SIAM J. OPTIM.
Vol. 10, No. 1, pp. 22–42

COMPUTING THE MINIMUM COST PIPE NETWORK INTERCONNECTING ONE SINK AND MANY SOURCES*

GUOLIANG XUE†, THEODORE P. LILLYS‡, AND DAVID E. DOUGHERTY§

Abstract. In this paper, we study the problem of computing the minimum cost pipe network interconnecting a given set of wells and a treatment site, where each well has a given capacity and the treatment site has a capacity that is no less than the sum of all the capacities of the wells. This is a generalized Steiner minimum tree problem which has applications in communication networks and in groundwater treatment. We prove that there exists a minimum cost pipe network that is the minimum cost network under a full Steiner topology. For each given full Steiner topology, we can compute all the edge weights in linear time. A powerful interior-point algorithm is then used to find the minimum cost network under this given topology. We also prove a lower bound theorem which enables pruning in a backtrack method that partially enumerates the full Steiner topologies in search for a minimum cost pipe network. A heuristic ordering algorithm is proposed to enhance the performance of the backtrack algorithm. We then define the notion of k-optimality and present an efficient (polynomial time) algorithm for checking 5-optimality. We present a 5-optimal heuristic algorithm for computing good solutions when the problem size is too large for the exact algorithm. Computational results are presented.

Key words. minimum cost pipe network, generalized Steiner minimum tree problem, bounding theorem, backtrack, interior-point methods, k-optimal

AMS subject classifications. 68Q20, 68Q25, 90B10, 90B12

PII. S1052623496313684

1. Introduction.

In the Euclidean Steiner minimum tree (ESMT) problem [21, 13], we are seeking a minimum cost network interconnecting a set of given points on the Euclidean plane, where the cost of a network is measured as the sum of edge lengths. Note that additional points joining the line segments may be added in order to reduce network cost. These added points are called *Steiner points* and the given points are called *regular points*. There is a large literature on the ESMT problem [6, 7, 31]. We refer readers to the survey paper [16] and the book [17].

In this paper, we study a generalization of the ESMT problem, which arises from communication networks [13] and groundwater treatment in civil and environmental engineering. Here we are also given a set of points on the Euclidean plane and are seeking the minimum cost network interconnecting these given points. In this problem, the cost of the network is the sum of the costs of all the edges in the network, but the cost of an edge is defined differently. One of the given points is a *sink* (or *service center*) and the rest of the given points are *sources* (or *clients*). For each source, there is a given positive capacity. The sink has a capacity that is no less than the sum of the capacities of the sources. For a given network, there is a flow on each edge, and the cost of that edge is the product of the length of the edge with the *cost per*

*Received by the editors December 11, 1996; accepted for publication (in revised form) September 22, 1998; published electronically October 20, 1999.
http://www.siam.org/journals/siopt/10-1/31368.html

†Department of Computer Science, University of Vermont, Burlington, VT 05405 (xue@cs.uvm.edu). This research was supported in part by U.S. Army grant DAAH04-96-1-0233 and by NSF grants ASC-9409285 and OSR-9350540.

‡Department of Civil and Environmental Engineering, University of Vermont, Burlington, VT 05405. Current address: HydroGeoLogic, Inc., 1155 Herndon Parkway, Suite 900, Herndon, VA 20170 (tpl@hgl.com). This research was supported in part by NSF grant OSR-9350540.

§Department of Civil and Environmental Engineering, University of Vermont, Burlington, VT 05405 (ddougher@emba.uvm.edu). This research was supported in part by NSF grant OSR-9350540.

unit length (CPUL) of that edge, which is a function of the flow of that edge. For example, a larger diameter pipe is needed for an edge with a bigger flow and a smaller diameter pipe is needed for an edge with a smaller flow. In most cases, the CPUL is a monotonically nondecreasing function of the flow, which is zero for zero flow and positive for positive flows. Related problems have been studied [3, 19, 36], where algorithms for computing a suboptimal solution were proposed. In this paper, we present exact and heuristic algorithms for solving this generalized Steiner minimum tree problem. We will use groundwater treatment as a motivating example. For applications in communication networks, we refer the readers to Gilbert [13].

The rest of this paper is organized as follows. In section 2, we present the physical problem and define the mathematical model. The notion of *topology* is also introduced in that section. In section 3, we present the notion of full Steiner topology and prove that there exists a minimum cost pipe network, which is the minimum cost network under a full Steiner topology. In section 4, we show that for a given full Steiner topology, we can compute the CPUL for all the edges in the topology in linear time. Then the minimum cost network under the given topology is solved using a powerful interior-point method [34]. In section 5, we prove a bounding theorem which enables the application of a backtrack algorithm that partially enumerates the full Steiner topologies in search of the minimum cost pipe network. A *max-min* heuristic ordering algorithm is proposed to enhance the performance of the backtrack algorithm. In section 6, we define the notion of k-optimality and present an efficient algorithm for checking 5-optimality. When a network is found to be not 5-optimal, we can easily change the topology to one with a lower cost. This leads to a heuristic algorithm for computing 5-optimal networks when the problem size is too large for the backtrack algorithm. We present some preliminary computational results in section 7 and conclude the paper in section 8.

2. The physical problem and its mathematical model. To treat contaminated groundwater in a given region, several wells are constructed in that region and contaminated water from within these wells is piped to a treatment site. Each well has a known location and a known flow-rate (or capacity). The treatment site has a known location and a known capacity, which is no less than the sum of the capacities of all wells. We need to build a pipe network to transport water from the wells to the treatment site. Given the type of materials to be used, it is desirable to build a pipe network that has the minimum cost. In contrast to many water supply problems, service reliability is not a significant requirement here, so no redundant arcs are needed in this problem. Depending on different flow-rates through different edges of the network, we need to use pipes of different sizes for edges with different flow-rates. For a given type of material, the CPUL of the pipe is a function of the flow-rate of that pipe. In many situations, this function, denoted by $f(\bullet)$, is *monotonically nondecreasing* with the flow-rate and is also *concave* (i.e., $f(\frac{x+y}{2}) \geq \frac{1}{2}(f(x) + f(y))$ for any x, y in the domain of the function [28]). In addition, $f(0) = 0$ and $f(x) > 0$ for any $x \in (0, \infty)$. We will assume this property for $f(\bullet)$ throughout this paper. The cost of an edge in the network is the product of the length of the edge with the CPUL of that edge. The cost of the network is the sum of the costs of all the edges. We are interested in computing a minimum cost pipe network.

In the following, we will present a precise mathematical model for this problem. Let $n \geq 3$ be a given integer. Let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of n points on the Euclidean plane R^2, where p_1 represents the location of the treatment site and $p_2, p_3, \ldots, p_n$ represent the locations of the $n-1$ wells, respectively. Let $c(p_1)$, $c(p_2)$,

$\ldots, c(p_n)$ be n positive numbers such that $c(p_1) \geq \sum_{k=2}^{n} c(p_k)$. One may think of $c(p_1)$ as the capacity of the treatment site and of $c(p_2), c(p_3), \ldots, c(p_n)$ as the capacities of the $n-1$ wells, respectively.

Since we need to transport water from all the wells to the treatment site, the pipe network will form a *connected graph* $G = (V, E)$, where E is the set of edges that correspond to the pipes and $V = \{v_1, \ldots, v_n, v_{n+1}, \ldots, v_{n+m}\}$ is the set of vertices such that $v_1, v_2, \ldots, v_n$ correspond to the points $p_1, p_2, \ldots, p_n$, respectively. The m additional vertices $v_{n+1}, \ldots, v_{n+m}$ correspond to some possible *Steiner points* in the pipe network where two or more pipes meet. For the moment, we assume that m can be any nonnegative integer. In section 3, we will prove that m can be restricted to nonnegative integers less than or equal to $n-2$. The previous discussion leads to the following definition.

