mobile environment.

We included in our comparison faster versions of OPT and GFC, OPT-D and GFC-D, which speed-up the computation by working on the Delaunay graph defined by the nodes instead of the complete graph. For EFS we also implemented a faster version, EFS-D, in which only forks consisting of Delaunay edges (but still all non-tree edges) can be considered for switching.

All algorithms were implemented in C++, including the branch and bound algorithm whose implementation is built on SCIL [12]. The heuristics were compiled using gpp with `-O2` optimization, and run on an AMD Duron 600MHz PC. The experiments were run on randomly generated testcases. For each instance size n between 10 and 100, in increments of 5, 50 different instances were generated by choosing n points uniformly at random from a grid of size $10,000 \times 10,000$.

In Table I and Figure 4 we report the *percent improvement over MST*, i.e.,

$$100 \times \frac{p(MST) - p(Algo)}{p(MST)}$$

for the compared algorithms. Running times are reported in Table II. The results show that OPT has practical running time up to 35 nodes, and produces an average improvement over MST of 5-6%. The Delaunay version of OPT has practical runtime up to 60 nodes, but gives slightly worse solutions.

The provably good GFC algorithm, its faster Delaunay version, GFC-D, as well as the natural Kruskal-like heuristic KR are all very fast, but give less than half of the optimum improvement. In contrast, EFS, EFS-D, and even the distributed ES heuristic, come on the average within a fraction of a percent of the optimal improvement with very well scaling runtime.

REFERENCES

- [1] D.M. Blough, M. Leoncini, G. Resta, and P. Santi. On the symmetric range assignment problem in wireless ad hoc networks. In *Proc. 2nd IFIP International Conference on Theoretical Computer Science*, page (to appear), 2002.
- [2] T. Chu and I. Nikolaidis. Energy efficient broadcast in mobile ad hoc networks. In *Proc. AD-HOC NetWorks and Wireless*, 2002 (to appear).

n	OPT	OPT-D	ES	EFS	EFS-D	KR	GFC	GFC-D
10	4.01	3.66	3.81	4.00	3.94	0.49	1.39	1.19
15	4.77	4.26	4.48	4.70	4.51	1.72	1.56	0.48
20	5.84	5.17	5.46	5.75	5.47	2.54	2.01	1.40
25	5.63	4.72	4.78	5.53	5.12	2.19	1.56	0.72
30	5.46	4.90	4.87	5.36	5.03	1.77	1.65	0.24
35	5.68	5.11	5.04	5.60	5.40	2.13	1.93	0.96
40	5.41 ³	4.82	5.01	5.51	5.25	1.82	1.37	0.26
45	—	5.37	5.13	5.77	5.47	2.17	2.22	0.67
50	—	5.36	5.55	5.90	5.62	2.45	2.03	0.33
55	—	6.09	5.61	6.54	6.21	2.65	2.60	1.19
60	—	5.46 ⁴	5.25	6.06	5.73	2.31	2.15	0.50
65	—	—	5.01	5.80	5.56	2.30	1.65	0.38
70	—	—	5.12	6.01	5.60	2.41	1.94	0.24
75	—	—	5.10	5.78	5.50	2.46	1.69	0.48
80	—	—	5.14	6.03	5.77	2.88	2.00	0.64
85	—	—	4.73	5.69	5.37	2.52	1.82	0.39
90	—	—	5.42	6.30	6.01	2.84	2.18	0.38
95	—	—	5.29	6.08	5.81	2.35	1.73	0.19
100	—	—	5.45	6.25	6.09	2.56	2.30	0.99

TABLE I
PERCENT IMPROVEMENT OVER THE MST (AVERAGES OVER 50
INSTANCES OF EACH SIZE).

