LA-UR 95-1411

Conf-9506188--1

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE: **BICRITERIA NETWORK DESIGN PROBLEMS**

AUTHOR(S): M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi,
D. J. Rosenkrantz, H. B. Hunt III

SUBMITTED TO: International Coloquium on Automata Languages
(ICALP '95)
June, 1995
Szeged, Hungary



Los Alamos

Los Alamos National Laboratory
Los Alamos New Mexico 87545

# Bicriteria Network Design Problems

*M. V. Marathe* [2] *R. Ravi* [3] *R. Sundaram* [4]
*S. S. Ravi* [1] *D. J. Rosenkrantz* [1] *H.B. Hunt III* [1]

**Abstract**

We study several bicriteria network design problems phrased as follows: given an undirected graph and two minimization objectives with a budget specified on one objective, find a subgraph satisfying certain connectivity requirements that minimizes the second objective subject to the budget on the first. First, we develop a formalism for bicriteria problems and their approximations. Secondly, we use a simple parametric search technique to provide bicriteria approximation algorithms for problems with two similar criteria, where both criteria are the same measure (such as the diameter or the total cost of a tree) but differ only in the cost function under which the measure is computed. Thirdly, we present an $(O(\log n), O(\log n))$-approximation algorithm for finding a diameter-constrained minimum cost spanning tree of an undirected graph on $n$ nodes. Finally, for the class of treewidth-bounded graphs, we provide pseudopolynomial-time algorithms for a number of bicriteria problems using dynamic programming. These pseudopolynomial-time algorithms can be converted to fully polynomial-time approximation schemes using a scaling technique.

## 1 Introduction

Several fundamental problems in the design of communication networks can be modeled as finding a network obeying certain connectivity constraints. In applications that arise in several real-life situations, the goal is to minimize several measures of cost associated with the network. We first develop a formalism for bicriteria problems and their approximations. A typical bicriteria problem, $(\mathcal{A}, \mathcal{B})$, is defined by identifying two minimization objectives of interest from a set of possible objectives. The problem specifies a budget value on the first objective, $\mathcal{A}$, and seeks to find a network having minimum possible value for the second objective, $\mathcal{B}$, such that this network obeys the budget on the first objective. As an example, consider the following *diameter-bounded minimum spanning tree problem* or (Diameter, Total cost) bicriteria problem: given an undirected graph $G = (V, E)$ with two different integral nonnegative weights $f_e$ (modeling the cost) and $g_e$ (modeling the delay) for each edge $e \in E$, and an

[1]Dept. of Computer Science, University at Albany - SUNY, Albany, NY 12222. Email:{ravi, djr, hunt}@cs.albany.edu. Supported by NSF Grants CCR 94-06611 and CCR 90-06396.

[2]Los Alamos National Laboratory P.O. Box 1663, MS M986, Los Alamos NM 87545. Email: madhav@c3.lanl.gov. Research supported by the Department of Energy under Contract W-7405-ENG-36.

[3]DIMACS, Dept. of Computer Science, Princeton University, Princeton, NJ 08544-2087. Email: ravi@cs.princeton.edu. Research supported by a DIMACS postdoctoral fellowship.

[4]Dept. of Computer Science, MIT LCS, Cambridge MA 02139. Email: koods@theory.lcs.mit.edu. Research supported by DARPA contract N0014-92-J-1799 and NSF CCR 92-12184.

integral bound $B$ (on the total delay), find a minimum $f$-cost spanning tree such that the diameter of the tree under the $g$-costs (the maximum delay between any pair of nodes) is at most $B$. The following hardness results can be derived by a reduction from the partition problem [10].

**Theorem 1.1** The (Diameter, Total cost), (Diameter, Diameter) and the (Total cost, Total cost) spanning tree problems are NP-hard even for series-parallel graphs.

An $(\alpha, \beta)$-approximation algorithm is defined as a polynomial-time algorithm that produces a solution in which the first objective value is at most $\alpha$ times the budget, and the second objective value is at most $\beta$ times the minimum for any solution obeying the budget on the first objective.

# 2 Summary of results and related research

There are two natural alternative ways of formulating general bicriteria problems, one where we impose the budget on the first objective and seek to minimize the second and two, where we impose the budget on the second objective and seek to minimize the first. We show that an $(\alpha, \beta)$-approximation algorithm for one of these formulations naturally leads to a $(\beta, \alpha)$-approximation algorithm for the other. Thus our notion of bicriteria approximations is invariant on the choice of the criterion that is budgeted in the formulation.