DEFINITION 2.1. *A* topology $T(P)$ *for a given point set* $P = \{p_1, p_2, \ldots, p_n\}$ *is an undirected connected graph* $G = (V, E)$*, where* E *is the set of edges and* $V = \{v_1, \ldots, v_n, v_{n+1}, \ldots, v_{n+m}\}$ *is the set of vertices such that* $v_1, v_2, \ldots, v_n$ *correspond to the points* $p_1, p_2, \ldots, p_n$*, respectively, and* $v_{n+1}, \ldots, v_{n+m}$ *correspond to* m *Steiner points. A* realization $R(T, P, S, A, X)$ *of* $T(P)$ *is given by the tuple* (S, A, X)*, where* S *is a set of points* $\{p_{n+1}, \ldots, p_{n+m}\} \in R^2$*,* A *is a set of* arcs *that is a subset of* $\{(i,j), (j,i) | \{i,j\} \in E\}$*, and* X *is a set of* flows $\{x(i,j) \geq 0 | (i,j) \in A\}$ *satisfying the following conditions:*

$$\sum_{(i,1)\in A} x(i,1) - \sum_{(1,j)\in A} x(1,j) \leq c(p_1), \tag{2.1}$$

$$\sum_{(k,j)\in A} x(k,j) - \sum_{(i,k)\in A} x(i,k) = c(p_k), \quad k = 2, 3, \ldots, n, \tag{2.2}$$

$$\sum_{(k,j)\in A} x(k,j) - \sum_{(i,k)\in A} x(i,k) = 0, \quad k = n+1, n+2, \ldots, n+m. \tag{2.3}$$

The cost of realization R *is given by*

$$F(R) = \sum_{(i,j)\in A} f(x(i,j)) ||p_i - p_j||, \tag{2.4}$$

where $|| \bullet ||$ *stands for the Euclidean norm. The* cost of topology T *is given by*

$$F(T) = \min\{F(R) | R \text{ is realization of } T\}. \tag{2.5}$$

A realization of a topology for P *is called a* pipe network *interconnecting* P*.*

Note that in the above definition, the number of vertices in a topology for point set P may be any integer greater than or equal to $|P|$. We assume implicitly that the vertices $v_1, v_2, \ldots, v_n$ correspond to the points $p_1, p_2, \ldots, p_n$. In a realization, the vertices $v_1, \ldots, v_n$ are always fixed at points $p_1, \ldots, p_n$ and the vertices $v_{n+1}, \ldots, v_{n+m}$ are fixed at some points $p_{n+1}, \ldots, p_{n+m}$. A realization also specifies the way the water is transported to the treatment site. Condition (2.1) says that the net flow into the treatment site is at most $c(p_1)$. Condition (2.2) says that the net flow out of the kth well is $c(p_k)$. Condition (2.3) says that the flow into any Steiner point equals the flow out of that Steiner point. For any two vertices i and j, we use $\{i, j\}$ to represent the (undirected) edge interconnecting i and j and use (i, j) to represent the (directed) arc from i to j. For graph notations not defined in this paper, we refer readers to [15].

It is clear that, given a realization R of a topology for a point set, we can obtain another realization R_1 of the same topology for the point set by deleting all the zero flows and arcs with zero flows in R. Realization R_1 has the property that every arc

has a positive flow and that the cost of R_1 is less than or equal to that of R. This observation will help us in reducing the number of topologies to be considered.

Now we can state the minimum cost pipe network problem formally as follows.

PROBLEM 2.1. *Given the point set P containing $p_1, p_2, \ldots, p_n \in R^2$, the positive constants $c(p_1), c(p_2), \ldots, c(p_n)$ where $c(p_1) \geq \sum_{k=2}^{n} c(p_k)$, and the function $f(\bullet)$ which is concave, monotonically nondecreasing such that $f(0) = 0$ and $f(x) > 0$ for any $x \in (0, \infty)$, compute a minimum cost pipe network interconnecting P.*

Note that Problem 2.1 becomes the ESMT problem [21] when the cost function $f(\bullet)$ always equals 1. Since the ESMT problem is NP-hard [12], polynomial time algorithms for solving Problem 2.1 do not seem to exist.

3. Full Steiner topologies.

DEFINITION 3.1. *Given a point set P containing n points $\{p_1, \ldots, p_n\}$ on the Euclidean plane, a* Steiner topology *for P is a topology for P that is a tree graph such that every vertex corresponding to a Steiner point has degree* 3.

We will prove that there is a minimum cost pipe network that is a realization of a Steiner topology. The following notation is needed in the proof.

DEFINITION 3.2. *Given a path in a pipe network, the flow of this path is the minimum of the flows on the arcs of the path.*

THEOREM 3.1. *For any given topology $G = (V, E)$ for P, there exists a Steiner topology T for P such that*

1. *T is a subgraph of G;*
2. *The cost of T is no greater than the cost of G.*

Proof. Let R be a minimum cost realization of G. As discussed in section 2, we may assume that every arc of R has a positive flow, without loss of generality. For any edge $\{i, j\}$ of G, at most one of (i, j) and (j, i) may be an arc of R, because otherwise we can reduce the network cost by reducing the flows on (i, j) and (j, i).

Let E' be the set of all edges $\{i, j\}$ such that either (i, j) or (j, i) is an arc in R. Let $G' = (V', E')$ be the subgraph of G induced by E'. It is clear that G' is also a topology for P and that R is a realization of both G and G'. If the arcs of R constitute a directed tree (with v_1 as the unique sink), then G' is a Steiner topology for P and the cost of G' is no greater than the cost of R (which is the cost of G) and the theorem is proved.

In the rest, we will prove that there exists a minimum cost realization of G whose arcs constitute a directed tree (with v_1 as the unique sink).

It follows from the definition of a realization that there is at least one directed path from u to v_1 for any vertex $v_1 \neq u$ of R. If the u–v_1 path is unique for any $u \neq v_1$, the arcs of R constitute a directed tree (with v_1 as the unique sink) and there is nothing that needs to be proved.

Suppose that for some k, there are two paths from v_k to v_1 with positive flows. Let the two paths be

$$\pi_1 = (v_{i_0}, v_{i_1}, v_{i_2}, \ldots, v_{i_{m_1}}) \quad \text{and} \quad \pi_2 = (v_{j_0}, v_{j_1}, v_{j_2}, \ldots, v_{j_{m_2}}),$$

where $i_0 = j_0 = k$ and $i_{m_1} = j_{m_2} = 1$. Let the flow of π_1 be r_1 and the flow of π_2 be r_2. Let $r = \min\{r_1, r_2\}$. Since both r_1 and r_2 are positive, r is also positive. Let the flow on arc $(v_{i_{t-1}}, v_{i_t})$ be $\alpha_t + r$ $(t = 1, 2, \ldots, m_1)$ and the flow on arc $(v_{j_{t-1}}, v_{j_t})$ be $\beta_t + r$ $(t = 1, 2, \ldots, m_2)$. Then the α_t's and the β_t's are all nonnegative. Now consider the function

$$F(z) = \sum_{t=1}^{m_1} ||v_{i_t} - v_{i_{t-1}}|| f(\alpha_t + r - z) + \sum_{t=1}^{m_2} ||v_{j_t} - v_{j_{t-1}}|| f(\beta_t + r + z) \tag{3.1}$$

defined for $z \in [-r, r]$. Since $f(\bullet)$ is a concave function, $F(\bullet)$ is also concave. It follows from the property of concave functions [28] that

$$F(0) \geq \frac{1}{2}(F(-r) + F(r)) \geq \min\{F(-r), F(r)\}. \tag{3.2}$$

However, for any $z \in [-r, r]$, $F(z) - F(0)$ is the increase in the network cost caused by shifting z amount of the flow from π_1 to π_2. Inequality (3.2) says that we can shift r amount of flow from π_1 to π_2 without increasing the cost of the network when $F(r) = \min\{F(-r), F(r)\}$ and shift r amount of flow from π_2 to π_1 without increasing the cost of the network when $F(-r) = \min\{F(-r), F(r)\}$. Whenever we perform such a shift, there is at least one edge of G whose corresponding flow will be changed from a nonzero value to zero. Furthermore, if an edge of G has zero flow before the shift, it will still have zero flow after the shift because we are shifting some flow to a path that has a positive flow. Therefore, after a finite number of such shifts, we can eliminate all the duplicate paths, without increasing the cost of the network. Therefore, there exists a minimum cost pipe network R' such that there is a unique path from v_k to v_1 in N that has a positive flow for every $k = 2, 3, \ldots, n$. ∎

DEFINITION 3.3. *Given a point set P containing n points $\{p_1, \ldots, p_n\}$ on the Euclidean plane, a* full Steiner topology *for P is an undirected connected graph $G = (V, E)$, where the vertex set V contains $2n-2$ vertices $\{v_1, \ldots, v_n, v_{n+1}, \ldots, v_{2n-2}\}$ and the edge set E contains exactly $2n-3$ edges such that the degree of each of the first n vertices is 1 and the degree of each of the last $n-2$ vertices is 3. In other words, a full Steiner topology is a tree whose leaves correspond to the points in P and whose interior vertices (which correspond to Steiner points) all have degree 3.*

DEFINITION 3.4. *Let T be a Steiner topology. An edge is called a* Steiner-regular edge *if exactly one end point of this edge corresponds to a Steiner point (the other corresponds to a regular point). Let $e = \{u, v\}$ be a Steiner-regular edge where u corresponds to a regular point p; we can* shrink *this edge by deleting the edge e and replacing the two vertices u and v by a new vertex that corresponds to the regular point p. Any vertex other than u and v is connected to this new vertex if and only if it was connected to u or v before the shrinking operation. Notice that a Steiner topology changes to another Steiner topology when a Steiner-regular edge is shrunk, provided that the degree of the resulting new vertex is no more than 3. A Steiner topology T' is called a* degeneracy *of T if T' can be obtained from T using zero or more shrink operations. We use $D(T)$ to denote the set of Steiner topologies that are degeneracies of T.*

In the rest of this section, we will prove that there exists a full Steiner topology that has a realization that is a minimum cost pipe network interconnecting P. In other words, any Steiner topology T is a degeneracy of some full Steiner topology. Therefore, we only need to consider pipe networks that are realizations of full Steiner topologies. This is largely due to the fact that realizations permit all Steiner topologies to be handled *as if* they were full, by allowing zero-length edges. Note that the number of full Steiner topologies for n regular points is $\frac{(2n-4)!}{(n-2)!2^{n-2}}$, while the number of Steiner topologies is $\sum_{s=0}^{n-2} \binom{n}{s+2} \frac{(n+s-2)!}{s!}$. Therefore, the number of Steiner topologies is much larger than the number of full Steiner topologies [17, 32].