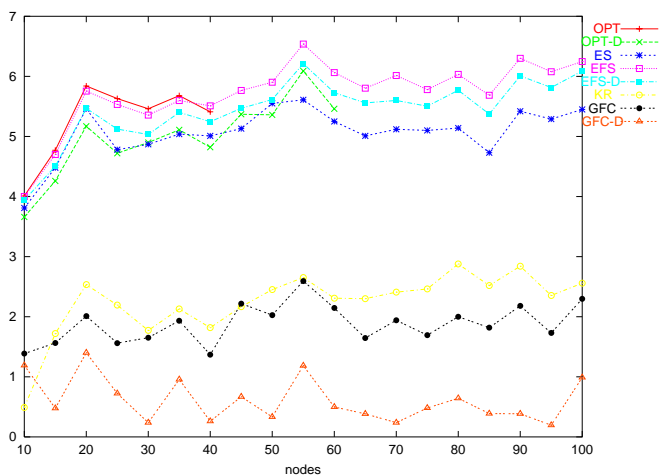


Fig. 4. Average percent improvement over MST for the implemented algorithms.

n	OPT	OPT-D	ES	EFS	EFS-D	KR	GFC	GFC-D
10	0.67	0.10	0.00	0.00	0.00	0.00	0.00	0.00
15	5.68	0.43	0.00	0.02	0.00	0.00	0.00	0.00
20	22.2	1.19	0.00	0.10	0.00	0.00	0.00	0.00
25	58.9	3.46	0.00	0.26	0.00	0.00	0.00	0.00
30	201	6.49	0.00	0.61	0.00	0.00	0.00	0.00
35	712	11.2	0.00	1.16	0.02	0.01	0.00	0.00
40	4725 ³	52.1	0.00	2.13	0.03	0.01	0.00	0.00
45	—	109	0.00	3.71	0.05	0.00	0.03	0.03
50	—	181	0.02	5.50	0.05	0.00	0.02	0.02
55	—	653	0.05	9.03	0.05	0.00	0.03	0.03
60	—	573 ⁴	0.05	12.48	0.06	0.00	0.05	0.05
65	—	—	0.05	17.9	0.09	0.04	0.03	0.03
70	—	—	0.03	25.5	0.10	0.04	0.01	0.01
75	—	—	0.02	33.4	0.09	0.02	0.00	0.00
80	—	—	0.05	44.9	0.12	0.00	0.00	0.00
85	—	—	0.06	55.0	0.16	0.00	0.00	0.00
90	—	—	0.09	75.5	0.21	0.00	0.00	0.00
95	—	—	0.11	101	0.26	0.00	0.05	0.05
100	—	—	0.14	123	0.32	0.00	0.05	0.05

TABLE II
RUNTIME (IN CPU SECONDS) FOR THE COMPARED ALGORITHMS
(AVERAGES OVER 50 INSTANCES OF EACH SIZE).

- [3] A.E.F. Clementi, P. Crescenzi, P. Penna, G. Rossi, and P. Vocca. On the complexity of computing minimum energy consumption broadcast subgraphs. In *Symposium on Theoretical Aspects of Computer Science*, pages 121–131, 2001, extended version available at <http://www.mat.uniroma2.it/~penna/papers/stacs01-TR.ps.gz>.
- [4] A.E.F. Clementi, P. Penna, and R. Silvestri. On the power assignment problem in radio networks. *Electronic Colloquium on Computational Complexity (ECCC)*, (054), 2000.
- [5] G. Călinescu, I.I. Măndoiu, and A.Z. Zelikovsky. Symmetric connectivity with minimum power consumption in radio networks. In *Proc. 2nd IFIP International Conference on Theoretical Computer Science*, 2002 (to appear).
- [6] L.M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243:289–305, 2000.
- [7] E. Lloyd, R. Liu, M. Marathe, R. Ramanathan, and S.S. Ravi. Algorithmic aspects of topology control problems for ad hoc networks. In *Proc. ACM MobiHoc*, page (to appear), 2002.
- [8] M. Padberg and L. Wolsey. Trees and cuts. *Anal. of Discrete Mathematics*, 17:511–517, 1983.
- [9] Ram Ramanathan and Regina Hain. Topology control of multihop wireless networks using transmit power adjustment. In *INFOCOM (2)*, pages 404–413, 2000.
- [10] T.S. Rappaport. *Wireless Communications: Principles and Practices*. Prentice Hall, 1996.
- [11] G. Robins and A. Zelikovsky. Improved Steiner tree approximation in graphs. In *Proceedings of the 11th ACM-SIAM Annual Symposium on Discrete Algorithms*, pages 770–779, 2000.
- [12] SCIL—Symbolic Constraints for Integer Linear programming. www.mpi-sb.mpg.de/SCIL.
- [13] S. Singh, C.S. Raghavendra, and J. Stepanek. Power-aware broadcasting in mobile ad hoc networks. In *Proceedings of IEEE PIMRC*, 1999.
- [14] A.S. Tanenbaum. *Computer Networks (3rd edition)*. Prentice Hall, 1996.
- [15] P.-J. Wan, G. Călinescu, X.-Y. Li, and O. Frieder. Minimum energy broadcast routing in static ad hoc wireless networks. In *Proc. IEEE INFOCOM*, vol. 2, pages 1162–1171, 2001.
- [16] J.E. Wieselthier, G.D. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *Proc. IEEE INFOCOM*, pages 585–594, 2000.
- [17] A. Zelikovsky. An 11/6-approximation algorithm for the network Steiner problem. *Algorithmica*, 9:463–470, 1993.
- [18] A. Zelikovsky. Better approximation bounds for the network and Euclidean Steiner tree problems. Technical Report CS-96-06, Department of Computer Science, University of Virginia, 1996.