## Arbitrary Graphs

We summarize several bicriteria spanning tree results including our new contributions in Table 1. The table contains the performance ratios for finding spanning trees under different pairs of minimization objectives. All results in the table extend to finding Steiner trees with at most a constant factor worsening in the performance ratios. We omit elaboration on these extensions. The horizontal entries denote the budgeted objective. For example the entry in row $i$, column $j$ denotes the performance guarantee for the problem of minimizing objective $j$ with a budget on the objective $i$. As a result of the equivalence mentioned earlier, the table is symmetric, i.e. entry $(i, j)$ is identical to entry $(j, i)$. For each of the problems catalogued in the table, two different costs are specified on the edges of the input undirected graph: the first objective is computed using the first cost function and the second objective, using the second cost function.

In the table, the "Diameter" objective is the maximum distance between any pair of nodes in the tree. The "Total cost" objective is the sum of the costs of all the edges in the tree. The "Degree" objective denotes the maximum degree of any node in the spanning tree; the entry (Degree, Degree) however refers to a generalization to a weighted variant based on two cost functions defined on the edges. This weighted variant of the degree objective is defined as the maximum over all nodes, of the sum of the costs of the edges incident on the node in the tree. When all edges in the graph have unit weight, this reduces to the usual notion of the maximum degree.

| Cost | Degree | Diameter | Total Cost |
|---|---|---|---|
| Degree | $(O(\lg n), O(\lg n))^*$ | $(O(\lg n), O(\lg n))$[20] | $(O(\lg n), O(\lg n))$[18] |
| Diameter | $(O(\lg n), O(\lg n))$[20] | $(1+\gamma, 1+\frac{1}{\gamma})^*$ | $(O(\lg n), O(\lg n))^*$ |
| Total Cost | $(O(\lg n), O(\lg n))$[18] | $(O(\lg n), O(\lg n))^*$ | $(1+\gamma, 1+\frac{1}{\gamma})^*$ |

**Table 1. Performance Guarantees for finding spanning trees in an arbitrary graph on $n$ nodes. Asterisks indicate results obtained in this paper. $\gamma > 0$ is a fixed accuracy parameter.**

The diagonal entries in the table follow as a corollary of the following general result proved using a parametric search algorithm.

**Theorem 2.1** Let $\mathcal{P}$ denote a single criterion minimization problem defined on a graph $G$ with costs $h$ associated with the elements of $G$ and let $\gamma > 0$ be a fixed accuracy parameter. Assume that there exists a $\rho$-approximation algorithm for $\mathcal{P}$. Then the bicriteria problem $\mathcal{P}_2$ defined on $G$ by specifying two different costs $f$ and $g$ on the elements of $G$ and the two objectives being minimizing the objective of $\mathcal{P}$ under the two different costs $f$ and $g$ has a $((1+\gamma)\rho, (1+\frac{1}{\gamma})\rho)$-approximation algorithm.

The diagonal entries in the table correspond to such bicriteria problems in which the two objectives are similar but only differ in the cost function on the edges under which they are computed; For such problems, we introduce a parametric search method on a hybrid cost function $h_e(\mu) = f_e + \mu g_e$ on the edges $e$, such that the single objective problem solved using the hybrid cost $h(\mu)$ for an appropriately chosen $\mu$ yields a good approximation for both the original objectives. For example, for the (Total cost, Total cost) problem, using Theorem 2.1 with $\rho = 1$ (using an exact algorithm to comput a MST) gives the corresponding result in our table. The result for (Diameter, Diameter) follows from known exact algorithms for minimum diameter spanning trees [7, 19]. Similarly, the result for (Degree, Degree) follows from the $O(\log n)$-approximation algorithm for the weighted degree problem in [18].

Ravi et al. [18] studied the degree-bounded minimum cost spanning tree problem. They provided an approximation algorithm with performance guarantee $(O(\log n), O(\log n))$. The problem of finding a degree-bounded minimum diameter spanning tree was studied by Ravi [20] in the context of finding good broadcast networks. He provided an approximation algorithm for the first problem with performance guarantee $(O(\log n), O(\log n))$ with an extra additive term of $O(\log^2 n)$ for the degree.