THEOREM 3.2. *There is a minimum cost pipe network interconnecting a point set P that is a realization of a full Steiner topology for P.*

Proof. From Theorem 3.1, we know that there exists a minimum cost pipe network interconnecting P that is a realization of a Steiner topology T. Assume that $P = \{p_1, p_2, \ldots, p_n\}$ and that the vertices of the topology are $\{v_1, v_2, \ldots, v_n,$

$v_{n+1}, \ldots, v_{n+m}\}$, where the first n vertices correspond to the points $\{p_1, p_2, \ldots, p_n\}$. The rest of the proof is composed of four parts.

1. *v_i $(i = 1, 2, \ldots, n)$ is a leaf vertex in T.* Let v_i be the interior vertex with the smallest index. If $i > n$, there is nothing to be proved. Now assume that $i \leq n$. Let $b_1, b_2, \ldots, b_k$ be all the vertices that are adjacent to v_i in T. We can change the name of vertex v_i to a new name v_{n+m+1}, then add a new vertex v_i and an edge $\{v_i, v_{n+m+1}\}$ to obtain a new Steiner topology T_1. We note that for any realization of T, there is a realization of T_1 with the same cost (obtained by forcing $p_{n+m+1} = p_i$). Repeating this process, we can obtain a Steiner topology in which the first n vertices are all leaf vertices.

2. *v_j $(j > n)$ is an interior vertex in T.* In any realization, the sum of flows into a Steiner point equals the sum of flows out of that Steiner point. Therefore, having a leaf Steiner vertex does not reduce the cost of the minimum cost realization of the topology. Hence, if v_j is a leaf vertex in T and $j > n$, we may delete v_j from T to obtain a new topology whose cost is no greater than the cost of T.

3. *Degree-2 interior vertices can be removed without increasing the cost.* This follows from the fact that the shortest connection between points is the straight line segment connecting them.

4. *A degree-k interior vertex can be split into $k-2$ degree-3 interior vertices without increasing the network cost* $(k \geq 4)$. Let k be an integer greater than or equal to 4. Let a be an interior vertex of degree k. Assume that the neighbors of a are $b_1, b_2, \ldots, b_k$. We may split the vertex a into a_1 and a_2, which are connected by an edge $\{a_1, a_2\}$, and replace the edges $\{a, b_1\}$ and $\{a, b_2\}$ by $\{a_1, b_1\}$ and $\{a_1, b_2\}$, replace the edges $\{a, b_j\}$ by $\{a_2, b_j\}$ for $j = 3, \ldots, k$. The resulting topology has a cost that is no greater than the cost of the previous topology. Repeating the above process, we can obtain a topology in which every interior vertex has degree exactly 3. This completes the proof of part 4 and also the proof of the theorem. ∎

4. Minimum cost network under a given topology. In this section, we show that the minimum cost pipe network under a given full Steiner topology can be computed efficiently. We will first show that the flows in the minimum cost realization of a given full Steiner topology can be computed in $O(n)$ time, without knowing the optimal locations of the Steiner points. We then show that computing the minimum cost network under a full Steiner topology is a special case of the well-studied problem of *minimizing a sum of Euclidean norms* [1, 2, 4, 5, 8, 24, 9, 34]. In [34], Xue and Ye presented a primal-dual interior-point algorithm for computing an ϵ-optimal solution [22] to the problem of minimizing a sum of Euclidean norms and proved that their algorithm requires $O(n^{1.5}(\log(n) + \log(\frac{\bar{c}}{\epsilon})))$ arithmetic operations if the problem has a tree structure, where $\bar{c}$ is a constant dependent on the input. We will show that this algorithm is also a polynomial time approximation scheme (PTAS) [18] for computing the minimum cost network under a given full Steiner topology that computes a $(1+\epsilon)$-approximation in $O(n^{1.5}(\log(n) + \log(\frac{1}{\epsilon})))$ time.

4.1. Computing the flows of the minimum cost network. Suppose that we are given a full Steiner topology. We may consider the tree to be rooted at v_1, which corresponds to the treatment site p_1. The root vertex has one child. Every other interior vertex has exactly two children. The leaf vertices $v_2, v_3, \ldots, v_n$ correspond to the wells $p_2, p_3, \ldots, p_n$. Clearly, in a minimum cost realization of the given topology, the flow from vertex v_k to its parent vertex must be $c(p_k)$ for $k = 2, 3, \ldots, n$. The flow from any interior vertex to its parent vertex must equal the sum of the flows into this vertex from its two children. Therefore, we can use a dynamic programming algorithm to compute the flows on all edges in linear time.

4.2. Approximating the optimal locations of the Steiner points. Once the flows of the minimum cost network under a given Steiner topology are computed, the problem of finding the optimal locations of the Steiner points becomes a special case of the following problem of minimizing a sum of Euclidean norms [24].

PROBLEM 4.1. *Let $c_1, c_2, \ldots, c_M \in R^2$ be column vectors in the Euclidean 2-space and $A_1, A_2, \ldots, A_M \in R^{N\times 2}$ be N-by-2 matrices each having full column rank. Find a point $u \in R^N$ such that the following sum of Euclidean norms is minimized:*

$$\begin{aligned} \min \quad & \sum_{i=1}^{M} ||c_i - A_i^T u|| \\ \text{s.t.} \quad & u \in R^N. \end{aligned} \tag{4.1}$$

This problem has been studied by Calamai and Conn [4, 5] and Overton [24], where second-order methods were proposed to solve the problem. Recently, Andersen [1], Conn and Overton [8], and Andersen and Christiansen [2] proposed computationally effective interior-point algorithms for solving (4.1). Xue and Ye [34] also proposed an interior-point algorithm for solving this problem and proved that their algorithm produces an ϵ-optimal solution in polynomial time.

DEFINITION 4.1. *Consider a minimization problem. Let ϵ be a positive number. A $(1+\epsilon)$-approximation is a feasible solution whose corresponding objective function value is no more than the* product *of the optimal objective function value and $(1+\epsilon)$. An ϵ-optimal solution is a feasible solution whose corresponding objective function value is no more than the* sum *of the optimal objective function value and ϵ.*

The notion of $(1+\epsilon)$-approximations can be found in [18], and the notion of ϵ-optimal solutions can be found in [22, 23, 34, 35].

In [34], a primal-dual interior-point algorithm was presented that computes an ϵ-optimal solution to problem (4.1) in polynomial time. They also proved that when the instance of problem (4.1) is obtained from a Euclidean multifacility location problem with a tree structure (as in the case of computing the minimum cost pipe network under a given tree topology), the total number of arithmetic operations required is $O(N^{1.5}(\log(N) + \log(\frac{\bar{c}}{\epsilon})))$, where $\bar{c} = \max_{1\le i\le M} ||c_i||$. We will show that this same algorithm is a PTAS that computes a $(1+\epsilon)$-approximation to the minimum cost network under a given full Steiner topology using $O(n^{1.5}(\log(n) + \log(\frac{1}{\epsilon})))$ arithmetic operations.

Let $x(v_i, v_j)$ be the flow on arc (v_i, v_j), so that $f(x(v_i, v_j))$ is the CPUL for the pipe interconnecting p_i and p_j. Therefore, optimal locations for the Steiner points can be determined by solving the following optimization problem:

$$\min \mathcal{F}(p_{n+1}, p_{n+2}, \ldots, p_{n+n-2}) = \sum_{v_j \text{ is the parent of } v_i} f(x(v_i, v_j))||p_i - p_j||. \tag{4.2}$$

The function in (4.2) is a nonsmooth, continuous, convex function. It is a special case of (4.1), where $N = 2n-4$, $M = 2n-3$; $c_1 = f(x(\text{child}(v_1), v_1))p_1$; $c_i = f(x(v_i, \text{parent}(v_i)))p_i$ for $i = 2, 3, \ldots, n$; and $c_i = 0$ for $i = n+1, n+2, \ldots, n+n-2$. The parent and child notations are those used in section 4.1.

Since problem (2.1) does not change if we *translate* the locations of the treatment site and all the wells by the same vector, we may assume, without loss of generality, that p_1 is at the origin, i.e., $p_1 = 0$. For $i = 2, 3, \ldots, n$, $||c_i|| = f(x(v_i, \text{parent}(v_i)))||p_i||$ is the cost of transporting the water from p_i to the treatment site via a straight line segment between p_i and p_1. Therefore, $\max_{1\le i\le n+n-2} ||c_i||$ is less than or equal to

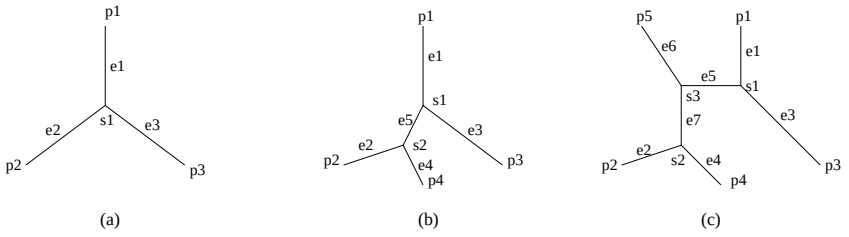

FIG. 1. *The correspondence between $n-3$ vectors and full Steiner topologies.*

the cost of the minimum cost network under the given topology (note that $p_1 = 0$ and $c_i = 0$, $i = n+1, n+2, \ldots, n+n-2$). Therefore, a $\bar{c}\epsilon$-optimal solution is also a $(1+\epsilon)$-approximation for the minimum cost network under a given full Steiner topology. To summarize, we have proved the following theorem.

