



ELSEVIER

Theory and Methodology

A branch-and-bound algorithm  
for the mini-max spanning forest problem

Takeo Yamada<sup>1</sup>, Hideo Takahashi<sup>2</sup>, Seiji Kataoka

Department of Computer Science, The National Defense Academy, Yokosuka, Kanagawa 239, Japan

Received 23 January 1995; accepted 1 May 1996

Abstract

The mini-max spanning forest problem requires to find a spanning forest of an undirected graph that minimizes the maximum of the costs of constituent trees. In a previous work we proved this problem NP-hard. In the current paper we present three lower bounds for this problem and develop a branch-and-bound algorithm to solve the problem exactly. The algorithm is implemented and numerical experiments are conducted on a series of test problems. The experiments compare the performances of the proposed bounds and search strategies in the algorithm as well as investigate the effects of instance characteristics on the behavior of the algorithm. Also, extension of the problem to the case of more than two root vertices as well as to the problem of determining the root locations are discussed. © 1997 Elsevier Science B.V.

Keywords: Spanning forest; Mini-max problem; Branch-and-bound method; Graph theory

1. Introduction

In a previous work [9], we formulated the mini-max spanning forest problem (MMSFP) in the following way. Let  $G = (V, E)$  be an undirected graph [2] where  $V$  is the set of vertices and  $E \subseteq V \times V$  is the set of edges. Each edge  $e \in E$  has an associated integer cost  $w(e) > 0$ . We assume  $G$  is connected and simple, i.e., there exist neither self-loops nor multiple edges. A tree is a connected acyclic subgraph of  $G$ , and more generally, an acyclic subgraph of  $G$  is said to be a forest. A forest consists of a set of mutually disjoint trees. For a tree  $T$ , its cost  $w(T)$  is defined as the sum of the costs of its constituent edges. A tree is a spanning tree of  $G$  if it covers all the vertices of  $G$ ,

and a spanning forest is similarly defined. Given a set of root vertices  $U := \{u_1, u_2, \dots, u_r\} \subseteq V$ , a  $U$ -rooted spanning forest  $F$  is a spanning forest of  $G$  consisting of  $r$  disjoint trees  $T_1, T_2, \dots, T_r$  such that  $u_i$  is a vertex of  $T_i$  ( $i = 1, 2, \dots, r$ ). The value of such a forest is defined by

$$w(F) := \max_{1 \leq i \leq r} \{w(T_i)\}. \tag{1}$$

MMSFP requires to find the  $U$ -rooted spanning forest  $F^*$  that minimizes  $w(F)$  over all  $U$ -rooted spanning forests of  $G$ .

Such a problem arises when we plan to construct a communication network to serve a set of stations from more than one broadcast centers using unreliable communication lines [1]. The reliability of every edge is assumed to be given as  $p(e)$ ,  $e \in E$ . Each station must be connected, directly or indirectly, to either one of the centers. A communication network

<sup>1</sup> Corresponding author. Email: yamada@cs.nda.ac.jp.

<sup>2</sup> Currently with Air Defense Control Wing, Air Self-Defense Force, Japan.

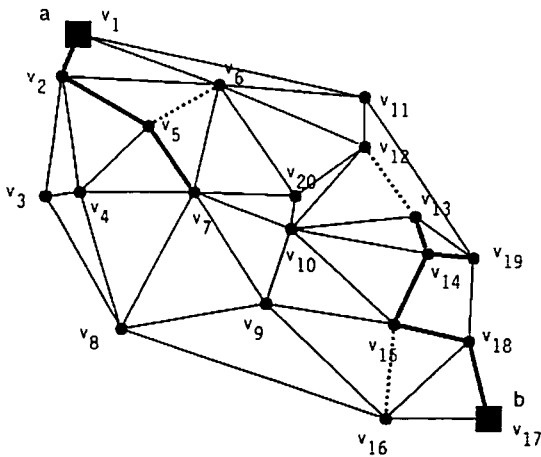


Fig. 1. Subproblem  $P$ . Bold and dotted lines stand for fixed subtrees and forbidden edges, respectively.

in such a situation may be represented by a spanning forest rooted at the set of the centers. For a spanning forest  $F = (T_1, T_2, \dots, T_r)$ , let  $P(T_i) := \prod_{e \in T_i} p(e)$  denote the reliability of the *subsystem* served by the  $i$ th center. We wish *all*  $P(T_i)$ 's be as large as possible, while *unbalanced* forests with some of the  $P(T_i)$ 's extremely smaller than others are not preferable. A reasonable way of achieving this objective is to maximize the minimum of  $\{P(T_i) \mid i = 1, 2, \dots, r\}$ . Then, putting  $w(e) := -\ln p(e)$  we immediately obtain MMSFP.

Alternatively, we may consider an election for  $r$  seats in a country consisting of  $r$  cities and several villages. Such a country can be represented as an undirected graph, where vertices and edges stand for the communities and their adjacency relations respectively. The problem is to divide the graph into  $r$  connected subgraphs, each of which representing a constituency consisting of a city and some villages, such that the total populations of the districts are as balanced as possible. Although in this problem population induces weights on vertices rather than on edges, this can be easily transformed into the standard MMSFP.

MMSFP differs from other network location/allocation problems [4] such as determining the location of fire stations and/or assigning areas or streets to these stations by the form of the objective function (1). Indeed, in such problems the objective function is usually the sum (or the maximum) of the distances

of each vertex to the assigned (fire) station, while in MMSFP it is defined in terms of the total costs of trees. Except for the well studied *minimum spanning tree* (MST) problem, which is the special case of MMSFP with  $r = 1$  and can be solved by the polynomial time algorithms of Kruskal [5] or Prim [8], problems with this type of objective functions have rarely been explored.