The (Diameter, Total cost) entry in Table 1 corresponds to the diameter-constrained minimum spanning tree problem introduced earlier. This problem arises naturally in the design of networks used in multicasting and multimedia applications [8, 9, 12, 14]. It is known [10] that this problem is NP-hard. In the special case when the two cost functions are identical, i.e., $f_e = g_e$ for all edges $e$, the diameter-bounded minimum spanning tree problem reduces to finding a spanning tree that has simultaneously small diameter (i.e., shallow) and small total cost (i.e., light), both under the same cost function. Awerbuch,

Baratz and Peleg [3] showed how to compute in polynomial-time such *shallow, light trees* while Khuller, Raghavachari and Young [13] studied an extension called *Light, approximate Shortest-path Trees* (LASTs). Kadaba and Jaffe [12] and Kompella et al. [14] considered the general diameter-bounded minimum spanning tree problem and presented heuristics without any guarantees. We present the first approximation algorithm for this problem; the performance ratios for both objectives are logarithmic.

## Treewidth-bounded graphs

We also study the bicriteria problems mentioned above for the class of treewidth-bounded graphs. These graphs were introduced by Robertson and Seymour [21]. Many hard problems have exact solutions when attention is restricted to the class of treewidth-bounded graphs and much work has been done in this area (see [1, 2, 5] and the references therein). Examples of treewidth-bounded graphs include trees, series-parallel graphs and bounded-bandwidth graphs. Independently, Bern, Lawler and Wong [5] introduced the notion of decomposable graphs. Later, it was shown [2] that the class of decomposable graphs and the class of treewidth-bounded graphs are one and the same. Bicriteria network design problems restricted to treewidth-bounded graphs have been previously studied in [1, 6].

We use a dynamic programming technique to show that for any class of decomposable graphs (or treewidth-bounded graphs), there are either polynomial-time (when the problem is in P) or pseudopolynomial-time algorithms (when the problem is NP-complete) for several of these bicriteria problems. We then show how to convert these pseudopolynomial-time algorithms into fully polynomial-time approximation schemes using a general scaling technique. We summarize our results for this class of graphs in Table 2. As before, the horizontal entries denote the budgeted objective.

Using our results for the (Degree, Diameter) case along with the techniques in [20], we can obtain an $O(\frac{\log n}{\log \log n})$-approximation algorithm for the minimum broadcast time problem restricted to the class of treewidth-bounded graphs (series-parallel graphs have a treewidth of 2), improving and substantially generalizing the results of Kortsarz and Peleg [16]. We omit the details due to lack of space.

| Cost Measures | Degree | Diameter | Total Cost |
|---|---|---|---|
| Degree | (open) pseudopoly | (open) pseudopoly | poly-time |
| Diameter | (open) pseudopoly | (weak NP-hard) pseudopoly | (weak NP-hard) pseudopoly |
| Total Cost | poly-time | (weak NP-hard) pseudopoly | (weak NP-hard) pseudopoly |

**Table 2. Bicriteria spanning tree results for treewidth-bounded graphs**

Due to lack of space, the rest of the paper consists of selected proof sketches.

# 3 Equivalence of bicriteria formulations

We formulate a general bicriteria problem in network design as follows: Given a graph $G$ and two integral cost functions, say $c$ and $d$, defined on a class $\mathcal{S}$ of subgraphs of $G$ (e.g., spanning trees of $G$), and a bound on the value of one of the costs (say $C$ for the $c$-cost), find a subgraph in $\mathcal{S}$ that has cost at most $C$ under the cost function $c$ and the minimum possible cost under $d$ given this restriction on the $c$-cost.[5] We call this the $C$-bounded minimum-$d$-cost subgraph problem. The alternative formulation would be to use a bound $D$ on the $d$-cost of the solution and ask for a minimum-$c$-cost subgraph under this restriction. This alternative formulation may be termed the $D$-bounded minimum-$c$-cost subgraph problem.

Note that such bicriteria problems are meaningful only when the two criteria are "hostile" with respect to each other in that the objective of minimizing one is incompatible with that of minimizing the other. A good example of such hostile objectives are the degree and the total edge cost of a spanning tree in an unweighted graph [18]. The notion of hostility between criteria can be formalized by defining two minimization criteria to be hostile whenever the minimum value of one of the objectives is monotonically nondecreasing as the budget on the value of the other objective is decreased.

**Theorem 3.1** The existence of a $(\rho_c, \rho_d)$-approximation algorithm for the $C$-bounded minimum-$d$-cost subgraph problem implies the existence of a $(\rho_d, \rho_c)$-approximation algorithm for the $D$-bounded minimum-$c$-cost subgraph problem.

The proof of the theorem uses binary search on the range of values of the $c$-cost with an application of the given approximation algorithm at each step of this search.

