The Quadratic Minimum Spanning Tree Problem

Arjang Assad
College of Business and Management, University of Maryland, College Park, Maryland 20742

Weixuan Xu
Institute of Applied Mathematics, Academia Sinica, Beijing, People's Republic of China

This article introduces a new optimization problem that involves searching for the spanning tree of minimum cost under a quadratic cost structure. This *quadratic minimum spanning tree* problem is proven to be NP-hard. A technique for generating lower bounds for this problem is discussed and incorporated into branch-and-bound schemes for obtaining exact solutions. Two heuristic algorithms are also developed. Computational experience with both exact and heuristic algorithms is reported.

1. INTRODUCTION

Let $G = (V, E)$ denote a connected undirected graph with edge set $E = \{e_1, \ldots, e_m\}$ and vertex set $V = \{v_1, \ldots, v_n\}$.

$$x_i = \begin{cases} 1, & \text{if edge } e_i \text{ is selected,} \\ 0, & \text{otherwise.} \end{cases}$$

The edges corresponding to nonzero components of the vector $x = (x_1, \ldots, x_m)$ define a subgraph of G. Let $\mathcal{T}$ denote the set of all such vectors that correspond to trees.

The well-known minimum spanning tree problem (MST) can be formulated as

$$\text{(MST)} \qquad \min \left\{ \sum_{i=1}^{m} c_i x_i \mid x \in \mathcal{T} \right\}, \tag{1}$$

where c_i is the cost attached to edge e_i. Algorithms for the minimum spanning tree problem date back to the early work of Kruskal [17] and Prim [22]. In the last two decades, the complexity of these algorithms has been reduced. Yao [28] provided an $O(m \log \log n)$ algorithm for a network with n vertices and m edges. Fredman and Tarjan [9] improved upon this bound with an $O(m\beta(m,$

Naval Research Logistics, Vol. 39, pp. 399–417 (1992)
 CCC 0894-069X/92/030399-19$04.00

$n)$) procedure ($\beta(m, n) = \min\{i | \log^{(i)} n \leq m/n\}$ where $\log^{(i)}(n)$ is the iterated logarithm). Gabow et al. [11] give a further improvement. Good descriptions of MST algorithms appear in [4] and [26].

In this article, we are interested in a spanning tree that minimizes a quadratic cost structure. We thus replace the linear objective in (1) with a quadratic form to obtain a new problem which may be called the quadratic minimum spanning tree problem (QMST) and is formulated as

$$\text{(QMST)} \qquad \text{minimize} \left\{ z(x) = \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} a_{ij} x_i x_j + \sum_{i=1}^{m} b_i x_i \,|\, x \in \mathcal{T} \right\}. \tag{2}$$

Here, we assume that a cost a_{ij} is associated with each pair of edges (e_i, e_j) with $i \neq j$. We call the a_{ij}'s *intercosts*. Note that there is also a direct cost b_i associated with selecting edge i.

The cost structure in (2) allows interaction effects between pairs of edges to be modeled. For example, when transmitting from one pipe to another, the costs may depend on the nature of the interface between the two pipes. The same pairwise interaction effect arises in the connection of overground and underground cables or in a transportation or road network with turn penalties. In all these cases, the presence of intercosts points to the quadratic (rather than linear) minimum spanning tree problem. One special case of the QMST that is likely to arise in practice is where the interaction effects are limited to adjacent edges alone. This means that a_{ij} vanishes if edges e_i and e_j are not adjacent. We shall return to this problem in Section 2.

The QMST also arises in the context of the following probabilistic version of the minimum spanning tree problem first introduced by Ishii and co-workers [13, 14]. Consider the minimum spanning tree problem of (1) in the case where the edge costs $c = (c_1, \ldots, c_m)$ are uncertain. One may model the cost vector c as a random variable $(\tilde{c})$ following a multivariate normal distribution; that is,

$$\tilde{c} \sim N(\mu, \Sigma)$$

where $\mu = (\mu_1, \ldots, \mu_m)$ is the mean cost vector $E(\tilde{c})$ and $\Sigma = (\sigma_{ij})$ is the variance-covariance matrix for $\tilde{c}$.

The probabilistic optimization problem:

$$\text{(SMST)} \qquad \text{minimize } z,$$

$$\text{subject to } \Pr\left\{ \sum_{i=1}^{m} \tilde{c}_i x_i \leq z \right\} \geq \alpha,$$

$$x \in \mathcal{T}$$

is called the *stochastic minimum spanning tree problem* (SMST). Standard transformations of chance-constrained programming (see Charnes and Cooper

[5] and Xu [27]) show that the preceding problem is equivalent to the deterministic problem:

$$\text{minimize} \left\{ z(x) = \sum_{i=1}^{m} \mu_i x_i + q_\alpha \left[\sum_{i=1}^{m} \sum_{j=1}^{m} \sigma_{ij} x_i x_j \right]^{1/2} \mid x \in \mathcal{T} \right\}, \tag{3}$$

where q_α is the α quantile of the standard normal distribution.

Ishii et al. [13, 14] address a special case of the SMST where the random variables representing edge costs are mutually independent so that the matrix Σ is diagonal. They propose an efficient algorithm to solve this restricted problem. When the edge costs are correlated, the objective in (3) will contain quadratic terms that complicate the problem. If all μ_i's are equal, problem (3) reduces to the QMST defined in (2) since the x_i's must sum to $n - 1$ for any $x \in \mathcal{T}$. The quadratic minimum spanning tree problem is thus a special case of the stochastic spanning tree problem. This provides another motivation to study the QMST.

The next section of this article investigates the complexity of the QMST and shows it to be NP-hard. A technique for obtaining lower bounds for this problem is described in Section 3. This technique is of some independent interest due to its simplicity. Here, it is used in the context of exact branch-and-bound algorithms for solving the QMST in Section 4. We propose three heuristic algorithms for this problem in Section 5 and provide computational results.

2. COMPLEXITY OF THE QMST PROBLEM

In this section, we relate the QMST problem to two well-known problems in combinatorial optimization: the Hamiltonian path problem and the quadratic assignment problem (QAP). We show that the QMST is at least as hard as the quadratic assignment problem and that it is NP-hard even with a simple quadratic cost structure.