In [9], we proved NP-hardness [3] of MMSFP with  $r \geq 2$ . In that paper we also developed a heuristic algorithm to solve MMSFP. The purpose of this paper is to supplement our previous work by presenting an exact algorithm to solve MMSFP. The algorithm is based on the *branch-and-bound* method, and we present three kinds of lower bounds for this purpose. Computational results on various types of randomly generated test problems are reported, comparing the performances of the lower bounds and branching strategies. Also, we discuss extension of MMSFP to the case of more than two root vertices as well as to the problem of determining the root locations.

## 2. A framework for an exact algorithm

In the current paper we primarily consider MMSFP with two root vertices  $U = \{a, b\}$ . Extension to the case of three or more root vertices is only briefly discussed. To solve MMSFP within the framework of the branch-and-bound method, we define a *subproblem*  $P$  of MMSFP in terms of a triplet of the sets of edges  $(T_{Pa}, T_{Pb}, X_P)$ . Here  $T_{Pa}$  and  $T_{Pb}$  are mutually disjoint sets of edges that constitutes subtrees rooted at  $a$  and  $b$  respectively, and  $X_P$  is another set of edges which is disjoint with  $T_{Pa} \cup T_{Pb}$ . The tree on side  $a$  itself is denoted by  $T_{Pa}$  if it is not confusing, and this is referred to as the *fixed subtree* on side  $a$ . Similarly  $T_{Pb}$  is the fixed subtree on side  $b$ , and  $X_P$  is said to be the set of *forbidden edges*. A spanning forest  $F = (T_a, T_b)$  is said to be a *solution* to  $P$ , or shortly  *$P$ -admissible*, if it includes  $T_{Pa} \cup T_{Pb}$  and does not include  $X_P$ , i.e.,  $F$  is  $P$ -admissible if  $T_{Pa} \subseteq T_a$ ,  $T_{Pb} \subseteq T_b$  and  $(T_{Pa} \cup T_{Pb}) \cap X_P = \emptyset$ .

**Example 1.** In the graph of Fig. 1 with details given in Table 1, two root vertices ( $a = v_1$ , and  $b = v_{17}$ ) are shown with squares. Bold lines stand for the fixed subtrees  $T_{Pa} = \{e_1, e_6, e_{14}\}$  and  $T_{Pb} =$

Table 1  
Data for the graph of Fig. 1

edge	$v_1$	$v_2$	cost	edge	$v_1$	$v_2$	cost	edge	$v_1$	$v_2$	cost
$e_1$	$v_1$	$v_2$	53	$e_2$	$v_1$	$v_6$	180	$e_3$	$v_1$	$v_{11}$	353
$e_4$	$v_2$	$v_3$	146	$e_5$	$v_2$	$v_4$	141	$e_6$	$v_2$	$v_5$	120
$e_7$	$v_2$	$v_6$	190	$e_8$	$v_3$	$v_4$	40	$e_9$	$v_3$	$v_8$	183
$e_{10}$	$v_4$	$v_5$	116	$e_{11}$	$v_4$	$v_7$	140	$e_{12}$	$v_4$	$v_8$	172
$e_{13}$	$v_5$	$v_6$	98	$e_{14}$	$v_5$	$v_7$	97	$e_{15}$	$v_6$	$v_7$	133
$e_{16}$	$v_6$	$v_{11}$	175	$e_{17}$	$v_6$	$v_{12}$	190	$e_{18}$	$v_6$	$v_{20}$	162
$e_{19}$	$v_7$	$v_8$	187	$e_{20}$	$v_7$	$v_9$	159	$e_{21}$	$v_7$	$v_{10}$	123
$e_{22}$	$v_7$	$v_{20}$	120	$e_{23}$	$v_8$	$v_9$	177	$e_{24}$	$v_8$	$v_{16}$	338
$e_{25}$	$v_9$	$v_{10}$	94	$e_{26}$	$v_9$	$v_{15}$	157	$e_{27}$	$v_9$	$v_{16}$	201
$e_{28}$	$v_{10}$	$v_{12}$	134	$e_{29}$	$v_{10}$	$v_{13}$	150	$e_{30}$	$v_{10}$	$v_{14}$	167
$e_{31}$	$v_{10}$	$v_{15}$	169	$e_{32}$	$v_{10}$	$v_{20}$	40	$e_{33}$	$v_{11}$	$v_{12}$	60
$e_{34}$	$v_{11}$	$v_{19}$	234	$e_{35}$	$v_{12}$	$v_{13}$	104	$e_{36}$	$v_{12}$	$v_{20}$	104
$e_{37}$	$v_{13}$	$v_{14}$	47	$e_{38}$	$v_{13}$	$v_{19}$	86	$e_{39}$	$v_{14}$	$v_{15}$	93
$e_{40}$	$v_{14}$	$v_{19}$	55	$e_{41}$	$v_{15}$	$v_{16}$	115	$e_{42}$	$v_{15}$	$v_{18}$	92
$e_{43}$	$v_{16}$	$v_{17}$	125	$e_{44}$	$v_{16}$	$v_{18}$	137	$e_{45}$	$v_{17}$	$v_{18}$	98
$e_{46}$	$v_{18}$	$v_{19}$	100								

$\{e_{37}, e_{39}, e_{40}, e_{42}, e_{45}\}$ , while the dotted lines represent the forbidden edges  $X_P = \{e_{13}, e_{35}, e_{41}\}$ .