# 4 Parametric search for approximations of similar objective functions

We now present the approximation algorithm used to prove Theorem 2.1. We illustrate the proof of the theorem by considering the case (Total cost, Total cost) in Table 1. In this problem, we are given two costs $f_e$ and $g_e$ on the edges $e \in E$ of the input graph $G = (V, E)$. We are also given a budget $B$ on the total cost of the spanning tree under $g$. Assume for now that the magnitude of costs $f$ on the edges are polynomial in the size of the input graph.

Let OPT denote the minimum cost of the tree under $f$ which obeys the restriction that its cost under $g$ is at most $B$. For any tree $T$ and cost function $f$, we use $COST_f(T)$ to denote the cost of $T$ under $f$. To simplify the analysis, we assume that $\gamma$ divides $OPT$. This can be enforced by scaling both the cost functions $f$ and $g$ by $\gamma$.

[5] We use the term "cost under $c$" or "$c$-cost" in this section to mean value of the objective function computed using $c$, and not to mean the total of all the $c$ costs in the network.

**Algorithm Two-Cost**$(G, f, g, B, \gamma)$

**Input**: A graph $G(V, E)$ with two cost functions $f$ and $g$ on the edges, a budget $B$ on the cost of the spanning tree under $g$ and a performance requirement $\gamma > 0$.
**Output**: A spanning tree $T$ of $G$ such that the cost of $T$ under $g$ is no more than $(1 + \frac{1}{\gamma})B$ and the cost of $T$ under $f$ is no more than $(1 + \gamma)OPT$.

1 Initialize $C := 1$
2 **While** (Test$(C)$ = NO) **do**
3 $C := C + 1$
4 Output the tree $T$ computed by **Test** as the solution.

**Procedure Test**$(C)$

1 $\mu := \frac{C}{B}$.
2 Compute a new cost function $h$ on the edges $e \in E$ as follows: $h(e) := f(e) + \mu g(e)$.
3 Compute a minimum spanning tree $T$ in the graph $G(V, E)$ under the cost function $h$.
4 **If** $COST_h(T) \leq (1 + \gamma)C$ **then** output YES **else** output NO.

**Lemma 4.1** The function $\mathcal{R}(C) = \frac{COST_h(MST)}{C}$ as $C$ takes increasing integral values from $1, 2, 3, \ldots$ is monotone nonincreasing.

*Proof*: Suppose for a contradiction that for two integral values of $C$, say $C_1$ and $C_2$ with $C_1 < C_2$, we have that $\mathcal{R}(C_1) < \mathcal{R}(C_2)$. Let $T_1$ and $T_2$ denote minimum spanning trees of $G$ under $h$ when $C = C_1$ and $C = C_2$ respectively. For $i \in \{1, 2\}$, let $F_i$ and $G_i$ denote the costs of the tree $T_i$ under $f$ and $g$ respectively. Thus, we have that $\mathcal{R}(C_i) = \frac{F_i}{C_i} + \frac{G_i}{B}$ for $i \in \{1, 2\}$.

Consider the cost under $h$ of the spanning tree $T_1$ when $C = C_2$. By the definition of $F_1$ and $G_1$, it follows that the cost of $T_1$ is $F_1 + \frac{G_1 \cdot C_2}{B}$. Thus the value of $\mathcal{R}(C_2)$ is at most this cost divided by $C_2$ which is $\frac{F_1}{C_2} + \frac{G_1}{B}$. This in turn is less than $\frac{F_1}{C_1} + \frac{G_1}{B}$, since $C_1 < C_2$. But $\frac{F_1}{C_1} + \frac{G_1}{B}$ is exactly $\mathcal{R}(C_1)$ contradicting the assumption that $\mathcal{R}(C_1) < \mathcal{R}(C_2)$. □

**Theorem 4.2** Let $C'$ be the value of $C$ when the Test procedure outputs YES. Let $T_{C'}$ denote the corresponding solution tree. Then $COST_f(T_{C'}) \leq (1 + \frac{1}{\gamma})OPT$ and $COST_g(T_{C'}) \leq (1 + \gamma)B$.

