Comput. & Ops. Res. Vol. 10, No. 12, pp. 125–138, 1985
Printed in the U.S.A.

0305-0548/85 $3.00 + .00
Pergamon Press Ltd.

METHODS FOR A NETWORK DESIGN PROBLEM IN SOLAR POWER SYSTEMS

J. R. BIRGE†
The University of Michigan, Ann Arbor, MI 48109, U.S.A.

and

V. MALYSHKO‡
Byelorussian State University Minsk 22080, U.S.S.R.

(*Received March* 1983; *in revised form November* 1983)

Scope and Purpose—The purpose of this paper is to present solution procedures for a network problem in solar power system design. The problem is to minimize the cost of connecting the heliostat components of this system with a central computer. The problem is significant because the length of the cable connections may be considerable if no optimization is performed[2]. The paper formulates this task as a modification of a common network problem. The formulation may also generalize to other network systems[4]. The paper then presents methods for this general problem along with an analysis of their worst case performance. Computational results are provided to indicate how the algorithms might perform on a actual problem.

Abstract—We consider the problem of minimizing cable connections between a central computer and a field of heliostats in the design of solar power systems. This practical task can be modeled as a p-median problem with additional constraints in a weighted graph. We compare an exact branch-and-bound method with two approximate algorithms. For the latter two methods, estimations of time complexity and accuracy are presented. Computational results are shown which should be useful in the design of such large-scale power systems.

1. INTRODUCTION

A terrestrial solar thermal system consisting of a field of heliostats used to reflect sunlight to a central tower receiver is considered. The receiver contains a liquid which is heated to produce steam which in turn drives a turbine as in a conventional steam generating plant. Pilot systems have been designed[2, 12] and built in many countries[3, 23] and have been studied for the optimal design of heliostats and configurations [15, 20].

The heliostats must be under constant control in order to maintain their reflective accuracy in directing sunlight to the receiver. Besides this function, such control makes the following operations possible:

initial powering each morning according to given solar radiation intensity;

following the Sun's movement during the day;

switching off of the system whenever solar radiation falls below a minimum level for a given length of time;

connecting and disconnecting separate groups of heliostats to regulate the amount of power required; and so on.

In the earliest plants[9], each heliostat was independently steered. In large systems, this cost becomes prohibitive and some central computer-based control is necessary[5]. With

†John R. Birge is Assistant Professor in the Department of Industrial and Operations Engineering at The University of Michigan. He received his PhD. in Operations Research from Department of Operations Research at Stanford University. His research interests include public sector modeling, combinatorial optimization, scheduling, and stochastic programming. His publications have appeared in *Mathematical Programming, Interfaces, Social Science Research*, and other journals.

‡Vladimir Malyshko is Associate Professor in the Department of Applied Mathematics at the Byelorussian State University in Minsk, USSR. He received his Ph.D. in Mathematical Cybernetics from the Minsk Institute of Mathematics. In 1981–82, he was a visiting scholar at The University of Michigan under the IREX program. His research interests include graph theory, network optimization, and analysis of computer algorithms and data structures. His publications have appeared in numerous Soviet and western journals, including *Kibernetika* (Kiev) and *Vesci Akademii Navuk BSSR*.

Visiting scholar under the IREX program for 1981–82 at The University of Michigan, Ann Arbor, MI 48109, U.S.A.

this system, costs can be further reduced if micro-processors are used as intermediary data concentrators instead of having each heliostat directly connected to the central computer. One possible configuration is that of the Smith helioelectric farm[21]. In their proposed plant, a central computer is linked to 600 minicomputers, each of which is connected to 16 microprocessors. The microprocessors control 32 heliostats each.

It should be emphasized that the minimization of expensive cable lengths is very important for large power plants. For example, to connect 21,000 heliostats independently to a central computer from a 1500 m by 1500 m field, 9000 km of cable are required[2]. This amount of cable may be reduced twenty fold by constructing a hierarchical system including microprocessors and by employing optimization methods to the design of that system.

In this article, we investigate methods for organizing connections in a two level control system for a field of heliostats to minimize the lengths of additional cable. We assume that the heliostats have been arranged to prevent adverse interactions. Given this arrangement, we then want to find the best connecting system.