$P$  is *feasible* if it has at least one  $P$ -admissible spanning forest. Otherwise, it is *infeasible*. Subproblem  $P$  requires to find a  $P$ -admissible spanning forest  $F$  such that  $w(F)$  is minimized. Clearly, the original MMSFP is identical to the subproblem defined by  $(\phi, \phi, \phi)$ . In subproblem  $P$  if  $(T_{Pa}, T_{Pb})$  is a spanning forest,  $P$  is a *terminal subproblem* to which the forest is obviously the solution. Otherwise, if  $E \setminus (T_{Pa} \cup T_{Pb} \cup X_P)$  is non-empty we pick up an edge  $e$  from this set such that  $e$  is incident either to  $T_{Pa}$  or  $T_{Pb}$ , and define two *child problems* of  $P$  in the following way. One child problem  $P \oplus \{e\}$  is obtained when we augment  $T_{Pa}$  or  $T_{Pb}$  by adding  $e$  to whichever subtree is incident to this edge. For another child  $P \ominus \{e\}$  we augment  $X_P$  by adding  $e$  to this set. Clearly  $P$  is solved if both of the children are solved. To construct a branch-and-bound algorithm we need to develop lower bounding procedures, which will be the topic of the next section.

Still, there remains a choice as to which edge we pick up first from the set of possible edges. Although any qualified edge suffices, the performance of the resulting algorithm may heavily depend on this choice. We try the strategy of taking the qualified edge of minimum (or maximum) weight first. This is referred to as the *min-first (max-first)* strategy.

### 3. Lower bounds

Let  $G_0$  denote the graph obtained from  $G$  by identifying its root vertices  $a$  and  $b$ . Then, every spanning forest of  $G$  corresponds to a spanning tree in  $G_0$  in an one-to-one way. For a subproblem  $P$  defined by  $(T_{Pa}, T_{Pb}, X_P)$ , a tree in  $G_0$  is  $P$ -admissible if it includes all edges of  $T_{Pa} \cup T_{Pb}$  and does not include any edge of  $X_P$ . It is easy to find the minimum  $P$ -admissible spanning tree in  $G_0$ , provided that  $P$  is feasible. A minor modification of Prim's or Kruskal's method suffices for this purpose. Let  $T_P^0$  be this spanning tree. By  $F_P^* = (T_{Pa}^*, T_{Pb}^*)$  we denote an optimal solution to  $P$ , with the objective value  $w^*(P) := w(F_P^*)$ . Then, the following inequality is straightforward.

$$\begin{aligned}
 w^*(P) &\equiv \max\{w(T_{Pa}^*), w(T_{Pb}^*)\} \\
 &\geq \lfloor \{w(T_{Pa}^*) + w(T_{Pb}^*)\} / 2 \rfloor \\
 &\geq \lfloor \{w(T_P^0)\} / 2 \rfloor.
 \end{aligned}$$

Thus,

$$\text{LB}_1(P) := \lfloor \{w(T_P^0)\} / 2 \rfloor \tag{2}$$

gives a lower bound to  $w^*(P)$ .

Note that if the spanning forest  $F_P^0 = (T_{Pa}^0, T_{Pb}^0)$  corresponding to  $T_P^0$  attains the lower bound, i.e., if  $w(F_P^0) = \text{LB}_1(P)$ ,  $F_P^0$  is necessarily optimal to  $P$ .

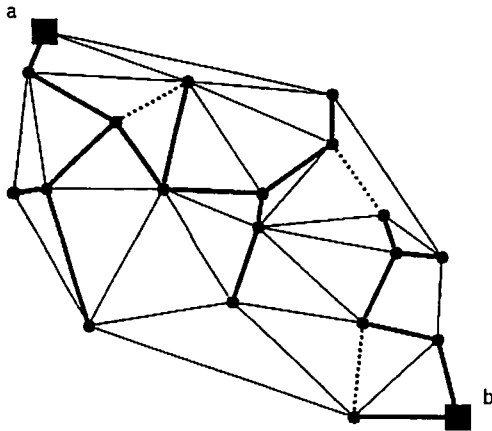


Fig. 2. Spanning forest  $F_p^0$ . Bold and dotted lines stand for fixed subtrees and forbidden edges, respectively.

**Example 2.** For the subproblem  $P$  of Fig. 1, the spanning forest  $F_p^0$  is as shown in Fig. 2. From this we obtain  $LB_1(P) = \lfloor (1149 + 510)/2 \rfloor = 830$ , while  $w(F_p^0) = 1149$ .

To improve the performance of the branch-and-bound algorithm to be developed, we introduce two more lower bounds for subproblem  $P$ . To this end, let  $G_{Pa}$  denote the auxiliary graph obtained from  $G$  by removing all edges of  $X_P \cup T_{Pb}$  as well as those incident to  $T_{Pb}$ . Similarly we define  $G_{Pb}$  with respect to  $b$ . Let  $\mathcal{F}_{Pa}(k)$  denote the set of all forests in  $G_{Pa}$  consisting of exactly  $k$  edges that altogether cover  $T_{Pa}$ . We introduce the *cumulative cost function* of the minimum spanning tree in  $G_{Pa}$  by

$$w_{Pa}^0(k) := \begin{cases} \min\{w^0(F) \mid F \in \mathcal{F}_{Pa}(k)\} & \text{if } \mathcal{F}_{Pa}(k) \neq \emptyset, \\ \infty & \text{otherwise,} \end{cases} \quad (3)$$

where  $w^0(F)$  denotes the sum of the edge costs over  $F$ . Note that this can be easily calculated for  $k = 1, 2, \dots, |V| - 1$  by applying Kruskal's method. Also we note that  $w_{Pa}^0(k) = \infty$  for  $k < |T_{Pa}|$ , and therefore

$$w_{Pa}^0(k) \geq w(T_{Pa}), \quad k = 1, 2, \dots, |V| - 1. \quad (4)$$

Similarly, we calculate  $w_{Pb}^0(k)$  for  $k = 1, 2, \dots, |V| - 1$ . Let

$$\Delta := \{(k_a, k_b) \mid k_a + k_b = |V| - 2, k_a \geq 0, k_b \geq 0\}$$

and define

$$LB_2(P) := \min_{(k_a, k_b) \in \Delta} [\max\{w_{Pa}^0(k_a), w_{Pb}^0(k_b)\}]. \quad (5)$$

Then this gives another lower bound by the following.