APPENDIX

The proof of the approximation ratios in [5] is based on the notion of *3-restricted decomposition* of a tree T , which is a partition of the edges of T into forks and individual edges. The power-cost of a 3-restricted decomposition Q , $p(Q)$, is the sum of power-costs of its elements, i.e., forks and individual edges. It is proved in [5] that, if there exists a 3-restricted decomposition Q with $P(Q) \leq \rho p(T)$, then the greedy algorithm in Figure 3 has approximation ratio $1 + \rho/2$ and there is PTAS with approximation ratio $\rho + \epsilon$. In the rest of the appendix we prove that $\rho \leq 5/3$ improving the ratio $7/4$ from [5].

Theorem 2: For any tree T , there is a 3-restricted decomposition Q of T such that

$$p(Q) \leq \frac{5}{3}p(T) \quad (6)$$

Proof: Without loss of generality, we assume that no two edges have the same cost (if there are ties, then we break them arbitrarily).

For any vertex u , we denote by $\max(u)$ the maximum edge in T incident to u (see Figure 5(a)). An edge $uv \in E$ is called a *root* if $uv = \max(u) = \max(v)$. Let R be the set of all roots

³Average does not include two instances not solved within one day.

⁴Average does not include one instance not solved within one day.

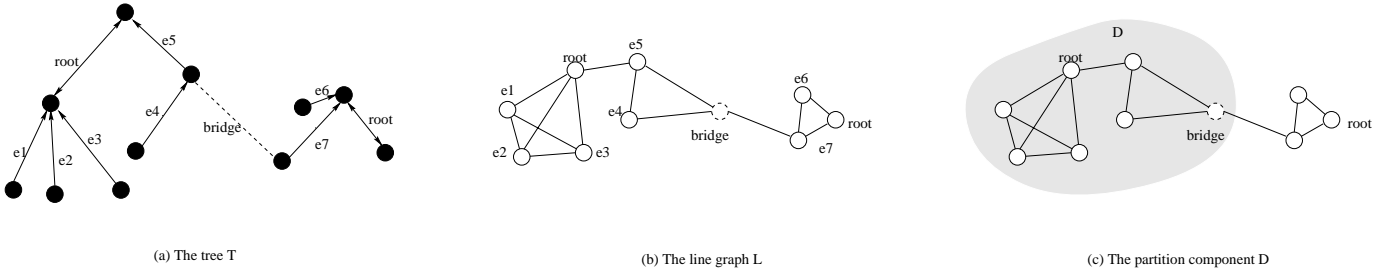


Fig. 5. (a) The original graph T : each vertex has a single outgoing arc denoting its maximum incident edge, double arcs are roots and dashed edges are bridges. (b) The corresponding line graph L : vertices represent edges of T and edges represent forks of T . (c) The partition of L into partition components: the dashed bridge is attached to the partition component D .

in T . An edge $uv \in E$ is called a *bridge* if $uv \neq \max(u)$ and $uv \neq \max(v)$. Let B be the set of all bridges in T . In the power-cost of T , each non-bridge and non-root edge is counted once, each root is counted twice (for both its endpoints) and each bridge is not counted at all:

$$p(T) = c(T) + c(R) - c(B). \quad (7)$$

Note also that $c(R) \geq c(B)$ simply because $p(T) \geq c(T)$.