Consider the class of QMST problems where the intercosts vanish for all nonadjacent pairs of edges so that $a_{ij} \neq 0$ only if i and j are adjacent edges. We refer to this subclass of QMST problems as the *adjacent-only QMST*, or AQMST for short.

Recall that a Hamiltonian path in a graph $G = (V, E)$ is a path that visits each vertex in V exactly once. Determining whether a given graph possesses a Hamiltonian path is NP-complete (see [12] or [16]). The following result shows that the Hamiltonian path problem can be formulated and solved as an AQMST. This result establishes that AQMST is NP-complete and shows QMST to be NP-hard by extension.

THEOREM 1: The Hamiltonian path problem polynomially transforms to the adjacent-only QMST.

PROOF: Consider an instance of the Hamiltonian path problem on the graph $G = (V, E)$. Define the AQMST on G by setting the intercosts a_{ij} equal to 1 if edges e_i and e_j are adjacent and zero, otherwise. Let $b_i = 0$ for all i in this AQMST.

Consider any solution x to the AQMST and let $T(x)$ denote the tree in G formed by the edges e_i with $x_i = 1$. We claim that $z(x) \geq 2(n-2)$ with equality holding if and only if $T(x)$ is a Hamiltonian path. To see this, let d_i be the degree of node v_i in the tree $T(x)$. There are $d_i(d_i - 1)$ ordered pairs of edges in $T(x)$ that are adjacent by virtue of sharing node v_i. Since $a_{jk} = 1$ for adjacent edges e_j and e_k, these pairs contribute exactly $d_i(d_i - 1)$ to the objective function. Summing the contributions over all nodes i, we see that

$$z(x) = \sum_{i=1}^{n} d_i(d_i - 1).$$

Since any tree has at least two nodes of degree 1, we can assume that $d_1 = d_n = 1$ in $T(x)$ with no loss of generality. Let $I = \{2, \ldots, n-1\}$ and $\tilde{d} = (d_2, \ldots, d_{n-1})$. $\tilde{d}$ must satisfy $\Sigma_{i \in I} d_i = 2(n-2)$ because $T(x)$ is a tree.

Now the function $\Sigma_{i \in I} d_i(d_i - 1)$ is strictly convex on the domain $\{\tilde{d} | \Sigma_{i \in I} d_i = 2(n-2), d_i \geq 1 \text{ for all } i\}$ and attains its unique minimum at $\tilde{d} = (2, \ldots, 2)$. This shows that $z(x) \geq 2(n-2)$ for any x corresponding to a tree and that equality holds if and only if all nodes in I have degree 2, that is, if x corresponds to a Hamiltonian path in G. Thus the existence of a Hamiltonian path can be verified by finding an optimal solution with value $2(n-2)$ for the AQMST.

To relate the QMST to the celebrated quadratic assignment problem (QAP), it is convenient to review the formulation of the QAP.

Consider n machines labeled $1, 2, \ldots, n$ that must be assigned to n locations denoted as $n+1, \ldots, 2n$. A set of costs (b_{ipjq}) are given to quantify the costs of interactions between pairs of machines. More precisely, b_{ipjq} is the cost of assigning machines i and j to locations p and q, respectively. Define the decision variables:

$$y_{ip} = \begin{cases} 1, & \text{if machine } i \text{ is assigned to location } p, \\ 0, & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, n$ and $p = n+1, \ldots, 2n$. The QAP can then be formulated as

$$\text{(QAP)} \qquad \text{minimize } w(y) = \sum_{i=1}^{n} \sum_{p=n+1}^{2n} \sum_{j=1}^{n} \sum_{q=n+1}^{2n} b_{ipjq} y_{ip} y_{iq} \tag{4}$$

$$\text{subject to } \sum_{p=n+1}^{2n} y_{ip} = 1, \qquad 1 \leq i \leq n, \tag{5}$$

$$\sum_{i=1}^{n} y_{ip} = 1, \qquad n+1 \leq p \leq 2n, \tag{6}$$

$$y_{ip} \in \{0, 1\}, \qquad \text{for all } i \text{ and } p. \tag{7}$$

If the costs (b_{ipjq}) can be expressed as $b_{ipjq} = f_{ij} d_{pq}$ for fixed matrices (f_{ij}) and (d_{pq}), the resulting QAP is called the Koopmans-Beckmann problem (see [19]).

The QAP has been studied extensively and useful surveys of the literature have appeared in [3] and [7]. Natural applications of this model arise in facility layout problems [18, 21]. Recently, Flood [8] has used this model to solve the weighted feedback arc set problem.

The complexity of the QAP has been established both theoretically and computationally. The Koopmans-Beckmann problem is known to be NP-complete [12] and Sahni and Gonzales [24] have shown that even the ε-approximate version of this problem remains NP-complete for any $\varepsilon > 0$. Computational tests have provided empirical evidence of the notorious difficulty of solving QAPs to optimality. In their survey, Finke, Burkard, and Rendl [7] indicate that the largest problem sizes solved to optimality have $n = 15$ machines. Indeed, the computational experience reported for exact algorithms for the QAP rarely exceed this limit [20, 15, 23]. Thus, even for QAPs of relatively small size, researchers have been forced to resort to heuristic techniques.

The next result shows that the QAP can be formulated as a QMST problem. Given the intractability of the QAP, this shows that the QMST is also a hard problem, independently of Theorem 1. However, the motivation for this transformation is to relate our work to the substantial literature on the QAP.

THEOREM 2: There is a polynomial transformation from the QAP to the QMST.

PROOF: Consider the bipartite graph G_0 joining vertices $\{1, \ldots, n\}$ and $\{n + 1, \ldots, 2n\}$ as shown in Figure 1(a). Any solution to the QAP provides an assignment that corresponds to n edges in G_0, no two of which share a common vertex.