**Proposition 3.**  $w^*(P) \geq LB_2(P)$ .

**Proof.** For an optimal spanning forest  $F^* = (T_{Pa}^*, T_{Pb}^*)$ , let us denote  $k_a^* = |T_{Pa}^*|$ ,  $k_b^* = |T_{Pb}^*|$ . We note that  $(k_a^*, k_b^*) \in \Delta$ . From (3) we have

$$w(T_{Pa}^*) \geq w_{Pa}^0(k_a^*), \quad w(T_{Pb}^*) \geq w_{Pb}^0(k_b^*),$$

which implies

$$\begin{aligned} w^*(P) &= \max\{w(T_{Pa}^*), w(T_{Pb}^*)\} \\ &\geq \max\{w_{Pa}^0(k_a^*), w_{Pb}^0(k_b^*)\} \\ &\geq LB_2(P). \quad \square \end{aligned}$$

Let the right-hand side of (5) be minimized at  $(k_a^\dagger, k_b^\dagger)$ , and  $(F_a^\dagger, F_b^\dagger)$  denotes the pair of solutions obtained in the minimization of (3) for these values of  $k$ . If  $F^\dagger = (F_a^\dagger, F_b^\dagger)$  is a spanning forest, we have  $w(F^\dagger) = LB_2(P)$ , implying that  $F^\dagger$  is necessarily optimal to  $P$ .

From (4) the following is also a lower bound.

$$LB_3(P) := \max\{w(T_{Pa}), w(T_{Pb})\}. \quad (6)$$

Although  $LB_3$  is always weaker than  $LB_2$ , i.e.,  $LB_3(P) \leq LB_2(P)$ , this may prove useful since it can be obtained much easier than  $LB_2$  or  $LB_1$ .

**Example 4.** For the subproblem  $P$  of Fig. 1, the auxiliary graphs  $G_{Pa}$  and  $G_{Pb}$  are as shown in Fig. 3. From this we calculate  $w_{Pa}^0(k_a)$  and  $w_{Pb}^0(k_b)$ , which are shown in Fig. 4 with  $k_b = |V| - 2 - k_a$ . From the figure we obtain  $LB_2(P) = \max\{w_{Pa}^0(8), w_{Pb}^0(10)\} = \max\{608, 723\} = 723$ . Also,  $LB_3(P) = \max\{270, 385\} = 385$ .

#### 4. A branch-and-bound algorithm

We now construct an exact algorithm to solve MMSFP. Although the algorithm is essentially a straightforward application of the standard branch-and-bound method to MMSFP, a few comments are

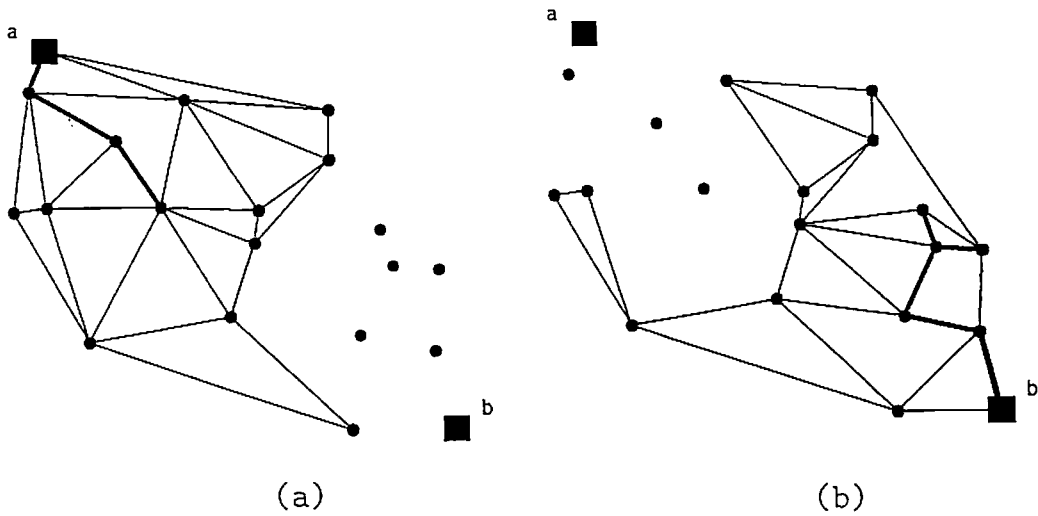


Fig. 3. Auxiliary graphs: (a)  $G_{Pa}$  and (b)  $G_{Pb}$ .

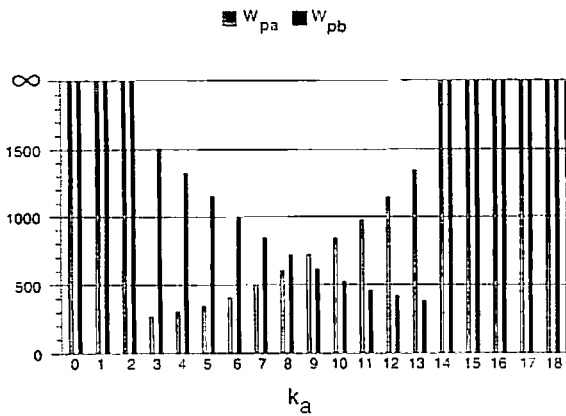


Fig. 4. Cumulative cost functions  $w_{Pa}(k_a)$  and  $w_{Pb}(k_b)$ , where  $k_b = |V| - 2 - k_a$ .

due on the details of the algorithm. Let  $\mathcal{P}$  denote the set of active subproblems, which is initially  $\mathcal{P} = \{\text{The subproblem } (\phi, \phi, \phi)\}$ . At the beginning, we set the *incumbent solution* as  $(w^*, F^*) := (\infty, \phi)$ , where  $F^*$  stands for the best spanning forest obtained so far and  $w^* = w(F^*)$  is the value of that forest. We pick up a subproblem  $P$  from  $\mathcal{P}$ , and examine this in the following way. By  $P$ , if it is not confusing, we also mean the subgraph obtained from  $G$  by removing all forbidden edges.