Let $L = (V(L), E(L), w)$ be the weighted line graph of T (see Figure 5(b)), such that

- the vertices of L correspond to edges of T , $V(L) = E(T)$;
- two vertices $e, e' \in V(L)$ are adjacent if the corresponding edges e and e' are adjacent in T ;
- the edge $(e, e') \in E(L)$ has weight $w(e, e') = \min\{c(e), c(e')\}$

Note that any edge of L corresponds to a fork in T . Similarly, an arbitrary matching X of L corresponds to the 3-restricted decomposition of T (edges of X correspond to forks and isolated vertices correspond to isolated edges in T) which we denote Q_X . It is easy to see that $p(Q_X) = 2c(T) - w(X)$. The rest of the proof of (6) constructs a matching X in L satisfying

$$w(X) \geq \frac{c(T) - c(R) + c(B)}{3} \quad (8)$$

which yields (6). Indeed, $c(B) \leq c(R)$, and (7) implies that

$$\begin{aligned} p(Q_X) &= 2c(T) - w(X) \\ &\leq \frac{5}{3}c(T) + \frac{1}{3}c(R) - \frac{1}{3}c(B) \\ &\leq \frac{5}{3}p(T) \end{aligned}$$

Note that each connected component of $T - B$ (and, therefore $L - B$) has a single root (see Figure 5(a)-(b)). We say that connected components are *adjacent* if there is a bridge connecting them. We order the connected components of $L - B$ such that each component, except the first one, is adjacent to a single previous component by so called *attached bridge*; such ordering is possible since T is a tree. Thus we partition all edges of T into edge subsets each consisting of a connected component of $T - B$ and (possibly) an attached bridge. Let D be a *partition component*, i.e., a subgraph of L induced by any of these edge subsets (see Figure 5(c)). The matching X is constructed as a

union of matchings X_D for each partition component D . Then (8) will follow from the following inequality

$$w(X_D) \geq \frac{c(D) - c(\text{root}) + c(\text{bridge})}{3}, \quad (9)$$

where root is the root of D and bridge is the attached bridge of D .

By Edmonds' Theorem it is sufficient to construct a fractional matching X_D satisfying (9). A *fractional matching* is an LP relaxation of matching with weight equal to

$$\sum_{(e, e') \in E(D)} x(e, e') w(e, e')$$

where $x(e, e')$ is an assignment of nonnegative fractions to every edge $(e, e') \in E(D)$ such that

- (i) The sum of fractions assigned to all edges incident to the same vertex $e \in V(D)$ is at most 1.
- (ii) The sum of fractions assigned to all edges with both endpoints in the same set of $2k + 1$ vertices in $V(D)$ is at most k .

We construct an initial fractional matching X_D with all non-zero fractions assigned to the following edges: for any vertex $u \in D$, let $\max(u) = e_0 > e_1 > e_2 > \dots > e_j$ be the edges incident to u which are in the same component as u . We set the fraction $x(e_i, e_{i+1}) = 1/3$, $i = 0, \dots, j - 1$, while the fraction x assigned to all other edges of L equals 0. It is immediate that x is a valid fractional matching since any vertex of L is fractionally matched with at most three other vertices. Every non-root edge contributes $1/3$ of its cost to the weight of this fractional matching (when the edge is fractionally matched with a bigger edge), thus making the weight of X_D equal to $(c(D) - c(\text{root}))/3$.

We now will modify the fractional matching X_D increasing its weight by $c(\text{bridge})/3$. Note that the both vertices bridge and root have degree at most 2 in X_D . Therefore, in case of root being adjacent to bridge in X_D , we can simply add $1/3$ to the fraction $x(\text{bridge}, \text{root})$ without violation (i)-(ii) obtaining claimed fractional matching. If root is not adjacent to bridge we modify the *bridge-to-root* path P in X_D : if the next after bridge edge f has a degree 2 we are done, otherwise, if f is adjacent to an edge e which is not on P we recursively increase by $x(e)$ by $1/3$. ■