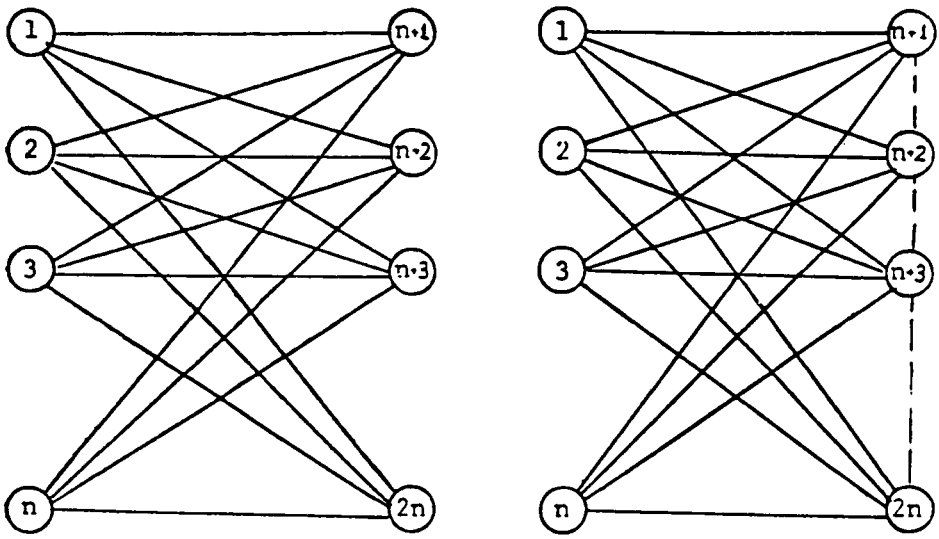

Figure 1. The graphs G_0 and $\bar{G}$.

Add $n - 1$ fictitious edges to join locations k and $k + 1$ for $n < k < 2n$ as shown by dashed lines in Figure 1(b). Denote the resulting graph by $\tilde{G}$ and define the QMST costs on this graph as follows:

(i) the intercost between any pair of nonadjacent edges (i, p), and (j, q) of G_0 equals b_{ipjq},
(ii) adjacent edges in G_0 have intercosts set to infinity,
(iii) any fictitious edge has an intercost of zero with all other edges.

Consider the subgraph of G_0 correspondong to a feasible solution to the QAP. Adding the $n - 1$ fictitious edges to this subgraph yields a tree in $\tilde{G}$. Under the preceding rules, the cost of this tree (in the QMST on $\tilde{G}$) agrees with the QAP objective value. Thus any solution to the QAP can be transformed to a solution for the QMST with equal value.

Conversely, take any solution to the QMST with a *finite* objective value. This corresponds to a tree in $\tilde{G}$ in which all nonfictitious edges are nonadjacent. Thus, there must be $(n - 1)$ fictitious edges and n nonfictitious ones, and the latter group defines a solution to the QAP with the same objective value as for the QMST. This completes the proof.

3. A LOWER BOUND FOR THE QMST

In this section, we develop an easily computable lower bound for the QMST. This bound specializes a bounding technique proposed in [2] for a wider class of 0, 1 programs to the QMST problem.

In what follows, we often use the fact that any spanning tree of a graph with n vertices has exactly $n - 1$ edges; that is,

$$\sum_{i=1}^{n} x_i = n - 1, \qquad \text{for any } x \in \mathscr{T}. \tag{8}$$

To develop the bound, introduce m parameters $\pi = (\pi_1, \pi_2, \ldots, \pi_m)$ and let

$$b_i(\pi) = b_i - (n - 2)\pi_i, \qquad 1 \le i \le m, \tag{9}$$

$$a_{ij}(\pi) = a_{ij} + \pi_j, \qquad 1 \le i, \ j \le m, \ \ i \ne j. \tag{10}$$

Define

$$f_i(\pi) = \min\left\{ b_i(\pi) + \sum_{\substack{j=1 \\ j \ne i}}^{m} a_{ij}(\pi)x_j \,|\, x_i = 1 \text{ and } x \in \mathscr{T} \right\} \tag{11}$$

for $1 \le i \le m$; and

$$\mathrm{PB}(\pi) = \min\left\{ \sum_{i=1}^{m} f_i(\pi)x_i \,|\, x \in \mathscr{T} \right\}. \tag{12}$$

Our next result demonstrates that $\mathrm{PB}(\pi)$ is a lower bound for the QMST. Note that the computation of $\mathrm{PB}(\pi)$ involves solving $m + 1$ minimum spanning tree (MST) problems. The MST defined by (11) has one edge fixed in the solution while all edges are available in (12).

THEOREM 3: $\mathrm{PB}(\pi)$ provides a lower bound for the QMST problem for any π.

PROOF: Since all x_i's are binary, $x_i^2 = x_i$. This allows us to write the objective function of the QMST in (2) as

$$z(x) = x^t A x,$$

where A is a matrix with diagonal entries $a_{ii} = b_i$ and off-diagonal entries a_{ij}. Let $\overline{A}$ be the corresponding matrix with entries defined by (9) and (10) so that

$$\overline{A} = A + Q,$$

where $Q = (q_{ij})$ is the matrix defined by

$$q_{ij} = \begin{cases} -(n-2)\pi_i, & \text{if } i = j, \\ \pi_j, & i \neq j. \end{cases}$$

Now, if $\overline{z}(x)$ is defined to equal $x^t \overline{A} x$, then

$$\overline{z}(x) = x^t \overline{A} x = x^t A x + x^t Q x = z(x),$$

since direct computation shows that $x^t Q x = 0$ for any $x \in \mathcal{T}$ by virtue of (8). Because $\overline{z}(x) = z(x)$, the original problem in (2) is equivalent to the new QMST

$$\min \left\{ \overline{z}(x) = \sum_{i=1}^{m} \left[b_i(x) + \sum_{\substack{j=1 \\ j \neq i}}^{m} a_{ij}(\pi) x_j \right] x_i \,|\, x \in \mathcal{T} \right\}. \tag{13}$$

The objective of (13) satisfies

$$\overline{z}(x) = \sum_{i=1}^{m} \left[b_i(\pi) + \sum_{\substack{j=1 \\ j \neq 1}}^{m} a_{ij}(\pi) x_j \right] x_i \geq \sum_{i=1}^{m} f_i(\pi) x_i \geq \mathrm{PB}(\pi)$$

by the definitions of $f_i(\pi)$ and $\mathrm{PB}(\pi)$ in (11) and (12). This completes the proof.