First, we check whether  $P$  is *feasible*. Due to the forbidden edges,  $P$  may not have any  $U$ -rooted span-

ning forest at all. This happens if, after identifying two root vertices, the graph  $P$  has two or more connected components. In this case, the subproblem is simply terminated. Next, we examine whether  $P$  is a terminal subproblem, in which case the incumbent is updated if necessary and we terminate this subproblem. Finally, if  $P$  is *split* in the sense that it consists of exactly two connected components with one root vertex each, we readily obtain an optimal solution to this subproblem by finding the minimum spanning trees independently within each component. Again we update the incumbent if necessary and terminate this subproblem.

If none of these cases applies, using the heuristic method in our previous work [9] we calculate an approximate solution  $\bar{F}(P)$  with corresponding upper bound  $UB(P) = w(\bar{F}(P))$ . We update the incumbent if necessary. Then we evaluate the lower bound by  $LB(P) := \max\{LB_1(P), LB_2(P), LB_3(P)\}$ . If  $LB(P) \geq w^*$ , the subproblem is terminated. Otherwise, if  $F_P^0$  attains  $LB_1$ , again we update the incumbent if necessary and terminate this subproblem. Alternatively, if we obtain the spanning forest  $F^\dagger$  that attains  $LB_2$ , the subproblem is analogously treated. If such a spanning forest is not obtained, we pick up an edge  $e \in E$  as stated in Section 2 to produce two child subproblems  $P \oplus \{e\}$  and  $P \ominus \{e\}$ . These are added to  $\mathcal{P}$ , and we repeat the whole process all over again until  $\mathcal{P}$  is empty.

Table 2  
Branch-and-bound results for  $P_{20,46}$

roots	#forests	LB	UB	exact		
				$w^*$	#subprobs	time
1, 20	$1.543 \times 10^{11}$	799	855	855	7595	10.4
2, 19	$1.632 \times 10^{11}$	799	855	848	15627	21.1
3, 18	$2.110 \times 10^{11}$	799	855	848	17431	23.8
4, 17	$2.476 \times 10^{11}$	799	855	848	17615	24.1
5, 16	$1.613 \times 10^{11}$	799	855	848	17611	23.7
6, 15	$1.242 \times 10^{11}$	799	855	848	16497	21.9
7, 14	$1.238 \times 10^{11}$	799	855	848	14441	18.6
8, 13	$1.494 \times 10^{11}$	799	855	852	39267	52.5
9, 12	$1.182 \times 10^{11}$	807	852	848	7803	10.4
10, 11	$1.123 \times 10^{11}$	807	852	852	7369	9.9
average	$1.565 \times 10^{11}$	800.6	854.2	849.5	16125.6	21.6

A few remarks are due at this point. Calculating an approximate solution may be skipped at some or all subproblems. From computational efficiency it might be better to perform this procedure at every fixed number of subproblems. The performance of the resulting algorithm depends on the accuracy of the approximation as well as its computational easiness. In evaluating the lower bounds, we may calculate some or all of  $LB_i$  ( $i = 1, 2, 3$ ). The more lower bounds we calculate, the stronger bounds we obtain in the expense of longer computational time. Finally, in picking up a subproblem from  $\mathcal{P}$ , *depth-first search* is most popular in the branch-and-bound framework. Our algorithm is equipped with this and *breadth-first search* strategy as well.

## 5. Numerical experience

The algorithm of the previous section has been implemented in C language and a series of test problems have been solved to optimality on DEC3000/400 workstation.

### 5.1. A numerical example

First we have solved MMSFP for the graph of Fig. 1, which is referred to as  $P_{20,46}$  in the sequel, with different pairs of root vertices. We have used  $LB_1 + LB_2$  for the lower bound, the depth-first search for picking up a subproblem from  $\mathcal{P}$ , and the min-first strategy in

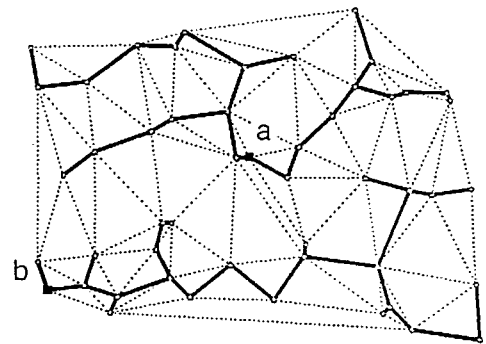


Fig. 5. Optimal spanning forest for  $P_{50 \times 127}$ .

choosing the branching edge. Table 2 summarizes the result of this computation, where LB and UB represent, respectively, the lower and upper bounds evaluated for the initial subproblem  $(\phi, \phi, \phi)$ ,  $w^*$  the exact value of the solution, and #subprob the number of subproblems generated in the branch-and-bound process. These are shown for each pair of root vertices together with the respective computation time in seconds.

The total numbers of spanning forests are also given as a reference. This has been calculated using the Kirchhoff's matrix formula (see, e.g., [2]) to enumerate spanning trees. Fig. 5 illustrates an example of an optimal solution for the graph  $P_{50 \times 127}$  consisting of 50 vertices and 127 edges. In obtaining this solution, the algorithm generated 16,467 subproblems in 66.96 seconds.