*Proof*: First consider the value of $COST_h(T)$ when $C = C^* = \frac{OPT}{\gamma}$ (Note that $C^*$ is integral by assumption). This is at most $OPT + \frac{C^* B}{B} = OPT + C^* = (1+\gamma)C^*$. Thus the value of $\mathcal{R}(C^*) = \frac{COST_h(MST)}{C^*} \leq 1+\gamma$. Since $\mathcal{R}(C^*) \leq 1+\gamma$, the function $\mathcal{R}(C)$ is monotone nonincreasing by Lemma 4.1, and $C'$ is the least integer such that $\mathcal{R}(C') \leq 1 + \gamma$, we have that $C' \leq C^*$. It is now easy to verify that the following inequalities hold for $COST_f(T_{C'})$ and $COST_g(T_{C'})$.
$COST_f(T_{C'}) \leq COST_h(T_{C'}) \leq OPT + \frac{C'}{B}B \leq OPT + C^* \leq (1 + \frac{1}{\gamma})OPT$.

$$\frac{C'}{B} COST_g(T_{C'}) \leq COST_h(T_{C'}) \leq C'(1+\gamma).$$

The second chain of inequalities implies that $COST_g(T_{C'}) \leq (1+\gamma)B$.

□

We can obtain a better running time by doing a binary search for $C$ in **Algorithm Two-cost** thus using only $O(\log F)$ calls to the polynomial-time test procedure **Test**, where $F$ denotes the ratio of maximum edge cost to the minimum edge cost under $f$. We omit the details.

# 5 Diameter-bounded minimum spanning trees

We now discuss our approximation algorithm for the diameter bounded minmum cost spanning tree problem with performance guarantee $(O(\log n), (\log n))$. The proof for diameter bounded minimum cost steiner tree is similar and is omitted. By an approximation preserving reduction from set cover and using the known hardness results in [4, 17], we can show that there is no polynomial-time approximation algorithm that outputs a Steiner tree of diameter at most the bound $D$, and cost at most $R$ times that of the minimum cost diameter-$D$ Steiner tree, for $R < \log k/8$, unless $NP \subseteq DTIME(n^{\log \log n})$.

We shall use the term "diameter cost" of a tree to mean the diameter of the tree under the $d$-costs, and the "building cost" of a tree to mean the total cost of all the edges in the tree computed using the $c$-costs. We shall also use the term "diameter-$D$ path (tree)" to refer to a path (tree) of diameter cost at most $D$ under $d$.

We now describe some background material that will be useful in understanding our algorithm and its analysis. Given a diameter bound $D$, the problem of finding a diameter-$D$ path between two specified vertices of minimum building cost has been termed the multi-objective shortest path problem (MOSP). This problem is NP-complete and Warburton [22] presented the first fully polynomial approximation scheme (FPAS) for this problem. Hassin [11] provided an alternative FPAS for the problem without a running-time dependency on the magnitude of the costs in the problem. His algorithm runs in time $O(m(\frac{n^2}{\epsilon} \log \frac{n}{\epsilon}))$, where $m$ and $n$ denote the number of edges and nodes in the input graph respectively. We use the latter result in implementing our algorithm.

Next, we state a tree decomposition result from [18], and use it in the proof of the performance guarantee.

**Lemma 5.1** Let $T$ be a tree with an even number of marked nodes. Then there is a pairing $(v_1, w_1), \ldots, (v_k, w_k)$ of the marked nodes such that the $v_i - w_i$ paths in $T$ are edge-disjoint.

A pairing of the marked nodes that minimizes the sum of the sizes of the tree-paths between the nodes paired up can be shown to obey the property in the claim above.

### Overview

The algorithm begins with an empty solution subgraph where each node is in a connected component (termed a *cluster*) by itself in the solution. Assume for simplicity that the number of nodes in the input graph is a power of two. The algorithm works in $\log_2 n$ iterations where $n$ is the number of nodes in the original graph, merging clusters in pairs during each iteration by adding edges between them. This pairing ensures that the number of iterations is as desired.

The clusters maintained by our algorithm represent node-subsets of the input graph $G$. However, they *do not* represent a partition of the nodes of the input graph. This is because of the way in which we merge the clusters in the algorithm. For each cluster we maintain a spanning tree on the nodes in the cluster. The spanning trees of the clusters maintained by the algorithm are not necessarily edge-disjoint as a result of our merging procedure. We sketch this procedure below. We identify a *center* in the spanning tree of each cluster. In each iteration, every cluster is paired with another cluster and merged with it by the addition of a path between their respective centers. This path may involve nodes that occur in either of the merging clusters or even nodes in other clusters currently maintained by the algorithm. However, while merging two clusters into one, we ensure that the new cluster formed has at most one copy of any node or edge.

### The Algorithm

1 Initialize the set of clusters $\mathcal{C}$ to contain $n$ singleton sets, one for each node of the input graph. For each cluster in $\mathcal{C}$, define the single node in the cluster to be the center for the cluster. Initialize the iteration count $i := 1$.