We call $\mathrm{PB}(\pi)$ the *parametrized lower bound* with parameters π. It is clearly desirable to have an efficient method for finding the best parameters π^* providing the largest lower bound:

$$\mathrm{PB}^* = \mathrm{PB}(\pi^*) = \max_{\pi} \mathrm{PB}(\pi). \tag{14}$$

The bound PB(π) developed directly in the preceding discussion can be related to a Lagrangian relaxation of a linear reformulation of the QMST. To see this, note that the problem in (2) can be reformulated as

$$\min \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} a_{ij} y_{ij} + \sum_{i=1}^{m} b_i x_i, \tag{15.1}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} y_{ij} = (n-1)x_j, \qquad j = 1, \ldots, m, \tag{15.2}$$

$$\sum_{j=1}^{m} y_{ij} = (n-1)x_i, \qquad i = 1, \ldots, m, \tag{15.3}$$

$$y_{ii} = x_i, \qquad i = 1, \ldots, m, \tag{15.4}$$

$$0 \leq y_{ij} \leq 1, \qquad \text{for all } i, j, \tag{15.5}$$

$$x \in \mathscr{T}. \tag{15.6}$$

The main advantage of this formulation is the elimination of the quadratic terms from the objective. Indeed, one can show that at optimality, the relation $y_{ij} = x_i x_j$ must hold (see [1, 10, 27]). Of course, (15.6) must be retained to ensure that the solution corresponds to a tree. Now let $y^{(i)} = (y_{i1}, \ldots, y_{im})$ for $i = 1, \ldots, m$ and append the zero m vector to $\mathscr{T}$: $\mathscr{T}_0 = \mathscr{T} \cup \{(0, \ldots, 0)\}$.

Suppose that we add the redundant constraints

$$y^{(i)} \in \mathscr{T}_0, \qquad i = 1, \ldots, m, \tag{15.7}$$

to the preceding formulation and relax the contraints in (15.2) via multipliers π_j. It is easy to see that the resulting Lagrangian problem is

$$\min \sum_{i=1}^{m} \sum_{\substack{j=1 \\ j \neq i}}^{m} a_{ij}(\pi) y_{ij} + \sum_{i=1}^{m} b_i(\pi) x_i,$$

$$\text{s.t.} \quad (15.3)\text{–}(15.7),$$

where $a_{ij}(\pi)$ and $b_i(\pi)$ are defined by (9) and (10). Closer inspection of this last problem shows that once x_i is fixed at 1, the vector $y^{(i)}$ can be determined by solving the minimum spanning tree problem in (11). Consequently, PB(π) in (12) is precisely the value of the preceding Lagrangian problem. A more general and detailed demonstration of this correspondence appear in [1].

Given this interpretation of PB(π), one can proceed to solve the Lagrangian dual problem (14) by a standard method such as subgradient optimization. However, Assad and Xu [2] show that a simple multiplier adjustment method can

also solve this problem effectively. We now give a brief account of this technique as applied to the QMST.

For a fixed π, let the *deviation* of $f = (f_1, \ldots, f_m)$ be defined as the gap between the largest and smallest components of f; that is

$$\phi(\pi) = \max_i\{f_i(\pi)\} - \min_i\{f_i(\pi)\}. \tag{16}$$

It can be shown that an ε-optimal solution to (14) can be characterized as one that has a small deviation. More precisely, if for some $\tilde{\pi}$, $\phi(\tilde{\pi}) \leq \varepsilon/(n-1)$, then $PB^* - PB(\tilde{\pi}) \leq \varepsilon$. This shows that an optimal solution to (14) is identified if the deviation of f can be reduced to zero for some π. The multiplier adjustment method that follows seeks such a solution by iteratively reducing the distance between the maximal and minimal components of f. The procedure is called the *leveling algorithm* in [2] to highlight this effect.

The Leveling Procedure

Step 0: Select a tolerance limit $\delta > 0$ for the stopping criterion and set $t = 1$, $\pi^1 = (0, \ldots, 0)$.

Step 1: Compute the cost coefficients b_i and a_{ij} from (9) and (10) with $\pi = \pi^t$ and solve the $m + 1$ MST problems

$$f_i(\pi^t) = \min\left\{b_i(\pi^t) + \sum_{j \neq i} a_{ij}(\pi^t)x_j \mid x_i = 1,\ x \in \mathcal{T}\right\}, \tag{16.1}$$

for $i = 1, \ldots, m$ and

$$\mathrm{PB}_t = \mathrm{PB}(\pi^t) = \min\left\{\sum_{i=1}^{m} f_i(\pi^t)x_i \mid x \in \mathcal{T}\right\}. \tag{16.2}$$

Step 2: Compute

$$\phi_t = \max_i\{f_i(\pi^t)\} - \min_i\{f_i(\pi^t)\}. \tag{16.3}$$

If $\phi_t < \delta$, stop. Otherwise update π by setting

$$\pi_i^{t+1} = \pi_i^t + \frac{1}{n-1} f_i(\pi^t), \qquad \text{for all } i, \tag{17.1}$$

let $t := t + 1$, and return to Step 1.

It is easy to see that the effect of the multiplier adjustment in Step 2 is to update the cost coefficients b and A according to

$$b_i(\pi^{t+1}) = b_i(\pi^t) - \frac{n-2}{n-1} f_i(\pi^t), \qquad \text{for all } i, \tag{17.2}$$

$$a_{ij}(\pi^{t+1}) = a_{ij}(\pi^t) + \frac{1}{n-1} f_i(\pi^t), \qquad \text{for all } i \text{ and } j \neq i. \tag{17.3}$$

The leveling procedure produces a sequence of lower bounds $\{PB_t\}$ for the QMST. A desirable property of this method that is not shared by subgradient techniques is monotonicity: One can show that $\{PB_t\}$ is nondecreasing in t. Moreover $\{\phi_t\}$ is a nonincreasing sequence and if this sequence converges to zero, then a solution to the Lagrangian dual in (14) is obtained [2].