Table 3  
Min-first vs. max-first strategy on  $P_{20,46}$

roots	min-first		max-first	
	#subprobs	time	#subprobs	time
1, 20	7595	10.4	26667	32.2
2, 19	15627	21.1	93151	116.0
3, 18	17431	23.9	45471	56.8
4, 17	17615	24.2	109559	140.1
5, 16	17611	23.7	96581	122.1
6, 15	16497	21.9	77957	97.9
7, 14	14441	18.6	92463	113.4
8, 13	39267	52.5	39733	51.5
9, 12	7803	10.4	6029	7.8
10, 11	7369	9.9	5155	6.5
average	16125.6	21.6	59276.6	74.4

Table 4  
The effects of lower bounds for  $P_{20,46}$

roots	LB <sub>1</sub>		LB <sub>1</sub> + LB <sub>2</sub>		LB <sub>1</sub> + LB <sub>3</sub>	
	#subprobs	time	#subprobs	time	#subprobs	time
1, 20	104745	67.5	7595	10.4	15781	11.7
2, 19	87427	58.2	15627	21.1	29127	20.6
3, 18	86943	58.5	17431	23.9	32393	23.2
4, 17	90251	61.0	17615	24.2	32301	23.3
5, 16	79331	53.3	17611	23.7	30583	21.8
6, 15	70033	46.2	16497	21.9	29529	21.2
7, 14	48875	31.1	14441	18.6	26415	18.3
8, 13	478251	323.7	39267	52.5	53758	40.9
9, 12	33687	22.4	7803	10.4	18257	12.7
10, 11	21521	14.1	7369	9.9	16941	11.8
average	110106.4	73.6	16125.6	21.6	28541.4	20.5

Table 3 compares the min-first strategy against the max-first in choosing a branching edge. Either from the numbers of subproblems generated and the CPU time required in solving the problem, the min-first strategy usually outperformed the max-first strategy.

A comparison is made in Table 4 using different sets of lower bounds. Three cases tested are: (i) LB<sub>1</sub> only, (ii) LB<sub>1</sub> + LB<sub>2</sub>, and (iii) LB<sub>1</sub> + LB<sub>3</sub>. Although the CPU time is almost comparable for (ii) and (iii), in the subsequent experiments we use LB<sub>1</sub> + LB<sub>2</sub> since in that case we usually have the least number of subproblems.

## 5.2. Numerical experiments

Next, we have conducted a series of experiments for the following sample problems:

- (1) Planar graph  $P_{n,m}$ .
- (2) Complete graph  $K_n$ .
- (3) Complete bipartite graph  $K_{n,n}$ .

Here  $n$  denotes the number of vertices and  $m$  is the number of edges. The vertices of these graphs are taken randomly on the plane of  $[0, 800] \times [0, 600]$ , and their edge costs are the rounded euclidean distances.

Table 5 gives the result of experiments for this set of problems. Here all entries of the table are the average of ten runs for each problem with different pairs of (randomly taken) root vertices.

As a variation of these test problems, we define  $RP_{n,m}$ ,  $RK_n$ , and  $RK_{n,n}$  by replacing the edge costs of respective graphs with random numbers in  $[1, 1000]$ . Table 6 summarizes the result of experiments for this class of subproblems. Comparing Tables 5 and 6, instances with random edge costs appear to be much easier to solve. This may be explained by the fact that in euclidean instances the edge costs are distributed in much narrower interval than in the random instances. For example, the standard deviation of the edge costs is 65.51 for  $P_{20,46}$ , as opposed to 289.14 for  $RP_{20,46}$ . This results in the existence of much larger number of near optimal solutions, making the bounding procedure less effective.

**6. MMSFP with more than two root vertices**

In this paper we have developed a branch-and-bound algorithm to solve MMSFP with two root vertices. Now we briefly discuss its extension to the case of three or more roots. This is rather straightforward, except for the following. We are given the set of root vertices  $U := \{u_1, u_2, \dots, u_r\}$ . A subproblem is  $P = (T_{P1}, T_{P2}, \dots, T_{Pr}, X_P)$ , where  $T_{Pi}$  is a subtree rooted at  $u_i$  ( $i = 1, 2, \dots, r$ ), and  $X_P$  is the set of forbidden edges. These are mutually disjoint. First, we have

$$w^*(P) \geq \lfloor \{w(T_P^0)\} / r \rfloor, \tag{7}$$

where  $T_P^0$  is the minimum  $P$ -admissible spanning tree on  $G_0$ , which is obtained from  $G$  by identifying all the root vertices. Thus the right-hand side of the above inequality gives  $LB_1$  for this case.

Next, we define  $G_{Pi}$  as the auxiliary graph obtained from  $G$  by removing all edges of  $X_P \cup (\cup_{j \neq i} T_{Pj})$  as well as those incident to  $(\cup_{j \neq i} T_{Pj})$  ( $i = 1, 2, \dots, r$ ). Let  $\mathcal{F}_{Pi}(k)$  denote the set of all forests in  $G_{Pi}$  consisting of exactly  $k$  edges that altogether cover  $T_{Pi}$ , and calculate

$$w_{Pi}^0(k_i) := \begin{cases} \min\{w^0(F) \mid F \in \mathcal{F}_{Pi}(k_i)\} & \text{if } \mathcal{F}_{Pi}(k_i) \neq \phi, \\ \infty & \text{otherwise,} \end{cases} \tag{8}$$

for  $i = 1, 2, \dots, r$ . Then we have the second lower bound

$$LB_2(P) := \min_{(k_1, \dots, k_r) \in \Delta} [\max\{w_{P1}^0(k_1), \dots, w_{Pr}^0(k_r)\}], \tag{9}$$

where

$$\begin{aligned} \Delta &:= \{(k_1, \dots, k_r) \mid k_1 + \dots + k_r \\ &= |V| - r, k_1 \geq 0, \dots, k_r \geq 0\}. \end{aligned}$$