THEOREM 4.1. *For any given full Steiner topology, a $(1+\epsilon)$-approximation to the minimum cost pipe network under this topology can be computed in $O(n^{1.5}(\log(n) + \log(\frac{1}{\epsilon})))$ time.* □

5. Partially enumerating the topologies. In [29], Smith proved the following theorem, which establishes a one-to-one correspondence between the set of full Steiner topologies on n regular points and a special set of $(n-3)$-element vectors.

THEOREM 5.1. *There is a one-to-one correspondence between full Steiner topologies on $n \geq 3$ fixed points, and $(n-3)$-element vectors $t = (t_1, t_2, \ldots, t_{n-3})$, whose ith entry t_i is an integer in the range $1 \leq t_i \leq 2i+1$. Therefore, the number of full Steiner topologies on n fixed points is $1 \cdot 3 \cdots (2n-5)$.* □

Figure 1 illustrates the one-to-one correspondence for the case of $n = 5$ and $t = (2, 5)$. Figure 1(a) illustrates the topology for three fixed points where the three edges are labeled e_1, e_2, and e_3. The only moving point is labeled s_1. To add the fourth fixed point into the network, we connect p_4 to an interior point on the edge labeled e_2 (since $t_{4-3} = 2$ in the topological vector). This point then becomes the second moving point s_2. Edge e_2 is broken into two parts, one part still labeled e_2 and the other labeled e_5 $(= 2 \times 4 - 3)$. The edge interconnecting p_4 and s_2 is labeled e_4 $(= 2 \times 4 - 4)$. After the above process, we obtain the topology for the first four fixed points, which is illustrated in Figure 1(b). Similarly, Figure 1(c) illustrates the topology for the first five fixed points, which is obtained by breaking the edge labeled e_5 $(= t_{5-3})$.

A brute-force algorithm for solving problem (2.1) is to compute the minimum cost network under a full Steiner topology for every full Steiner topology interconnecting the n fixed points. Since there are $1 \times 3 \times \cdots \times (2n-5)$ different full Steiner topologies for a set of n fixed points, a complete enumeration method is very expensive, even with the aid of the efficient algorithm of [34].

One way to tackle this problem is to use a backtrack algorithm that enumerates only part of the topologies. We need a bounding theorem that can be used to prune hopeless branches. Such a bounding theorem will be proved in the next subsection.

5.1. A bounding theorem.

THEOREM 5.2. *For any $k = 3, 4, \ldots, n-1$ and any $t_{k-2} \in \{1, 2, \ldots, 2k-3\}$, the cost of a full Steiner topology (interconnecting the points $p_1, p_2, \ldots, p_k$) with the topological vector $(t_1, t_2, \ldots, t_{k-3})$ is no greater than the cost of the full Steiner topology (interconnecting the points $p_1, p_2, \ldots, p_{k+1}$) with the topological vector $(t_1, t_2, \ldots, t_{k-3},$*

t_{k-2}). *Note that we assumed that the topological vector for* $p_1, p_2, \ldots, p_k$ *is a prefix of the topological vector for* $p_1, p_2, \ldots, p_{k+1}$.

Proof. This is a generalization of Theorem 4 in [29], which deals with flow-independent minimum cost networks. Suppose we have found a minimum cost realization of the topology for $p_1, p_2, \ldots, p_{k+1}$. Delete the leaf vertex corresponding to p_{k+1} from the rooted tree. This will reduce the CPUL for all the arcs along the path from the parent vertex of p_{k+1} to the root p_1, due to the assumption that $f(\bullet)$ is a monotonically nondecreasing function. In addition, the parent vertex of p_{k+1} can now be removed (since it is now a degree-2 interior vertex) to shorten the network. Optimizing the modified network will further reduce the cost. □

5.2. The backtrack algorithm. A backtrack algorithm for computing the minimum cost pipe network is given as Algorithm 1. It follows from Theorem 5.2 that Algorithm 1 correctly computes the minimum cost feasible network. However, for the algorithm to be practically efficient, we need to have a good initial upper bound and a good ordering of the regular points so that the backtrack algorithm will generate only a small portion of the whole tree.

ALGORITHM 1. Partially enumerating the topologies by backtracking.

Step_1 Compute an upper bound UB or set $UB := \infty$. Set $k := 4$ and $t_1 := 3$.
Step_2 Compute the minimum cost network under the topology with topological vector $(t_1, t_2, \ldots, t_{k-3})$. Let C be the cost of the current network.
Step_3 **if** $C \geq UB$ **then goto** Step_4 **else goto** Step_5 **endif**
Step_4 **if** $t_{k-3} > 1$ **then**
  $t_{k-3} := t_{k-3} - 1$
  goto Step_2
elseif $k = 4$ **then**
  stop ; UB is the minimum cost and $(bt_1, bt_2, \ldots, bt_{n-3})$ is the optimal topological vector.
else
  $k := k - 1$
  goto Step_4
endif
Step_5 **if** $k = n$ **then**
  Set $UB := C$ and save the current topological vector in $(bt_1, bt_2, \ldots, bt_{n-3})$.
  goto Step_4.
else
  $k := k + 1$
  $t_{k-3} := 2k + 1$
  goto Step_2
endif

5.3. Initial upper bound. In order for the backtrack algorithm to be effective, we also need a good initial upper bound. In the flow-independent case, one can always use the cost of the minimum spanning tree as the initial upper bound. In the flow-dependent case, even such an initial upper bound is not available. In this section, we present a min-min heuristic algorithm for computing the initial upper bound. The min-min heuristic builds up a tree network in the following way. Initially, only the treatment site p_1 is on the tree. The cost of this tree is zero. For each point $p \in P$, we

ALGORITHM 2. The min-min heuristic algorithm for initial upper bound.

Step_1 Let $q_1 := p_1$ and $c(q_1) := c(p_1)$. Let $Q := \{q_1\}$. Delete p_1 from P.
Step_2 **for** each $p \in P$ **do**
compute $w(p) = c(p)||q_1 - p||$
endfor
Let $q_2 := p_j$ and $c(q_2) := c(p_j)$, where p_j is a point in P and $w(p_j) := \min\{w(p)|p \in P\}$. Add q_2 to Q. Delete p_j from P.
Step_3 **for** each $p \in P$ **do**
compute the minimum cost network interconnecting q_1, q_2, and p.
Let $w(p)$ be the minimum cost.
endfor
Let $q_3 := p_j$ and $c(q_3) := c(p_j)$, where p_j is a point in P and $w(p_j) := \min\{w(p)|p \in P\}$. Add q_3 to Q. Delete p_j from P.
Let T_3 be the empty topological vector for Q.
Step_4 Let $k := |Q|$.
for each $p \in P$ **do**
Let $q_{k+1} := p$ and $c(q_{k+1}) := c(p)$.
for $i := 1$ **to** $2k - 3$ **do**
Let $f(p, i)$ be the cost of the topology (T_k, i) interconnecting $Q \cup \{q_{k+1}\}$.
endfor
Let $w(p) := \min\{f(p, i)|1 \le i \le 2k - 3\}$.
endfor
Let $q_{k+1} := p_j$ and $c(q_{k+1}) := c(p_j)$, where p_j is a point in P and $w(p_j) := \min\{w(p)|p \in P\}$. Add q_{k+1} to Q. Delete p_j from P.
Let T_{k+1} be the topological vector for Q which has a cost of $f(q_{k+1}, i) := w(q_{k+1})$ which is generated in the (appropriate) inner **for** loop.
Step_5 **if** $P = \emptyset$ **then stop** ; **otherwise goto** Step_4.

can add p to the current tree by directly interconnecting p with p_1. The associated (minimum) cost of adding this point to the current tree is given by $c(p)||p_1 - p||$. We select the point whose associated cost is minimum and add it to the current tree. This point is then deleted from P. Suppose that the current tree has k vertices and we are trying to add another point from P to the current tree, which has $2k - 3$ edges. Let p be a point in P. To connect p to the current tree, we may select an edge in the current tree and connect p to this edge (creating a new Steiner point). There are $2k-3$ such choices. Each such choice corresponds to a full Steiner topology interconnecting the regular points in the current tree and the point p. For each of these full Steiner topologies, there is an associated cost of the topology. We define the minimum of the costs of these topologies as the cost of point p. In the min-min heuristic, the point with minimum cost is selected to be the next point to be added to the current tree. This process continues until we have a tree interconnecting all the regular points.

Our min-min heuristic can be considered as a generalization of Prim's algorithm [26] for computing a minimum spanning tree; i.e., we are always selecting the next vertex whose addition to the tree will result in the smallest additional cost. Our motivation, however, comes from the incremental optimization heuristic of Dreyer and Overton [9].