2 Repeat until there remains a single cluster in $\mathcal{C}$

3 Let the set of clusters $\mathcal{C} = \{C_1 \ldots, C_{\frac{n}{2^i-1}}\}$.

4 Construct a complete graph $G_i$ as follows.

5 The node set $V_i$ of $G_i$ is $\{v : v$ is the center of a cluster in $\mathcal{C}\}$.

6 Between every pair of nodes $v_x$ and $v_y$ in $V_i$, include an edge $(v_x, v_y)$ in $G_i$ of cost equal to a $(1+\epsilon)$-approximation of the shortest building cost of a diameter-$D$ path between $v_x$ and $v_y$ in $G$, where $\epsilon$ is the accuracy parameter input to the algorithm. Since a FPAS is available to compute such an estimate [11], the costs of all the edges in $G_i$ can be computed in polynomial-time.

7 Find a minimum-cost perfect matching in $G_i$.

8 For each edge $e = (v_r, v_s)$ in the matching

9 Let $P_{rs}$ be the path in $G$ represented by $e = (v_r, v_s)$. Add this path to merge the clusters $C_r$ and $C_s$ for which $v_r$ and $v_s$ were centers respectively, to form a new cluster $C_{rs}$, say. The node set of the cluster $C_{rs}$ is defined as the union of the node sets $C_r, C_s$ and the nodes in $P_{rs}$.

10 Define the union of the edge sets of the spanning trees for $C_r$ and $C_s$ and the set of edges in $P_{rs}$ to be $E_{rs}$. The edges in $E_{rs}$ form a connected

graph on $C_{rs}$. Choose one of $v_r$ or $v_s$ as the center $v_{rs}$ of the cluster $C_{rs}$. Using only the diameter cost function $d$, find a shortest-path tree rooted at $v_{rs}$ in the graph $(C_{rs}, E_{rs})$. This is the spanning tree for the cluster $C_{rs}$.

11 Set $\mathcal{C} := \mathcal{C} - \{C_r, C_s\} \cup \{C_{rs}\}$.

12 $i := i + 1$.

13 Output the spanning tree of the single cluster in $\mathcal{C}$.

We prove the performance guarantee using a series of lemmas.

At each iteration, since the clusters are paired up using a perfect matching and merged to form new clusters, the number of clusters halves. Thus we have the following lemma.

**Lemma 5.2** The total number of iterations of the above algorithm is $\lceil \log_2 n \rceil$.

**Lemma 5.3** Let $C$ be a cluster with center $v$ formed at iteration $i$ of the algorithm. Then any node $u$ in $C$ has a diameter-$iD$ path to $v$ in the spanning tree of $C$ maintained by the algorithm.

*Proof*: The proof is by induction on the iteration count $i$. The basis when $i = 1$ is trivial. To prove the induction step, consider a cluster $C_{rs}$ formed at iteration $i$ ($> 1$) by merging two clusters $C_r$ and $C_s$ with centers $v_r$ and $v_s$ respectively. Suppose $v_{rs} = v_r$. Consider a node $u \in C_{rs}$. $u$ is in either $C_r$,$C_s$ or the path $P_{rs}$. In all these cases, using the inductive hypothesis and the fact that the diameter cost of path $P_{rs}$ is at most $D$, it is easy to show that $u$ has a diameter-$iD$ path to the center $v_{rs} = v_r$ in the graph $(C_{rs}, E_{rs})$. Since we compute a shortest-path tree in this graph rooted at $v_{rs}$ using the diameter costs, it follows that the path in this tree between any node and $v_{rs}$ has diameter cost no more than $iD$. □

**Corollary 5.4** Let $C$ be a cluster formed at iteration $i$ of the algorithm. Then the diameter cost of the spanning tree of $C$ maintained by the algorithm is at most $2iD$.

**Lemma 5.5** Let $OPT_D$ be the minimum building cost of any diameter-$D$ spanning tree of the input graph. At each iteration $i$ of the algorithm, the cost of the minimum cost matching in $G_i$ found in Step 7 is at most $(1 + \epsilon) \cdot OPT_D$, where $\epsilon > 0$ is the accuracy parameter input to the algorithm.