To investigate the speed of convergence of ϕ_t to 0, one can track the *reduction ratio* $RR_t = \phi_t/\phi_{t-1}$. Table 1 presents the results of a computational study of this ratio for QMST problems of various sizes n (number of nodes). The leveling algorithm was applied to 20 different randomly generated QMST problems for each n. The mean and standard deviation of RR_t was calculated across all iterations of the 20 sample problems. These are listed in Table 1 under the columns μ and σ. The parameter δ for the stopping criterion in Step 2 was set at $\delta = 1$. The average number of iterations taken by the algorithm is also listed for each n. Note that the reduction ratio RR_t stays consistently in the range 0.3–0.45. In fact, no value of RR_t larger than 0.5 was observed in any of the test problems. These results show that the convergence of $\{\phi_t\}$ to zero is rapid and does not deteriorate as n increases.

The generation of the random problems used for Table 1 was performed as follows: For each problem, a matrix $A = (a_{ij})$ of costs was generated by sampling all entries above or on the diagonal from the discrete uniform distribution between 0 and 20. All costs are thus integers between 0 and 20. The matrix A is then completed by symmetry. The computational work for this article was performed on the University of Maryland Univac 1100/80 computer.

The leveling procedure clearly aims for simplicity and relatively low computational burden. More sophisticated bounding techniques readily come to mind. For instance, replacing (15.2) by the constraints $y_{ij} = y_{ji}$ for $i \neq j$ also results in an equivalent linear reformulation. These constraints may be relaxed in a Lagrangian way to obtain a bound. However, the increase in the number of multipliers complicates the subgradient optimization procedure. This was indicated by some preliminary tests we performed. In fact, a more disaggregate formulation used by Frieze and Yadegar [10] for the QAP did not produce superior Lagrangian bounds when compared to the leveling algorithm [2]. For

Table 1. Convergence of the lower bounds obtained by the leveling algorithm.

Problem size	Reduction ratio		
n	μ	σ	No. of iterations
6	0.399	0.051	5.26
7	0.407	0.067	5.33
8	0.416	0.043	5.36
9	0.398	0.038	5.30
10	0.366	0.041	5.13
11	0.356	0.028	5.06
12	0.355	0.025	5.03
13	0.332	0.012	5.00
14	0.315	0.027	5.00
15	0.307	0.027	5.00

this reason, we did not investigate bounds for such disaggregate formulations of the QMST.

4. EXACT ALGORITHMS

In this section, we present two exact branch-and-bound algorithms for the QMST. The first algorithm is designed for the general QMST and relies upon the bound described in Section 3. The second algorithm focuses on the adjacent-only QMST and uses a bounding technique specialized for the structure of the AQMST.

Branch-and-Bound Algorithm I

This is a depth-first search procedure with branching performed on single edges. Every node of the branch-and-bound tree corresponds to a partitioning of the edges into three sets, those fixed into the tree, those excluded from the tree, and free edges whose status remains to be determined. Denoting the index sets of these edges by U, V, and F, we have

$$x_i = \begin{cases} 1, & \text{if } i \in U, \\ 0, & \text{if } i \in V, \end{cases} \tag{18}$$

and $F = \{1, \ldots, m\} - (U \cup V)$.

The objective function of (2) can be expressed in terms of the free variables $\{x_i, i \in F\}$ alone by direct computation:

$$z(x) = C + \sum_{i \in F} \hat{b}_i x_i + \sum_{i \in F} \sum_{\substack{j \in F \\ j \neq i}} a_{ij} x_i x_j, \tag{19}$$

where C is a constant term

$$C = \sum_{i \in U} \left[b_i + \sum_{\substack{j \in U \\ j \neq i}} a_{ij} \right],$$

and

$$\hat{b}_i = b_i + \sum_{j \in U} (a_{ij} + a_{ji}).$$

The expression in (19) defines a new quadratic objective over the binary variables $\{x_i | i \in F\}$. Since any solution x must be a tree and must also satisfy (18), the search is restricted to $\mathcal{T}(U, V)$—the subset of $\mathcal{T}$ where (18) is satisfied. The leveling procedure remains valid for this restricted problem provided $\mathcal{T}(U, V)$ replaces $\mathcal{T}$ in (16.1) and (16.2). Also, since

$$\sum_{i \in F} x_i = n - |U| - 1$$

for any $x \in \mathcal{T}(U, V)$, n must be replaced with $n - |U|$ in (17.1)–(17.3).

The branch-and-bound procedure can now be outlined as follows. At each node of the search tree, the lower bound PB_t is computed from the leveling algorithm. An upper bound is also computed by evaluating the cost of the tree obtained when solving (16.2) for $t = 1$ (that is, the first time Step 1 is entered). This upper bound replaces the current best upper bound CUB (the incumbent) if it has a lower cost. If $PB_t \geq CUB$, we say that the node is *fathomed* and backtrack to evaluate another node. Otherwise we branch on the edge e_k where $k = \text{argmin}_{i \in F}\{f_i(\pi^1)\}$ and create two descendants with x_k fixed at 0 and 1. The motivation for this branching rule is that because $\pi^1 = 0$, the MST that can be constructed with x_k fixed at 1 has a lower cost than the MST corresponding to fixing any other x_i at 1. In a depth-first approach, therefore, one may hope that the node with $x_k = 1$ will produce a better upper bound while the node with $x_k = 0$ may produce a larger lower bound.

To test the performance of this algorithm, it was applied to test problems with size (n) ranging from 6 to 12. All graphs are complete graphs with n vertices and $m = n(n - 1)/2$ edges. The off-diagonal elements of the intercost matrix for each problem are integers that are randomly generated and distributed uniformly over [0, 20]. The diagonal elements are also random integers, distributed uniformly over [0, 100].

Table 2 summarizes the results of the tests. The branch-and-bound algorithm was run with two choices of the lower bound: PB_1 and PB_2. These are shown under the columns $t = 1$ and $t = 2$. Using PB_2 provides a better lower bound but increases the computation time as the leveling procedure must run through two iterations rather than one. As expected, the improved bounds reduce the size of the branch-and-bound tree.

Branch-and-Bound Algorithm II (for the AQMST)

The special structure of the AQMST allows us to evaluate its lower bound more efficiently. Algorithm II takes advantage of the specialized bound developed below.

Let the sets U, V, and F be associated with the current node as before. Let A_i denote the index set of all edges adjacent to e_i and put $F_i = F \cap A_i$. Using

Table 2. Computational results for the branch-and-bound algorithm I.