The heuristic of Dreyer and Overton requires $O(n^2)$ local minimizations (solving an instance of Problem 4.1). Our min-min heuristic is more expensive: it requires $O(n^3)$ local minimizations. To see why this is so, we note that there are $2k - 3$

edges in the tree interconnecting k regular points and that there are $n-k$ regular points left to be selected to join the tree. Therefore, we need to perform $(n-k) \times (2k-3)$ local minimizations to select the $(k+1)$th point to join the tree. Therefore, the min-min heuristic requires $O(n^3)$ local minimizations. This time complexity is affordable, compared with the exponential time required by the backtrack algorithm. Our limited computational experiments show that this heuristic produces better initial upper bounds than using a random topology or the max-min heuristic to be discussed in the next section.

5.4. A heuristic ordering. In the backtrack algorithm, a branch is cut off only if its associated cost is no less than the cost of the current incumbent. If we use the ordering determined by the min-min heuristic, a cut-off is not likely to happen high in the search tree, because the cost of the current tree could be relatively small and a small mistake may not lead to a cost greater than the cost of the current incumbent.

ALGORITHM 3. The max-min heuristic ordering algorithm.

Step_1 Let $q_1 := p_1$ and $c(q_1) := c(p_1)$. Let $Q := \{q_1\}$. Delete p_1 from P.
Step_2 **for** each $p \in P$ **do** compute $w(p) = c(p)||q_1 - p||$ **endfor**
Let $q_2 := p_j$ and $c(q_2) := c(p_j)$ where p_j is a point in P and $w(p_j) := \max\{w(p)|p \in P\}$. Add q_2 to Q. Delete p_j from P.
Step_3 **for** each $p \in P$ **do**
compute the minimum cost network interconnecting q_1, q_2, and p.
Let $w(p)$ be the minimum cost.
endfor
Let $q_3 := p_j$ and $c(q_3) := c(p_j)$, where p_j is a point in P and $w(p_j) := \max\{w(p)|p \in P\}$. Add q_3 to Q. Delete p_j from P.
Let T_3 be the empty topological vector for Q.
Step_4 Let $k := |Q|$.
for each $p \in P$ **do**
Let $q_{k+1} := p$ and $c(q_{k+1}) := c(p)$.
for $i := 1$ **to** $2k-3$ **do**
Let $f(p,i)$ be the cost of the topology (T_k, i) interconnecting $Q \cup \{q_{k+1}\}$.
endfor
Let $w(p) := \min\{f(p,i)|1 \le i \le 2k-3\}$.
endfor
Let $q_{k+1} := p_j$ and $c(q_{k+1}) := c(p_j)$, where p_j is a point in P and $w(p_j) := \max\{w(p)|p \in P\}$. Add q_{k+1} to Q. Delete p_j from P.
Let T_{k+1} be the topological vector for Q which has a cost of $f(q_{k+1}, i) := w(q_{k+1})$ which is generated in the (appropriate) inner **for** loop.
Step_5 **if** $P = \emptyset$ **then stop** ; **otherwise goto** Step_4.

In this section, we present a max-min heuristic ordering such that cut-offs are likely to happen high in the search tree. In the max-min heuristic, we also start with the treatment site. For each point p in P, the associated cost is defined in the same way as in the min-min heuristic. However, we select the point that has the *maximum* cost as the next point to be included in the tree. This heuristic ordering algorithm is given as Algorithm 3.

The max-min heuristic algorithm also requires $O(n^3)$ local minimizations. Our computational results show that the initial upper bound produced by the max-min heuristic is usually not as good as that produced by the min-min heuristic. However,

TABLE 1
Number of full Steiner topologies $T(n)$ as a function of the number of regular points n.

n	$T(n)$	n	$T(n)$
3	1	8	10395
4	3	9	135135
5	15	10	2027025
6	105	11	34459425
7	945	12	654729075

the max-min ordering reduces dramatically the number of topologies to be considered in the backtrack algorithm.

6. A 5-optimal heuristic algorithm. Because the minimum cost pipe network problem is NP-hard, the exact algorithm described in the previous section is too expensive to be practical when n is large. For n regular points, the number of different full Steiner topologies is $1 \times 3 \times 5 \times \cdots \times (2n-5)$. These values for $3 \le n \le 12$ are illustrated in Table 1. Although we can compute the minimum cost network under a given full Steiner topology very efficiently, the backtrack algorithm is impractical for large n, limited by the huge number of topologies.

In this section, we introduce the notion of k-optimality of a pipe network and present efficient algorithms for checking whether a given pipe network is 5-optimal. In case the given pipe network is not k-optimal, our algorithm also provides a pipe network that has a lower cost. Our notion of k-optimality for pipe networks is motivated by the notion of k-optimality for ESMTs discussed in a private communication of Overton and Xue [25] and the ideas in [11].

DEFINITION 6.1. *Let T be a full Steiner topology interconnecting the points in P. Let k be an integer such that $3 \le k \le n$. A size-k component (C) of T is a subtree of T which has k leaf vertices (in C) and no degree-2 vertices (in C).*

DEFINITION 6.2. *Let T be a full Steiner topology interconnecting the points in P. Let R be the minimum cost network under topology T. A size-k component C of T defines a minimum cost pipe network problem with k regular points, where the point in R that corresponds to the unique vertex in C that is the ancestor of all the other vertices in C is the* new treatment site *and every other point in R that corresponds to a leaf vertex of C is a* new well *whose* new capacity *is defined to be the amount of flow that needs to be transmitted from its corresponding vertex to v_1 in T. T is said to be a k-optimal topology if for any size-k component C of T, its corresponding leaf vertices are connected optimally in the minimum cost realization R of T.*

Note that, given a full Steiner topology, the optimal flows (flows in the minimum cost network under the given topology) on the edges can be computed without knowing the locations of the Steiner points. Therefore, given locations of the leaf vertices of a component C in R, we can formulate another minimum cost pipe network problem where the leaf vertices of C are treated as regular points located at the corresponding locations given in R. In order to check for k-optimality, we need to be able to select all the size-k components and to be able to find the minimum cost pipe network for k given points efficiently. In the following, we illustrate how to check for 5-optimality efficiently.

Figure 2 illustrates a size-5 component. If the vertices a, b, c, d, e are treated as regular points, then the vertices x, y, z are the Steiner points. Note that such a size-5 component is uniquely determined by the *center vertex* y and the vertex c. For any Steiner point y, there are at most three choices of vertex c (since the degree of y is

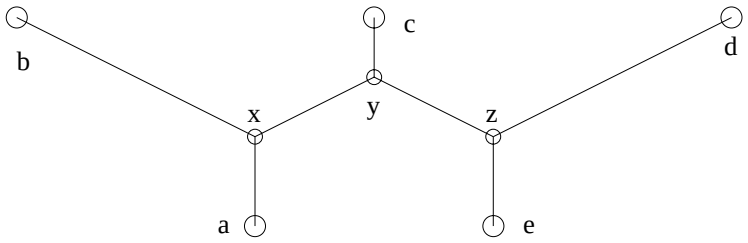

FIG. 2. *A size-5 component.*

3). Once y and c are chosen, we can find the *set* of vertices $\{a, b, d, e\}$ if the other two neighbor vertices of y are both Steiner points. Since there are $n-2$ Steiner points in a full Steiner topology interconnecting n regular points, there are at most $n-2$ choices for y. Therefore, there are at most $3n-6$ size-5 components in a full Steiner topology interconnecting n regular points. Since there are fifteen possible full Steiner topologies interconnecting five regular points, we need to solve at most $45n-90$ minimum cost networks under a full Steiner topology interconnecting five points. Therefore, we have proved the following theorem.

THEOREM 6.1. *Let R be the minimum cost realization of a full Steiner topology T interconnecting P. We can check that T is 5-optimal or find a topology whose cost is lower than the cost of T after solving at most $45n-90$ minimum cost networks under a full Steiner topology interconnecting five points.* □

Now we can present our heuristic algorithm for computing a 5-optimal pipe network interconnecting n regular points. This algorithm is presented as Algorithm 4.

ALGORITHM 4. A 5-optimal heuristic algorithm.

```
Step_1 Run either the min-min heuristic or the max-min heuristic to obtain a topology T
       and its minimum cost realization R.
Step_2 if running out of time then
          Output the current topology T and its realization R; stop
       endif
Step_3 for each size-5 component C of T do
          if C is not optimally connected in T do
             Change topology T by replacing the connections of C by
             its optimal connection;
             Compute the realization R of this new topology T.
             goto Step_2.
          endif
       endif
Step_4 The topology T is 5-optimal. Output T and its realization R.
```

We do not know the time complexity of Algorithm 4 because we do not know how many changes of topology are needed to reach a 5-optimal topology. All we know is the following. Step_1 requires $O(n^3)$ local minimizations. After that, we need to perform at most $45n-90$ local minimizations (for five-point instances) to either find a *better* topology or confirm that the current topology is 5-optimal. Our computational results show that this algorithm finds an optimal solution or a good suboptimal solution in much less time, compared with the backtrack algorithm.