*Proof*: It is easy to show using a simple induction on the iteration count that, at any iteration $i$, the set of centers of clusters for this iteration are distinct nodes of $G$. Since these are exactly the nodes in $G_i$, we have that the graph $G_i$ has at most one copy of any node of $G$. We can now apply Lemma 5.1 on the optimal diameter-$D$ spanning tree of the input graph with the nodes of $G_i$ marked. The lemma yields a pairing between these centers such that the pairs are connected using edge-disjoint paths in the optimal tree. Note that all these paths have diameter cost at most $D$ since they are derived from a diameter-$D$ tree. Furthermore, the sum of the building costs of all these paths is at most

$OPT_D$ since they form edge-disjoint fragments of a tree of total building cost $OPT_D$. Thus we have identified a pairing between the nodes in $G_i$ of "cost" at most $OPT_D$ where the cost of a pair is the minimum building cost of a diameter-$D$ path between its endpoints.

In constructing $G_i$, the cost assigned to an edge between a pair of nodes is a $(1+\epsilon)$-approximation to the minimum building cost of a diameter-$D$ path between these nodes in $G$ (see Step 6). Thus between every pair identified above, there is an edge in $G_i$ of cost at most $(1+\epsilon)$ times the building cost of the path between them in the optimal tree. This identifies a perfect matching in $G_i$ of cost at most $(1+\epsilon) \cdot OPT_D$ and completes the proof. □

Note that the spanning tree finally output by the above algorithm is a subgraph of the union of all the paths added by all the matchings over all the iterations. Lemma 5.2, corollary 5.4, and lemma 5.5 prove the performance guarantees of the algorithm.

# 6 Treewidth-bounded Graphs

In this section we briefly discuss our ideas by describing the algorithm for solving the diameter $B$-bounded minimum cost spanning tree problem.

Let $f$ be the cost function on the edges for the first objective (diameter) and $g$, the cost function for the second objective (total cost). Let $\Gamma$ be any class of decomposable graphs. Let the maximum number of terminals associated with any graph $G$ in $\Gamma$ be $k$. Following [5], it is assumed that a given graph $G$ is accompanied by a parse tree specifying how $G$ is constructed using the rules and that the size of the parse tree is linear in the number of nodes.

Let $\pi$ be a partition of the terminals of $G$. For every terminal $i$ let $d_i$ be a number in $\{1 \ldots B\}$. For every pair of terminals $i$ and $j$ in the same block of the partition $\pi$ let $d_{ij}$ be a number in $\{1 \ldots B\}$. Corresponding to every partition $\pi$, set $\{d_i\}$ and set $\{d_{ij}\}$ we associate a cost for $G$,
$Cost^{\pi}_{\{d_i\},\{d_{ij}\}}$ = Minimum total cost under the $g$ function of any forest containing a tree for each block of $\pi$, such that the terminal nodes occurring in each tree are exactly the members of the corresponding block of $\pi$, no pair of trees is connected, every vertex in $G$ appears in exactly one tree, $d_i$ is an upper bound on the maximum distance (under the $f$ function) from $i$ to any vertex in the same tree and $d_{ij}$ is an upper bound the distance (under the $f$ function) between terminals $i$ and $j$ in their tree. For the above defined cost, if there is no forest satisfying the required conditions the value of Cost is defined to be $\infty$.

Note that the number of cost values associated with any graph in $\Gamma$ is $O(k^k \cdot B^{O(k^2)})$. We now show how the cost values can be computed in a bottom-up manner given the parse tree for $G$. To begin with, since $\Gamma$ is fixed, the number of primitive graphs is finite. For a primitive graph, each cost value can be computed in constant time, since the number of forests to be examined is fixed. Now consider computing the cost values for a graph $G$ constructed from subgraphs $G_1$ and $G_2$, where the cost values for $G_1$ and $G_2$ have already been computed. Notice that any forest realizing a particular cost value for $G$

decomposes into two forests, one for $G_1$ and one for $G_2$ with some cost values. Since we have maintained the best cost values for all possibilities for $G_1$ and $G_2$, we can reconstruct for each partition of the terminals of $G$ the forest that has minimum cost value among all the forests for this partition obeying the diameter constraints. We can do this in time independent of the sizes of $G_1$ and $G_2$ because they interact only at the terminals to form $G$, and we have maintained all relevant information.

Hence we can generate all possible cost values for $G$ by considering combinations of all relevant pairs of cost values for $G_1$ and $G_2$. This takes time $O(k^4)$ per combination for a total time of $O(k^{2k+4} \cdot B^{O(k^4)})$. As in [5], we assume that the size of the given parse tree for $G$ is $O(n)$. Then the dynamic programming algorithm takes time $O(n \cdot k^{2k+4} \cdot B^{O(k^4)})$.