	Number of nodes		Depth of tree		CPU (sec)		Optimal value
n	$t = 1$	$t = 2$	$t = 1$	$t = 2$	$t = 1$	$t = 2$	z^*
6	19	11	5	4	0.11	0.21	207
7	63	23	7	4	0.51	0.92	326
8	131	39	9	5	1.66	2.71	412
9	577	113	14	9	9.93	11.63	533
10	2397	359	19	15	57.28	58.17	638
11	6289	1069	22	16	208.86	252.30	740
12	54413	7391	32	26	2214.08	2284.37	917

(19), we can write the objective function as

$$z(x) = C + \sum_{i \in F} \left[\hat{b}_i + \sum_{j \in F_i} a_{ij} x_j \right] x_i, \tag{20}$$

where the sum over j can be restricted to F_i since only edges adjacent to i need be considered.

If we assume that all a_{ij}'s are nonnegative, then the term within brackets in (20) can be bounded below by

$$g_i = \begin{cases} \hat{b}_i + \min\{a_{ij} | j \in F_i\}, & \text{if } U \cap A_i = \emptyset, \\ \hat{b}_i, & \text{otherwise,} \end{cases} \tag{21}$$

when x_i equals 1. Note that (21) merely states that when no edges adjacent to e_i are already fixed in the tree, then an interaction term a_{ij} will be contributed to (20) for some edge e_j adjacent to e_i, since e_i must be adjacent to at least one other edge in the tree. A lower bound on (20) is therefore provided by

$$\text{LB} = C + \min \left\{ \sum_{i \in F} g_i x_i | x \text{ satisfies (18) and } x \in \mathscr{T} \right\}. \tag{22}$$

This lower bound does not require the values of f_i. This is an advantage over the bound (17). Whereas a total of $m + 1$ minimal spanning trees must be solved to obtain PB, LB requires the solution of only one MST in (22). A similar, but more complicated expression can be derived when some a_{ij}'s are negative [27].

The branch-and-bound algorithm for the AQMST uses the lower bound in (22) and costs out the tree that solves (22) to obtain an upper bound at that node of the search tree. The branching rule is to select the edge with the smallest g_i value.

Table 3 compares Algorithms I and II on AQMST test problems with sizes (n) ranging from 6 to 15. The intercosts for adjacent edges are randomly chosen

Table 3. Computational results for Algorithms I and II on the AQMST.

	Number of nodes			Depth of tree			CPU (sec)			Optimal value
n	Ia	Ib	II	Ia	Ib	II	Ia	Ib	II	z^*
6	21	11	21	4	3	4	0.12	0.23	0.04	142
7	35	13	69	7	5	8	0.35	0.57	0.18	203
8	69	35	69	7	5	7	0.98	2.48	0.25	212
9	159	81	165	9	8	9	3.13	8.52	0.73	228
10	145	73	149	8	8	8	4.69	13.94	0.87	227
11	373	185	377	11	10	11	15.42	49.69	2.55	237
12	705	359	735	11	10	11	43.20	146.70	5.90	261
13	365	179	365	11	11	11	29.07	105.33	3.75	239
14	2415	1263	2441	15	15	15	233.52	971.74	28.02	286
15	—	—	9337	—	—	17	—	—	125.26	—

integers from a uniform distribution on [0, 20]. Columns (Ia) and (Ib) of this table correspond to the choices $t = 1$ or $t = 2$ in Algorithm I.

The results show Algorithm II to be clearly superior to Algorithm I in terms of computation time. This confirms the value of devising an algorithm that takes advantage of the special structure of the AQMST.

5. HEURISTIC ALGORITHMS

The computational results of the preceding section show that solving the QMST to optimality is highly time consuming even for problems of moderate size. In this section, we turn to approximate algorithms and present three heuristics for the QMST that are easy to implement.

Heuristic Algorithm H1 (Average Contribution Method)

Let us focus on a single edge e_k and rewrite the objective $z(x)$ of (2) so as to separate all terms containing index k; that is,

$$z(x) = \left[b_k + \sum_{j \neq k} (a_{jk} + a_{jk}) x_j \right] x_k + z_2(x),$$

where the term $z_2(x)$ does not involve x_k. Suppose that e_k goes into the solution tree. We estimate the contribution of e_k to the objective value as follows: Of the $(m - 1)$ terms in the summation within brackets, all but $(n - 2)$ terms must be zero. So the average contribution of e_k to the objective is estimated to be

$$p_k = b_k + \frac{(n-2)}{m-1} \sum_{j \neq k} (a_{jk} + a_{kj}), \qquad 1 \leq k \leq m. \tag{23}$$

After evaluating the average contributions p_k for all k, we solve the MST:

$$P = \min \left\{ \sum_{k=1}^{m} p_k x_k | x \in T \right\}, \tag{24}$$

to obtain the heuristic solution.

Algorithm H1 requires the evaluation of p_k for $k = 1, \ldots, m$ and calls for an MST to be solved in (24). Its overall complexity is thus $O(m^2)$. Since the number of coefficients in A is also $O(m^2)$, heuristic H1 is as fast as possible.

Heuristic Algorithm H2 (Sequential Fixing Method)

In heuristic H1, the average contributions of all edges are estimated independently. However, fixing certain edges into the tree should affect the estimated contribution of the remaining edges. This motivates the approach taken in heuristic H2. Upon each evaluation of the edge contributions, this approach fixes *only one* edge into the growing spanning tree. The average contributions of the

remaining edges are then reassessed prior to fixing another edge. In this way, the edges in the spanning tree are determined sequentially.

To define the average contribution at an intermediate step of the algorithm, let U, V, and F denote the index sets of the fixed, excluded, and free edges as in (18). Given U, V must include all edges that close a cycle with edges in U. Focus on a free edge e_k ($k \in F$). Based on (19), we can collect all occurrences of x_k in $z(x)$ into the expression

$$z_k(x) = \left[\hat{b}_k + \sum_{\substack{j \in F \\ j \neq k}} (a_{jk} + a_{kj})x_j\right] x_k$$

and write the objective as $z(x) = z_k(x) + \tilde{z}(x)$, where $\tilde{z}(x)$ involves no x_k terms. The summation within brackets has $NF = |F| - 1$ terms of which all but $NT = n - 2 - |U|$ must vanish. The average contribution of e_k is thus

$$q_k = \hat{b}_k + \frac{NT}{NF} \sum_{\substack{j \in F \\ j \neq k}} (a_{jk} + a_{kj}). \tag{25}$$

Note that this expression coincides with p_k in (23) when F contains all indices ($U = V = \emptyset$).