7. Computational results. In this section, we present preliminary computational results for both the backtrack algorithm and the 5-optimal heuristic algorithm.

TABLE 2
Constants used in the computation of the CPUL.

ϵ	g	S_f	ν	α	β	γ
0.000015	32.2	0.003	0.0000166	4.7156213	40.406146	1.0727788

TABLE 3
The eight regular points in test problem 1.

i	$x(i)$	$y(i)$	Capacity	i	$x(i)$	$y(i)$	Capacity
01	1647846.62	432550.91		05	1648180.00	433210.00	0.0278
02	1646685.00	432430.00	0.1114	06	1648440.00	433470.00	0.0278
03	1647540.00	433210.00	0.0278	07	1648960.00	432950.00	0.0278
04	1647545.00	432690.00	0.1114	08	1649220.00	431650.00	0.0836

Both algorithms were implemented in F77 and run on a 100-MHz SGI Indy workstation with MIPS R4000 CPU and 1-MB secondary cache. In all cases, the duality gap tolerance was chosen as 0.00001.

In our first three test problems, we have used data from an application to a groundwater remediation problem at the Lawrence Livermore National Laboratory in Livermore, CA. The locations of the wells were the result of a particular management policy imposed upon an optimization method presented by Rizzo and Dougherty [27]. Treatment sites were located as the weighted center of each of three well fields. To further test the performances of the backtrack algorithm and the 5-optimal heuristic algorithm, we combine the well fields of test problems 2 and 3 to form a field of 18 wells. We then apply the algorithms to the first 15 and first 16 regular points in this list. The fifth test problem was randomly generated to test the 5-optimal heuristic algorithm. For all but one of the cases in the first four test problems, the 5-optimal heuristic that starts with the min-min heuristic found the optimal solution. The 5-optimal heuristic that starts with the max-min heuristic found the optimal solution in all of the first four test problems. Based on our computational result, we estimate that the run time of the backtrack algorithm on the fifth problem is about 800 years of CPU time on our SGI workstation. Therefore, we did not apply that backtrack algorithm on this problem. As a result, we do not know if the result of the 5-optimal heuristic algorithm is optimal or suboptimal.

For all test problems, the CPUL is computed in the following way [30]. For a given flow-rate q, the diameter d of the pipe is computed by

$$d(q) = 0.66 \left[\epsilon^{1.25} \left(\frac{q^2}{gS_f} \right)^{4.75} + \nu q^{9.4} \left(\frac{1}{gS_f} \right)^{5.2} \right]^{0.04}, \tag{7.1}$$

and the CPUL is $f(q)$, which is defined as

$$f(q) = \alpha + \beta d(q)^{\gamma}, \tag{7.2}$$

where the constants are given in Table 2.

Those constants are determined by the energy gradient, the material properties of the pipe and fluid, the cost of digging a trench, etc. For more details on those engineering properties, we refer readers to Lillys [20].

The input data for test problem 1 are given in Table 3. For each regular point, the table shows its index, its x-coordinate, its y-coordinate, and its capacity. Note that the capacity entry for the first regular point is empty. This means that the first regular point is the treatment site, whose capacity can be computed as the sum of the capacities of the other regular points. We will use the same format for all the other problems.

TABLE 4
Test results for test problem 1.

	Initial bound	Final func	Time (sec)	Search perc
Original ordering	87015.910155	73314.693982	230.56	0.10E+01
max-min ordering	84202.713809	73314.693982	21.57	0.84E-01
min-min 5-opt	84235.450263	73314.693982	2.81	
max-min 5-opt	84202.713809	73314.693982	3.70	

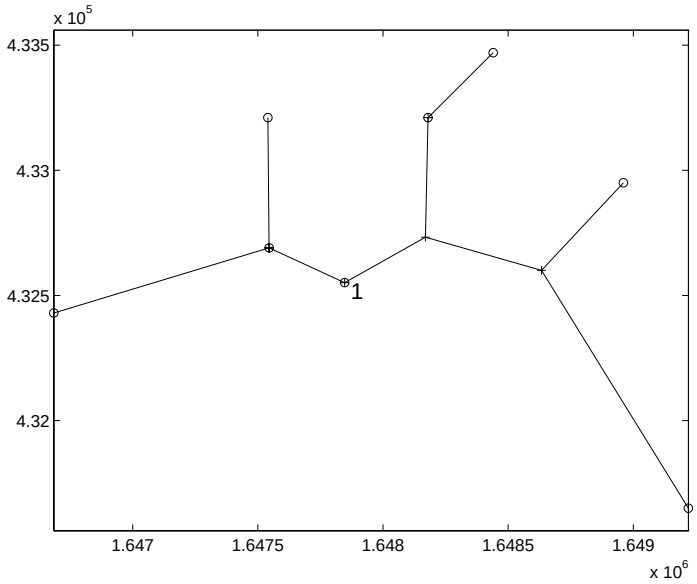

FIG. 3. *The optimal pipe network for test problem* 1.

Computational results for the first test problem are presented in Table 4. The first row of the table reports the result of the backtrack algorithm without any ordering heuristic, i.e., using the ordering given in the input. The initial upper bound 87015.910155 is obtained from the topological vector whose ith entry is $2i+1$ ($i = 1, 2, \ldots, n-3$). The final cost 73314.693982 (which is the optimal cost in this case) can be found in the column "Final func." The run time of the algorithm is 230.56 seconds. The last column shows the percentage of the total number of topologies searched. In this case, we have searched all the different topologies. The second row of the table reports the result of the backtrack algorithm with the max-min heuristic. Note that the number of topologies searched is only 8.4% of the total number of topologies. As a result, the run time required is only 21.57 seconds. The third row of the table reports the result of the 5-optimal heuristic algorithm. The last entry is left empty because it is hard to compare the minimization of a size-5 component with the minimization of a size-n component. For this problem, the 5-optimal heuristic algorithm using the min-min heuristic found the optimal solution in 2.81 seconds. The 5-optimal heuristic algorithm using the max-min heuristic found the optimal solution in 3.70 seconds. The minimum cost pipe network for this problem is illustrated in Figure 3, where a regular point is denoted by an o and a Steiner point is denoted by a +. To keep the picture clean, we only labeled the first regular point by the label 1 to the right of the regular point.

TABLE 5
The nine regular points in test problem 2.

i	$x(i)$	$y(i)$	Capacity	i	$x(i)$	$y(i)$	Capacity
01	1652303.48	434129.13		06	1651820.00	434250.00	0.0334
02	1652340.00	433730.00	0.0278	07	1652860.00	434250.00	0.0278
03	1652860.00	433730.00	0.0278	08	1652080.00	434510.00	0.0334
04	1651560.00	433990.00	0.0334	09	1652600.00	434510.00	0.0278
05	1652600.00	433990.00	0.0278				

TABLE 6
Test results for test problem 2.

	Initial bound	Final func	Time (sec)	Search perc
Original ordering	46876.355882	36139.255833	485.59	0.16E+00
max-min ordering	42078.166784	36139.255833	121.63	0.44E-01
min-min 5-opt	42115.487371	36649.223822	5.39	
max-min 5-opt	42078.166784	36139.255833	3.68	

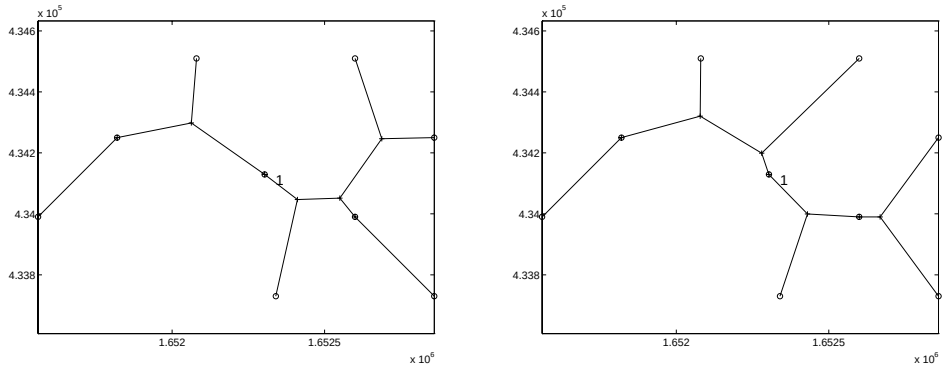

FIG. 4. *The optimal (left) and suboptimal (right) pipe network for test problem 2.*

The input data for test problem 2 are given in Table 5. The computational result for this problem is given in Table 6. For this problem, the backtrack algorithm without heuristic ordering found the optimal solution in 485.59 seconds, searching 16% of the total number of topologies. The backtrack algorithm with the max-min heuristic ordering found the optimal solution in 121.63 seconds, searching 4.4% of the total number of topologies. The 5-optimal heuristic algorithm using the min-min heuristic spent 5.39 seconds finding a suboptimal solution whose cost is 1.014 times the optimal cost. The 5-optimal heuristic algorithm using the max-min heuristic spent 3.68 seconds. It successfully found the optimal solution.