**Acknowledgements** We thank Professors S. Arnborg and H. L. Bodlaender for pointing out to us the equivalence between treewidth bounded graphs and decomposable graphs. We wish to thank A. Ramesh for bringing [15] to our attention. We also thank Dr.Vachaspati Kompella for making his other papers available to us. Finally, we thank the referees for their constructive suggestions.

## References

[1] S. Arnborg, J. Lagergren and D. Seese, "Easy Problems for Tree-Decomposable Graphs," *J. Algorithms*, Vol. 12, 1991, pp. 308-340.

[2] S. Arnborg, B. Courcelle, A. Proskurowski and D. Seese, "An Algebraic Theory of Graph Problems," *J. ACM*, Vol. 12, 1993, pp. 308-340.

[3] B. Awerbuch, A. Baratz, and D. Peleg, "Cost-sensitive analysis of communication protocols," *Proc. of 9th Symp. on Principles of Distributed Computing (PODC)*, pp. 177-187, (1990).

[4] M. Bellare, S. Goldwasser, C. Lund, and A. Russell, "Efficient probabilistically checkable proofs," *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing* (1993), pp. 294-304.

[5] M.W. Bern, E.L. Lawler and A.L. Wong, "Linear -Time Computation of Optimal Subgraphs of Decomposable Graphs," *J. Algorithms*, Vol. 8, 1987, pp. 216-235.

[6] H.L. Bodlaender, "Dynamic programming on graphs of bounded treewidth," *Proc. of the 15th ICALP*, LNCS Vol. 317, 1988, pp. 105-118.

[7] P. M. Camerini, and G. Galbiati, "The bounded path problem," *SIAM J. Alg. Disc., Meth.* Vol. 3, No. 4 (1982), pp. 474-484.

[8] C.-H. Chow, "On multicast path finding algorithms," *Proc. of IEEE INFOCOM '91*, pp. 1274-1283 (1991).

[9] A. Frank, L. Wittie, and A. Bernstein, "Multicast communication in network computers," *IEEE Software*, Vol. 2, No. 3, pp. 49-61 (1985).

[10] M. R. Garey and D. S. Johnson, *Computers and Intractability: A guide to the theory of NP-completeness*, W. H. Freeman, San Francisco (1979).

[11] R. Hassin, "Approximation schemes for the restricted shortest path problem," *Math. of O. R.*, Vol. 17, No. 1, pp. 36-42 (1992).

[12] B. Kadaba and J. Jaffe, "Routing to multiple destinations in computer networks," *IEEE Trans. on Comm.*, Vol. COM-31, pp. 343-351, (Mar. 1983).

[13] S. Khuller, B. Raghavachari, and N. Young, "Balancing Minimum Spanning and Shortest Path Trees," *Proc., Fourth Annual ACM-SIAM Symposium on Discrete Algorithms* (1993), pp. 243-250.

[14] V.P. Kompella, J.C. Pasquale and G.C. Polyzos, "Multicasting for multimedia applications," *Proc. of IEEE INFOCOM '92*, (May 1992).

[15] V.P. Kompella, J.C. Pasquale and G.C. Polyzos, "Multicast Routing for Multimedia Communication," *IEEE/ACM Transactions on Networking*, pp. 286-292, (1993).

[16] G. Kortsarz and D. Peleg, "Approximation algorithms for minimum time broadcast," *Proc. of the 1992 Israel Symposium on Theoretical Computer Science* LNCS 601, (1994).

[17] C. Lund and M. Yannakakis, "On the Hardness of Approximating Minimization Problems," *Proc., 25th Annual ACM Symp. on Theory of Computing*, (1993), pp. 286-293.

[18] R. Ravi, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and H.B. Hunt III, "Many birds with one stone: Multi-objective approximation algorithms," *Proc. of the 25th Annual ACM Symposium on the Theory of Computing* (1993), pp. 438-447.

[19] R. Ravi, R. Sundaram, M. V. Marathe, D. J. Rosenkrantz, and S. S. Ravi, "Spanning trees short or small," in *Proc. of the 5th Annual ACM-SIAM Symposium on Discrete Algorithms,* (1994), pp 546-555.

[20] R. Ravi, "Rapid Rumor Ramification: Approximating the minimum broadcast time," in *Proc. of the 25th Annual IEEE Foundations of Computer Science* (1994), pp. 202-213.

[21] N. Robertson and P. Seymour, "Graph Minors IV, Tree-width and well-quasi-ordering," *J. Combin. Theory Ser. B* 48, 227-254 (1990).

[22] A. Warburton, "Approximation of Pareto optima in multiple-objective, shortest path problems," *Oper. Res.* 35, pp. 70-79 (1987).