For ease of notation, define the set functions $h_k(\cdot)$ on subsets of $\{1, \ldots, m\}$ as follows

$$h_k(S) = \sum_{j \in S} (a_{jk} + a_{kj}), \tag{26}$$

so that q_k can be expressed as

$$q_k = b_k + h_k(U) + \frac{NT}{NF} h_k(F - \{k\}) \tag{27}$$

using (25) and the definition of $\hat{b}_k$. We now state the sequential fixing heuristic.

Heuristic H2

Step 0 Initialize $U = \emptyset$, $F = \{1, \ldots, m\}$, $NF = m - 1$, and $NT = n - 1$. Compute q_k from (27) for $k = 1, \ldots, m$.

Step 1 Find r such that $q_r = \min\{q_k | k \in F\}$ and set $U := U \cup \{r\}$, $NT := NT - 1$. If $NT < 0$, stop.

Step 2 Remove from F edge e_r and any edge e_i that closes a cycle with the edges in U. Update $NF = |F| - 1$ and the weights q_k for all $k \in F$ according to (27). Return to Step 1.

This procedure can be viewed as a greedy heuristic that selects the minimum weight edge e_r (according to the weights q_k) and fixes it into the solution. The weights are updated after each selection and the corresponding purge of F. The main computational burden of the procedure is at step 0 where $O(m^2)$ additions are required to compute the q_k's. In step 2, one can use data structures similar

to those in [4, 28] to keep track of the components of the subgraph defined by edges in U. The addition of a new edge e_r to U corresponds to a merge of two components. Moreover, by recording the components to which the end points of each edge in F belong, one can identify the edges that close cycles in the current forest as those with both end points in the same component.

The updating of q_k is straightforward once the set of edges D deleted from F in Step 2 is found. The expressions in (27) are updated as follows:

$$h_k(F - \{k\}) := h_k(F - D - \{k\}) = h_k(F - \{k\}) - h_k(D)$$

and

$$h_k(U) := h_k(U \cup \{r\}) = h_k(U) + a_{rk} + a_{kr}.$$

In this way, the updating requires $O(m)$ operations while the identification of D takes no more than $O(n^2)$ operations in the worst case. Since Step 1 is executed $n - 1$ times, the overall complexity of the heuristic is $O(m^2)$ if $m \sim O(n^2)$.

Computational Results

Tables 4 and 5 compare the behavior of the heuristic algorithms on both the QMST and the AQMST. The first seven problems of Table 4 ($n \leq 12$) are identical to those of Table 2. The remaining problems are generated randomly in the same manner. The same procedure is used to generate random AQMST problems in Tables 3 and 5, although different streams of random numbers are utilized.

In addition to H1 and H2, we have included columns for H3. This heuristic merely records the best solution encountered in a fixed number of iterations of

Table 4. Three heuristics for the QMST.

	Heuristic value			CPU time (sec)			Optimal
n	H1	H2	H3	H1	H2	H3	value
6	211	207*	207*	0.0026	0.0044	0.0788	207
7	326*	335	326*	0.0044	0.0080	0.1468	326
8	468	412*	412*	0.0070	0.0126	0.2468	412
9	727	586	564*	0.0108	0.0198	0.3426	533
10	709	638*	666	0.0156	0.0294	0.5156	638
11	900	740*	761	0.0222	0.0416	0.8076	740
12	1017	937*	937*	0.0308	0.0580	1.1856	917
13	1397	1087	1085*	0.0418	0.0784	1.7306	—
14	1818	1348	1262*	0.0552	0.1048	2.2522	—
15	1781	1621*	1652	0.0716	0.1346	2.8422	—
16	1912	1578*	1634	0.0920	0.1720	4.1054	—
17	2409	1910	1891*	0.1164	0.2168	5.0016	—
18	2920	2227*	2303	0.1452	0.2690	6.5746	—
19	2887	2453*	2535	0.1794	0.3296	8.8288	—
20	3439	2688	2711	0.2192	0.4006	10.1036	—

*Indicates the best among three heuristic values.

Table 5. Three heuristics for the AQMST.

	Heuristic value			CPU time (sec)			Optimal
n	H1	H2	H3	H1	H2	H3	value
6	131*	131*	131	0.0026	0.0044	0.0618	131
7	204	170*	170*	0.0044	0.0078	0.1750	170
8	185	176*	176*	0.0070	0.0128	0.2468	176
9	396	276*	283	0.0104	0.0196	0.3300	266
10	353	242*	280	0.0154	0.0294	0.5242	242
11	474	308*	327	0.0218	0.0418	0.7884	295
12	428	305*	318*	0.0304	0.0570	1.2096	299
13	415	312	278*	0.0410	0.0772	1.7250	278
14	429	317*	335	0.0544	0.1022	2.1432	295
15	439	333	326*	0.0708	0.1342	3.5484	324

*Indicates the best among three heuristic values.

Algorithm I with $t = 1$. The procedure H3 thus corresponds to an early termination of the branch-and-bound search procedure. We allowed $2n$ iterations of Algorithm I on a problem of size n.

The computational results show H2 and H3 to be superior to H1 in terms of accuracy. Moreover, H2 seems to slightly outperform H3 on QMST's. Note that the computation times are smallest for H1, followed by H2, and then H3. Thus although H2 and H3 are roughly comparable in terms of their solution quality, H3 takes significantly more time to execute.

6. CONCLUSION AND FUTURE DIRECTIONS

This article introduced the quadratic minimum spanning tree problem (QMST), which involves finding the minimum spanning tree under a quadratic cost structure on the edges. We showed that this problem is NP-complete and demonstrated that it includes the well-known quadratic assignment problem (QAP) as special case.

A simple branch-and-bound algorithm based on a lower bounding procedure for the QMST was used to solve problems with up to 15 nodes to optimality. Given that QMST is at least as hard as the QAP, one must resort to heuristics for larger problems. Two different heuristics were developed and tested. Both are simple to implement and exhibit encouraging performance.