The minimum cost pipe network (left) and the suboptimal pipe network (right) for test problem 2 are illustrated in Figure 4. Note that the 5-optimal heuristic algorithm using the min-min heuristic failed to connect the two regular points at the top-right corner with a Steiner point. This is as expected, because the 5-optimal heuristic only checks for size-5 components. A k-optimal heuristic algorithm with a larger k might produce better solutions, at the cost of longer run time.

Table 7 illustrates the input data and the computational result for test problem 3. For this problem, the backtrack algorithm without heuristic ordering found the optimal solution in 29309.47 seconds, searching 3.2% of the total number of topolo-

TABLE 7
The input data (top) and test results (bottom) for test problem 3.

i	$x(i)$	$y(i)$	Capacity	i	$x(i)$	$y(i)$	Capacity
01	1652204.14	432529.22		07	1652860.00	432690.00	0.0334
02	1652340.00	431650.00	0.0334	08	1652600.00	432950.00	0.0334
03	1652860.00	431650.00	0.0334	09	1651820.00	433210.00	0.0334
04	1651040.00	431910.00	0.0500	10	1652340.00	433210.00	0.0334
05	1651560.00	432430.00	0.0334	11	1652860.00	433210.00	0.0334
06	1652340.00	432690.00	0.0334				

	Initial bound	Final func	Time (sec)	Search perc
Original ordering	88021.645072	66484.340380	29309.47	0.32E-01
max-min ordering	86697.786286	66484.340380	340.99	0.41E-03
min-min 5-opt	79166.346702	66484.340380	9.72	
max-min 5-opt	86697.786286	66484.340380	9.39	

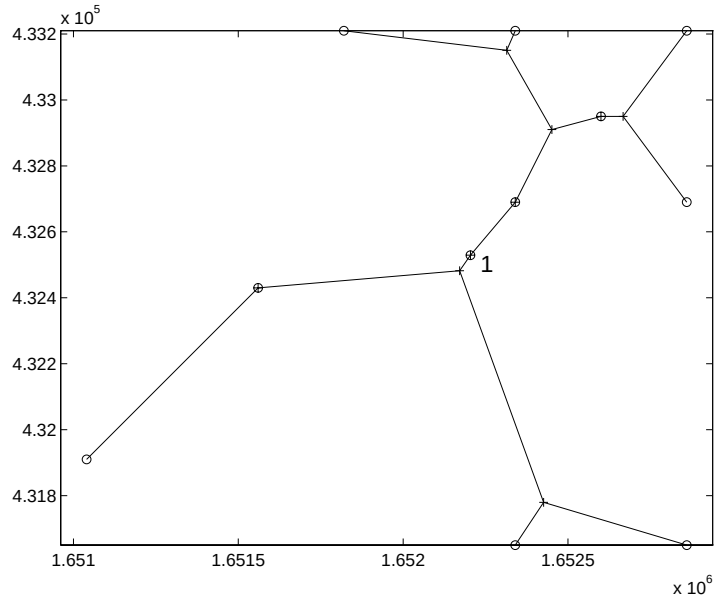

FIG. 5. *The optimal pipe network for test problem* 3.

gies. The backtrack algorithm with the max-min heuristic ordering found the optimal solution in 340.99 seconds, searching only 0.041% of the total number of topologies. The 5-optimal heuristic algorithm using the min-min heuristic found the optimal solution in 9.72 seconds. The 5-optimal heuristic algorithm using the max-min heuristic found the optimal solution in 9.39 seconds. Figure 5 illustrates the minimum cost pipe network for this problem.

The regular points and associated capacities for test problem 4 are given in Table 8. We have applied the algorithms to the first 15 regular points and the first 16 regular points, respectively. Table 9 illustrates the computational results. In both cases, the 5-optimal heuristic algorithm found the optimal solutions within a minute. The backtrack algorithm spent 3.36 hours for the 15-point case and 7.82 hours for the 16-point case. This shows that the run time of the backtrack algorithm is doubled with an additional regular point. Therefore, to apply the backtrack algorithm to the

TABLE 8
The 16 *regular points in test problem* 4.

i	$x(i)$	$y(i)$	Capacity	i	$x(i)$	$y(i)$	Capacity
01	1652296.67	433253.33		09	1652340.00	432690.00	0.0334
02	1651040.00	431910.00	0.0500	10	1652340.00	433210.00	0.0334
03	1651560.00	432430.00	0.0334	11	1652340.00	433730.00	0.0278
04	1651560.00	433990.00	0.0334	12	1652600.00	432950.00	0.0334
05	1651820.00	433210.00	0.0334	13	1652600.00	433990.00	0.0278
06	1651820.00	434250.00	0.0334	14	1652600.00	434510.00	0.0278
07	1652080.00	434510.00	0.0334	15	1652860.00	431650.00	0.0334
08	1652340.00	431650.00	0.0334	16	1652860.00	432690.00	0.0334

TABLE 9
Test results for test problem 4.

$n = 15$	Initial bound	Final func	Time (sec)	Search perc
max-min ordering	157470.028030	98133.591436	12127.08	0.50E-07
min-min 5-opt	147002.048885	98133.591436	31.96	
max-min 5-opt	157470.028030	98133.591436	36.15	
$n = 16$	Initial bound	Final func	Time (sec)	Search perc
max-min ordering	166726.146322	103061.764655	28141.68	0.41E-08
min-min 5-opt	157363.223440	103061.764655	43.00	
max-min 5-opt	166726.146322	103061.764655	47.37	

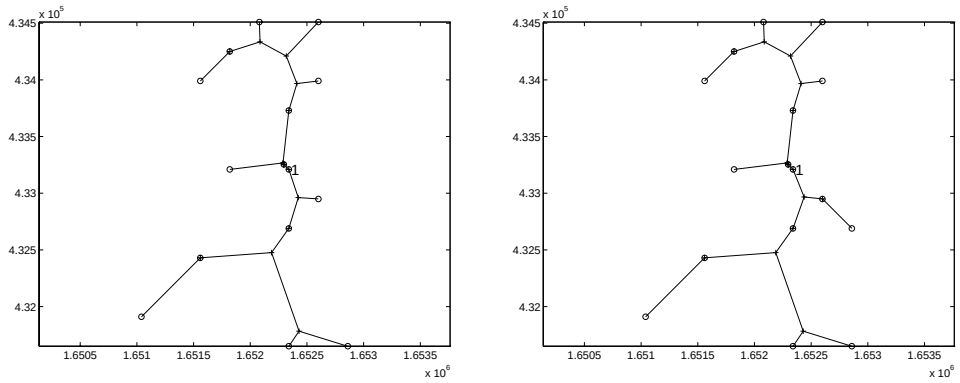

FIG. 6. *The optimal pipe networks for* 15 *points (left) and* 16 *points (right).*

36-point case is not realistic with the current computing power. On one hand, our computational results show that 17 or 18 regular points are about the limit for computing an optimal solution using the exact algorithm in one workstation CPU day. On the other hand, these results show that the 5-optimal heuristic is very practical. The optimal pipe network for both cases in test problem 4 are illustrated in Figure 6.

In test problem 5, we used 36 regular points which are selected from a huge set of grid points [20]. The coordinates and capacities of the regular points are illustrated in Table 10. For this problem, we applied the 5-optimal heuristic algorithm with the min-min heuristic and produced a network with a cost of 693567.504664. The 5-optimal heuristic found a network whose cost is 416116.527856, after making 84 changes in the topology. Note that this is a 40% reduction in the initial cost. The total run time is 1042.43 seconds. We suspect that the result is suboptimal but did not verify it because the estimated run time of the exact algorithm is about 800 years on a workstation. The resulting network is illustrated in Figure 7.

TABLE 10
The 36 regular points in test problem 5.

i	$x(i)$	$y(i)$	Capacity	i	$x(i)$	$y(i)$	Capacity
01	1647400	432950		19	1651040	433730	0.032801
02	1651820	432170	0.071895	20	1649220	433990	0.050551
03	1651560	433210	0.036717	21	1652860	431650	0.070822
04	1648960	434250	0.105160	22	1650520	433210	0.128993
05	1648960	433990	0.121502	23	1651040	431910	0.187886
06	1647400	433730	0.119121	24	1648700	434510	0.189312
07	1647140	431910	0.064512	25	1652080	432170	0.144190
08	1651040	434250	0.011117	26	1648960	434510	0.045043
09	1647400	432170	0.066358	27	1646880	431650	0.038324
10	1650000	433730	0.115583	28	1651820	432950	0.136873
11	1649220	434250	0.069011	29	1646880	431910	0.127853
12	1649480	433990	0.198850	30	1651820	434250	0.138132
13	1648700	433210	0.166491	31	1650520	434510	0.009700
14	1649480	432430	0.156950	32	1652600	433470	0.080098
15	1652340	433210	0.063542	33	1648440	432950	0.085227
16	1647400	432430	0.040603	34	1648180	432170	0.038722
17	1651300	433210	0.035788	35	1648700	433990	0.130234
18	1649220	432170	0.034979	36	1651820	433990	0.174764