We note that  $LB_2$  can be calculated by the following dynamic programming recursion. Let

$$v(j, b) := \begin{cases} \min_{(k_j, \dots, k_r) \in \Delta(j, b)} [\max\{w_{Pj}^0(k_j), \dots, w_{Pr}^0(k_r)\}] & \text{if } b \geq 0, \\ 0 & \text{otherwise,} \end{cases}$$

where

$$\begin{aligned} \Delta(j, b) &:= \{(k_j, \dots, k_r) \mid k_j + \dots + k_r = b, \\ &k_j \geq 0, \dots, k_r \geq 0\}. \end{aligned}$$

Then, we have

$$v(j, b) = \min_{0 \leq k_j \leq b} [\max\{w_{Pj}^0(k_j), v(j+1, b-k_j)\}], \tag{10}$$

$$j = 1, 2, \dots, r; b = 0, 1, \dots, |V| - r.$$

This can be solved backward beginning with the terminal condition

$$v(r+1, b) = 0, \quad b = 0, 1, \dots, |V| - r$$

and we obtain

$$LB_2(P) = v(1, |V| - r). \tag{11}$$

As in Section 3, we also have the third lower bound

$$LB_3(P) := \max\{w(T_{P1}), \dots, w(T_{Pr})\}. \tag{12}$$

**7. Root vertices location problem**

In the previous sections, the root vertices have been assumed to be given and fixed. However, in the communication network planning we may have to determine the locations of the broadcast centers *before*

Table 5  
Results of experiments (euclidean distance)

problem	LB	UB	exact		
			$w^*$	#subprobs	time
$P_{10,18}$	561.3	685.0	670.9	242.6	0.1
$P_{20,46}$	800.6	854.4	849.5	16125.6	21.6
$P_{30,74}$	1238.1	1341.6	1295.9	1010057.2	2382.1
$P_{40,98}$	1367.0	1478.1	1431.8	13707115.4	46071.4
$P_{50,127}$	1630.5	1684.8	1657.5	2651957.4	11434.6
$K_{10}$	597.0	718.3	697.5	1553.6	1.7
$K_{20}$	973.1	1121.9	1070.2	590880.0	2914.9
$K_{30}$	1218.2	1318.4	1294.7	1619908.2	20398.5
$K_{40}$	1513.9	1623.6	1577.1	1008649.6	25041.8
$K_{5,5}$	1873.3	2095.1	1966.8	217.6	0.1
$K_{10,10}$	3230.6	3464.4	3377.9	59967.0	163.4
$K_{15,15}$	5368.5	5518.0	5448.7	1787684.8	11662.6
$K_{20,20}$	6397.2	6527.4	6478.9	2597846.4	30886.0

Table 6  
Results of experiments (random distance)

problem	LB	UB	exact		
			$w^*$	#subprobs	time
$RP_{10,18}$	1679.8	2234.3	2186.8	386.4	0.2
$RP_{20,46}$	2360.8	2740.8	2642.1	14902.0	20.4
$RP_{30,74}$	2978.0	3440.9	3193.4	113756.4	258.3
$RP_{40,98}$	4547.2	4999.9	4903.1	4285503.8	13314.7
$RP_{50,127}$	4970.6	5461.7	5167.6	250056.2	1119.5
$RK_{10}$	483.5	618.0	573.4	214.0	0.2
$RK_{20}$	406.0	496.0	453.9	5237.0	26.3
$RK_{30}$	485.6	535.0	512.1	5149.8	65.0
$RK_{40}$	535.6	583.6	559.4	99697.0	2432.8
$RK_{5,5}$	596.0	734.1	720.7	44.6	0.0
$RK_{10,10}$	836.9	925.9	898.0	1195.0	3.1
$RK_{15,15}$	835.6	988.8	894.7	101991.0	662.7
$RK_{20,20}$	1023.7	1129.8	1059.8	528078.8	6485.1

we solve MMSFP. In this section we extend MMSFP to include the problem of determining the root locations in the following way. Let  $c_i$  denote the *site cost* of a root placed at vertex  $i \in V$ . In the communication network example, this may be regarded as the *unreliability* of the broadcast center placed at this vertex. Let us take  $U = (u_1, u_2, \dots, u_r) \subseteq V$  as a set of root vertices, and consider a  $U$ -rooted spanning forest  $F = (T_1, T_2, \dots, T_r)$ . The total cost of the  $i$ th subsystem is given by the site cost  $c_{u_i}$  plus the tree cost  $w(T_i)$ . Analogous to MMSFP, we wish all these be as

small as possible. Therefore, we evaluate this forest by

$$w(U, F) = \max_{1 \leq i \leq r} \{c_{u_i} + w(T_i)\}. \quad (13)$$

The problem is to find a set of root vertices  $U$  and a spanning forest  $F$  rooted thereupon such that  $w(U, F)$  is minimized. This problem is denoted as MMSFP/L.

To solve MMSFP/L, let us augment graph  $G$  by adding a set of *virtual roots*  $S = \{s_1, s_2, \dots, s_r\}$  and a set of edges  $E_S := S \times V$ . Let the cost of edge  $(s_i, j) \in E_S$  be  $M + c_j$ , where  $M$  is a very large number ( $M \approx \infty$ ). The resulting graph is denoted as  $\tilde{G}$ . Now we

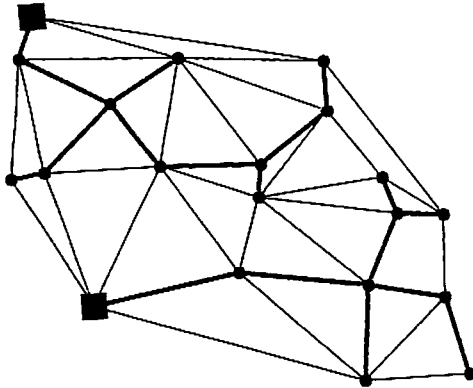


Fig. 6. Solution to MMSFP/L.

solve MMSFP with  $S$  as the set of root vertices, and let  $\tilde{F} = (\tilde{T}_1, \tilde{T}_2, \dots, \tilde{T}_r)$  denote an optimal solution. Then we obtain the following.