The present work constitutes a first attempt at devising solution techniques for the QMST. Since this problem had not been studied before, it is hard to compare our results with competing techniques. Naturally, one may view the QMST as a general quadratic binary program and investigate how techniques for this class of problems can be applied to the QMST. While this direction is worthwhile to pursue, one must bear in mind that the tree constraints of the QMST restrict the feasible set significantly. Moreover, the success and power of solution techniques for the QMST are probably correlated with their effectiveness for the QAP. Future work can first concentrate on methods that have had encouraging results on the QAP. In particular, the use of novel heuristic techniques such as simulated annealing and tabu search can be pursued [3, 6, 25].

Finally, the QMST is a problem with large data requirements: For a network with m edges, m^2 intercosts must be specified. In practical settings, one may expect the intercosts to exhibit some special structure, as does the AQMST. Another direction for future work is to focus on QMST problems with simplified cost structures.

ACKNOWLEDGMENTS

The authors are indebted to Professor Bruce Golden of the University of Maryland for bringing the stochastic spanning tree problem to our attention. We also thank an anonymous associated editor of the journal whose detailed comments strengthened the article significantly.

REFERENCES

[1] Assad, A.A., "Alternative Lagrangean Relaxations for Quadratic Integer Programs," in *Proceedings of the 1985 S.E. TIMS Meeting*, R.D.J. Hammesfahr, Ed., 1985, pp. 171–173.

[2] Assad, A. A., and Xu, W., "On Lower Bounds for a Class of 0-1 Programs," *Operations Research Letters*, **4**, 175–180 (1985).

[3] Burkard, R. E., "Quadratic Assignment Problems," *European Journal of Operational Research*, **15**, 283–289 (1984).

[4] Camerini, P. M., Galbiati, G., and Maffioli, F., "Algorithms for Finding Optimum Trees: Description, Use, and Evaluation," *Annals of Operations Research*, **13**, 265–397 (1988).

[5] Charnes, A., and Cooper, W. W., "Deterministic Equivalents for Optimizing and Satisficing Under Chance Constraints," *Operations Research*, **11**, 18–39 (1963).

[6] Connolly, D.T., "An Improved Annealing Scheme for the QAP," *European Journal of Operational Research*, **46**, 93–100 (1990).

[7] Finke, G., Burkard, R.E., and Rendl, F., "Quadratic Assignment Problems," *Annals of Discrete Mathematics*, **31**, 61–82 (1987).

[8] Flood, M., "Exact and Heuristic Algorithms for the Weighted Feedback Arc Set Problem: A Special Case of the Skew-Symmetric Quadratic Assignment Problem," *Networks*, **20**, 1–23 (1990).

[9] Fredman, M.L., and Tarjan, R.E., "Fibonacci Heaps and their Uses in Network Optimization Algorithms," *Proceedings of the 25th Annual IEEE Symposium on FOCS*, 1984, pp. 338–346.

[10] Frieze, A., and Yadegar, J., "On the Quadratic Assignment Problem," *Discrete Applied Mathematics*, **5**, 89–98 (1983).

[11] Gabow, H.P., Galil, Z., Spencer, T., and Tarjan, R.E., "Efficient Algorithms for Finding Minimum Spanning Trees in Undirected and Directed Graphs," *Combinatorica*, **6**, 109–122 (1986).

[12] Garey, M.R., and Johnson, D.S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, San Franciso, 1979.

[13] Ishii, H., Shiode, S., and Nishida, T., "Chance-Constrained Spanning Tree Problem," *Journal of the Operations Research Society of Japan*, **24**, 147–157 (1981).

[14] Ishii, H., Shiode, S., Nishida, T., and Namasuya, Y., "Stochastic Spanning Tree Problem," *Annals of Discrete Mathematics*, **3**, 263–273 (1981).

[15] Kaku, B.K., and Thompson, G.L., "An Exact Algorithm for the General Quadratic Assignment Problem," *European Journal of Operational Research*, **23**, 382–390 (1986).

[16] Karp, R.M., "Reducibility Among Combinatorial Problems," in *Complexity of Computer Computations*, R.E. Miller and J.W. Thatcher, Eds., Plenum, New York, 1972, pp. 85–103.

[17] Kruskal, J.B., "On the Shortest Spanning Subtree of a Graph and the Traveling

Salesman Problem," *Proceedings of the American Mathematical Society*, **7**, 48–50 (1956).
[18] Kusiak, A., and Heragu, S.S. "The Facility Layout Problem," *European Journal of Operational Research*, **29**, 229–251 (1987).
[19] Lawler, E.L., "The Quadratic Assignment Problem," *Management Science*, **19**, 586–590 (1962).
[20] Liggett, R.S., "The Quadratic Assignment Problem: An Experimental Evaluation of Solution Strategies," *Management Science*, **27**, 442–458 (1981).
[21] Love, R.F., Morris, J.G., and Wesolowsky, G.O., *Facilities Location: Models and Methods*, North-Holland, New York, 1988.
[22] Prim, R.C., "Shortest Connection Networks and Some Generalizations," *Bell Systems Technical Journal*, **36**, 1389–1401 (1957).
[23] Roucairol, C., "A Parallel Branch and Bound Algorithm for the Quadratic Assignment Problem," *Discrete Applied Mathematics*, **18**, 211–225 (1987).
[24] Sahni, S., and Gonzales, T., "P-Complete Approximation Problems," *Journal of the ACM*, **23**, 555–565 (1976).
[25] Shorin-Karpov, J., "Tabu Search Applied to the Quadratic Assignment Problem," *ORSA Journal on Computing*, **2**, 33–45 (1990).
[26] Tarjan, R.E., *Data Structures and Network Algorithms*, SIAM, Philadelphia, PA, 1983.
[27] Xu, W., "The Quadratic Minimum Spanning Tree Problem and Related Topics," unpublished Ph.D. dissertation, Department of Mathematics, University of Maryland, College Park, MD, 1984.
[28] Yao, A., "An $O(|E| \log \log |V|)$ Algorithm for Finding Minimum Spanning Trees," *Information Processing Letters*, **4**, 21–23 (1975).

Manuscript received 12/18/86
Manuscript revised 3/4/91
Manuscript accepted 9/10/91