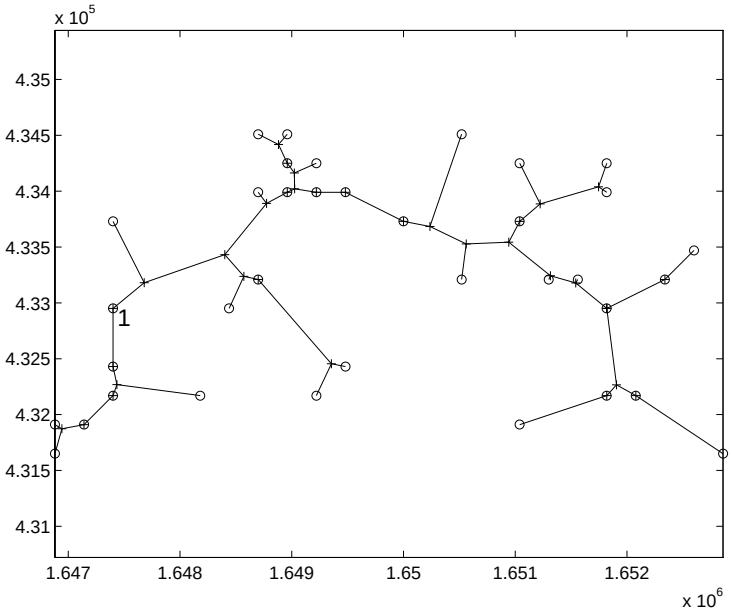

FIG. 7. *The 5-optimal network for the 36 points in test problem 5.*

8. Conclusions. We have presented a backtrack algorithm for computing the minimum cost pipe network interconnecting a single sink and many sources. This algorithm, when used with the max-min heuristic ordering, can solve problems with 11 regular points on a workstation in several minutes. It can also solve a problem with 15 regular points on a workstation in an hour. For larger problems, the algorithm becomes too expensive. In order to provide *good* suboptimal solutions to larger problems, we have also presented a 5-optimal algorithm for computing 5-optimal pipe networks. This algorithm, which starts from a min-min heuristic, can change to a *better* pipe network in $O(n)$ time if the current one is not 5-optimal. Our computa-

tional results show that this algorithm is very fast and often finds optimal or close to optimal solutions. Several interesting problems remain unsolved. We mention a few of them to conclude this paper.

The first one concerns the quality of the initial upper bound. For the ESMT problem, the cost of the minimum spanning tree is within a factor of $\frac{2}{\sqrt{3}}$ of the cost of the Steiner minimum tree [10, 14]. For the problem considered in this paper, there is no known polynomial time algorithm that can compute a solution whose cost is within a constant factor of the optimal cost. We suspect that both the min-min heuristic and the max-min heuristic produce 2-approximations to the minimum cost pipe network. Our computational results support this conjecture. However, we have not been able to arrive at a proof.

Although the number of full Steiner topologies is much smaller than the number of Steiner topologies, it is still superexponential in n. Therefore, it is very important to limit the number of topologies considered. For the ESMT problem, Winter and Zachariasen [32, 33] have developed a very effective technique of enumerating *equilateral points* (eq-points for short). They can rule out the vast majority of full topologies that cannot possibly have a degeneracy corresponding to the optimal topology. Generalizing the concept of eq-points to the pipe network problem is an intriguing topic for further research.

When there are several treatment sites and several wells, the problem becomes harder and more interesting. We are currently investigating this problem. Also, the 5-optimal algorithm can be implemented in parallel because the checking of different size-5 components can be done independently. Results on parallel implementations of the 5-optimal heuristic algorithm will be reported in a separate paper.

Acknowledgment. The authors would like to thank two referees and the associate editor for their helpful comments and remarks on the first version of the paper.

REFERENCES

[1] K.D. Andersen, *An efficient Newton barrier method for minimizing a sum of Euclidean norms*, SIAM J. Optim., 6 (1996), pp. 74–95.

[2] K.D. Andersen and E. Christiansen, *A Symmetric Primal-Dual Newton Method for Minimizing a Sum of Norms*, Manuscript, Odense University, Denmark, 1995.

[3] S. Bhaskaran and F.J.M. Salzborn, *Optimal design of gas pipeline network*, J. Oper. Res. Soc., 30 (1979), pp. 1047–1060.

[4] P.H. Calamai and A.R. Conn, *A stable algorithm for solving the multifacility location problem involving Euclidean distances*, SIAM J. Sci. Stat. Comput., 1 (1980), pp. 512–526.

[5] P.H. Calamai and A.R. Conn, *A projected Newton method for l_p norm location problems*, Math. Programming, 38 (1987), pp. 75–109.

[6] E.J. Cockayne and D.E. Hewgill, *Exact computation of Steiner minimal trees in the plane*, Inform. Process. Lett., 22 (1986), pp. 151–156.

[7] E.J. Cockayne and D.E. Hewgill, *Improved computation of plane Steiner minimal trees*, Algorithmica, 7 (1992), pp. 219–229.

[8] A.R. Conn and M.L. Overton, *A Primal-Dual Interior Point Method for Minimizing a Sum of Euclidean Distances*, manuscript, 1994.

[9] D.R. Dreyer and M.L. Overton, *Two heuristics for the Steiner tree problem*, J. Global Optim., 13 (1998), pp. 95–106.

[10] D.Z. Du and F.K. Hwang, *An approach for proving lower bounds: solution of Gilbert-Pollak conjecture on Steiner ratio*, in Proceedings of 31st IEEE Foundations of Computer Science, 1990, pp. 76–85.

[11] D.Z. Du, Y.J. Zhang, and Q. Feng, *On better heuristic for Euclidean Steiner minimum trees*, in Proceedings of 32nd IEEE Foundations of Computer Science, 1991, pp. 431–439.

[12] M.R. Garey, R.L. Graham, and D.S. Johnson, *The complexity of computing Steiner minimal trees*, SIAM J. Appl. Math., 32 (1977), pp. 835–859.

[13] E.N. Gilbert, *Minimum cost communication networks*, Bell System Tech. J., 46 (1967), pp. 2209–2227.
[14] E.N. Gilbert and H.O. Pollak, *Steiner minimal trees*, SIAM J. Appl. Math., 16 (1968), pp. 1–29.
[15] F. Harary, *Graph Theory*, Addison-Wesley, New York, 1969.
[16] F.K. Hwang, *A primer of the Euclidean Steiner problem*, Ann. Oper. Res., 33 (1991), pp. 73–84.
[17] F.K. Hwang, D.S. Richard, and P. Winter, *The Steiner Tree Problem*, Ann. Discrete Math. 53, North-Holland, Amsterdam, 1992.
[18] E. Horowitz and S. Sahni, *Fundamentals of Computer Algorithms*, Computer Science Press, Rockville, MD, 1978.
[19] D.H. Lee, *Low cost drainage networks*, Networks, 6 (1976), pp. 351–371.
[20] T.P. Lillys, *Optimal Piping Networks for Sub-Surface Remediation Designs*, M.S. thesis, Department of Civil and Environmental Engineering, University of Vermont, Burlington, 1997.
[21] Z.A. Melzak, *On the problem of Steiner*, Canad. Math. Bull., 4 (1961), pp. 143–148.
[22] Yu. E. Nesterov and A. Nemirovskii, *Interior Polynomial Algorithms in Convex Programming,* SIAM, Philadelphia, 1994.
[23] Yu. E. Nesterov and M. J. Todd, *Self-scaled barriers and interior-point methods for convex programming*, Math. Oper. Res., 22 (1997), pp. 1–42.
[24] M.L. Overton, *A quadratically convergent method for minimizing a sum of Euclidean norms*, Math. Programming, 27 (1983), pp. 34–63.
[25] M.L. Overton and G.L. Xue, *private communications,* February 1996.
[26] R.C. Prim, *Shortest connection networks and some generalizations*, Bell System Tech. J., 36 (1967), pp. 1389–1401.
[27] D.M. Rizzo and D.E. Dougherty, *Design optimization for multiple management period groundwater remediation*, Water Resources Research, 32 (1996), pp. 2549–2562.
[28] R.T. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.
[29] W.D. Smith, *How to find Steiner minimal trees in Euclidean d-space*, Algorithmica, 7 (1992), pp. 137–177.
[30] P.K. Swamee and K.J. Akalank, *Explicit equations for pipe-flow problems*, J. Hydraulic Engineering, ASCE, 102 (1976), pp. 657–664.
[31] P. Winter, *An algorithm for the Steiner problem in the Euclidean plane*, Networks, 15 (1985), pp. 323–345.
[32] P. Winter and M. Zachariasen, *Large Euclidean Steiner Minimum Trees in an Hour*, Technical report 96/34, http://www.diku.dk/~pawel/publications.html.
[33] P. Winter and M. Zachariasen, *Large Euclidean Steiner minimum trees: An improved exact algorithm*, Networks, 30 (1998), pp. 149–66.
[34] G.L. Xue and Y.Y. Ye, *An efficient algorithm for minimizing a sum of Euclidean norms with applications*, SIAM J. Optim., 7 (1997), pp. 1017–1036.
[35] Y.Y. Ye, *Interior-point algorithms for quadratic programming*, in Recent Developments in Mathematical Programming, S. Kumar, ed., Gordon and Breach Science, New York, 1991, pp. 237–262.
[36] J.Z. Zhang and D.T. Zhu, *A bilevel programming method for pipe network optimization*, SIAM J. Optim., 6 (1996), pp. 838–857.