**Lemma 5.** Each  $\tilde{T}_i$  includes at most one edge from  $E_S$ .

**Proof.** If some  $\tilde{T}_i$  includes more than one edge from  $E_S$ , we have  $w(\tilde{F}) \geq 2M$ . However, by connecting the minimum spanning tree of  $G$  to  $s_1$  through edge  $(s_1, 1)$ , we obtain a spanning forest of  $\tilde{G}$  with the associated value  $M + c_1 + w_{MST} < 2M$ , where  $w_{MST}$  denotes the cost of the minimum spanning tree on  $G$ . This is a contradiction.  $\square$

If each  $\tilde{T}_i$  includes exactly one edge  $(s_i, u_i)$  from  $E_S$  ( $i = 1, 2, \dots, r$ ), we have a set of  $r$  distinct vertices  $U^* = (u_1, u_2, \dots, u_r) \subseteq V$ . At the same time, by natural correspondence  $\tilde{F}$  induces a  $U^*$ -rooted spanning forest  $F^*$ . Then, we obtain the following.

**Proposition 6.** If each  $\tilde{T}_i$  includes exactly one edge from  $E_S$  ( $i = 1, 2, \dots, r$ ), the above obtained  $U^*$  and  $F^*$  is optimal to MMSFP/L.

**Proof.** Obvious.  $\square$

For the case of two roots ( $r = 2$ ), we have a sufficient condition for Proposition 6. Let  $a$  be such that  $c_a = \min_{i \in V} c_i$ . For  $i \in V$ ,  $w_{MST}^{(i)}$  denotes the cost of the minimum spanning tree on  $G^{(i)}$ , which is obtained

from  $G$  by removing vertex  $i$  and all incident edges. Then, we have the following.

**Proposition 7.** For  $r = 2$  if there exists a vertex  $b \in V \setminus \{a\}$  such that  $w_{MST}^{(b)} \leq w_{MST}$  and  $c_b - c_a < w_{MST}$ , each  $\tilde{T}_i$  includes exactly one edge from  $E_S$  ( $i = 1, 2$ ).

**Proof.** If only one of  $\tilde{T}_1$  and  $\tilde{T}_2$  includes an edge from  $E_S$ , clearly the minimum of such a forest is  $w(\tilde{F}) = M + c_a + w_{MST}$ . On the other hand, let  $T_1$  and  $T_2$  be the minimum spanning trees that cover  $\{s_1\} \cup V \setminus \{b\}$  and  $\{s_2, b\}$  respectively. For this forest  $F = (T_1, T_2)$  we have

$$w(F) = \max\{M + c_a + w_{MST}^{(b)}, M + c_b\} < w(\tilde{F}).$$

This is a contradiction, which completes the proof.  $\square$

**Example 8.** For the graph of Example 1, if  $c_i \equiv i$  for all  $i \in V$  the assumption of Proposition 7 is satisfied with  $a = 1$  and  $b = 17$ , where we have  $w_{MST} = 1718$  and  $w_{MST}^{(b)} = 1620$ . Solving MMSFP on  $\tilde{G}$  we obtain optimal root locations and the corresponding spanning forest as shown in Fig. 6 with  $w' = \max\{1 + 348, 8 + 834\} = 848$ .

### 8. Conclusion

We have presented three lower bounds for MMSFP, developed a branch-and-bound algorithm to this problem, and solved a series of small test problems. To solve larger sized problems, we certainly need more improved bounding procedures as well as more sophisticated programming techniques. Another prospective approach to the exact solution of MMSFP would be the *branch-and-cut* method [7], which is based on the mathematical analysis of the polyhedral structure of the problem.

Nevertheless, we can easily expect practical applications whose sizes are far beyond the reach of such an exact algorithm. Especially, this applies to problems with more than two roots and/or root-vertices location problems. To such problems the so-called *meta-heuristic* algorithms [6] such as simulated annealing, tabu search or genetic algorithms might prove useful in obtaining satisfactory solutions within reasonable CPU time. These are left for future study.

## Acknowledgements

The authors are grateful to anonymous referees for their helpful comments.

## References

- [1] Ahuja, R.K., Magnanti, T.L., and Orlin, J.B. (1993), *Network Flows: Theory, Algorithms, and Applications*, Prentice-Hall.
- [2] Chen, W.K. (1976), *Applied Graph Theory: Graphs and Electrical Networks*, North-Holland.
- [3] Garey, M.R., and Johnson, D.S. (1978), *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman.
- [4] Handler, G.Y., and Mirchandani, P.B. (1979), *Location on Networks: Theory and Algorithms*, MIT Press.
- [5] Kruskal, J.B. (1956), "On the shortest spanning tree of a graph and the travelling salesman problem", *Proceedings of the American Mathematical Society* 7, 48–50.
- [6] Osman, J.H., and Kelly, J.P., eds. (1996), *Meta-Heuristics: Theory and Applications*, Kluwer.
- [7] Padberg, M.W., and Rinaldi, G. (1991), "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems", *SIAM Review* 33, 60–100.
- [8] Prim, R.C. (1957), "Shortest connection networks and some generalizations", *Bell Systems Technical Journal* 36, 1389–1401.
- [9] Yamada, T., Takahashi, H., and Kataoka, S., "A heuristic algorithm for the mini-max spanning forest problem", *European J. Operational Research* (to appear).