Practical task

We are given $n+1$ points, (x_i, y_i), $i = 1, 2, \ldots, n+1$, where, for $i = 1, 2, \ldots, n$, the point is the site of a heliostat or microprocessor, and, for $i = n+1$, the point is the site of the central computer. We must allocate microprocessors according to the following conditions:

(a) Every heliostat must be connected with a microprocessor and every microprocessor must be connected with r heliostats;

(b) Every microprocessor must be connected with the central computer;

(c) The lengths of connecting cables must be minimized.

In our discussion, we propose three alternative approaches:

(1) An exact method which reduces the task to a p-median problem for a weighted graph and solves this using the simplex method and branch-and-bound ([16], p. 112).

(2) An approximate algorithm based on local optimization[19].

(3) An approximate algorithm based on constructing the minimum spanning tree using the algorithm in [18] and sequentially allocating p-medians to the vertices of this tree.

When n is large, the first algorithm may be inefficient, and using the approximate algorithms may be desirable. We present worst case estimations of the time complexity and accuracy for Algorithms 2 and 3. We also present computational results on a series of randomly generated problems. These results indicate that Algorithm 3 has the least average time requirement, but that Algorithm 2 achieves more accurate results. The cost of additional optimization upon these solutions should be weighed in specific applications.

2. MODEL DESCRIPTION AND SOLUTION METHODS

Let V be the set of given points, v_i, with coordinates, (x_i, y_i), where $i = 1, 2, \ldots, n+1$. The Euclidean distance ρ_{ij} between any pair of points, v_i, v_j, is given by

$$d(v_i, v_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \tag{1}$$

We consider the complete graph $G(V, E)$, defined on the given vertex set, V, where every edge in E has associated with it a nonnegative distance.

Let p be the number of medians (excluding the central point) which must be placed on the given set of vertices V. Then,

$$\frac{n-p}{r} = p, \quad \text{or} \quad p = \frac{n}{r+1}; \tag{2}$$

where the microprocessor is connected to r heliostats at sites other than its own.

The original practical problem may be reduced to finding the subset $V_p^* \subset V$ such that

$$\sigma(V_p^*) = \min\{\sigma(V_p^*)\},$$

where

$$\sigma(V_p) = \sum_{\forall v_j \in V/V_p} d(V_p, v_j), \tag{3}$$

$$d(V_p, v_j) = d(v_{i(j)}, v_j),$$

where a heliostat at v_j is connected to a microprocessor at $v_{i(j)} \in V_p$ and $|\{v_j: v_j \in V/V_p \text{ and } v_{i(j)} = v_l\}| \leq r$ for all $v_l \in V_p$. If the last constraint, requirement (a), is not present, then

$$d(V_p, v_j) = \min_{v_i \in V_p} d(v_i, v_j),$$

and the problem is known as the p-median problem. The additional requirement that each median (microprocessor) must be connected with exactly r vertices (heliostats), however, precludes direct application of methods for the p-median problem.

The problem is also similar to the *capacitated tree problem* [8, 17], in which a minimum spanning tree is found that satisfies a capacity constraint on the arcs. Our problem is different in that capacity applies to the median microprocessor. Also, each microprocessor is directly connected to the heliostats allocated to it in contrast to a tree in which nodes may be joined by several branches.

Other network problems may also be represented by this formulation. A distributed data processing system (see, e.g. [4]) for example, may consist of a large central processor and many small processors. In order to improve overall efficiency, intermediate processors may be placed at certain of the small processor sites. The problem described here would then apply to allocating these intermediate processors to minimize connection costs to the small processors. This may also have the additional benefit of decreased processing time.

Our problem can also be formulated as a binary integer linear program. We define the decision variables as

$$x_{ij} = \begin{cases} 1 \text{ if vertex } v_j \text{ is assigned to vertex } v_i, \\ \quad \text{or } v_i \text{ is assigned to } v_j, \\ 0 \text{ otherwise.} \end{cases} \tag{4}$$

Because of symmetry in the distances d_{ij}, we only need one variable for each pair of vertices i and j.

Letting $d_{ij} = d(v(i), v(j))$, the problem is:

$$\min \sum_{j=1}^{n+1} \sum_{i=1}^{j-1} d_{ij} x_{ij} \tag{5}$$

$$\text{subject to } \sum_{i=j+1}^{n+1} x_{ji} + \sum_{i=1}^{j-1} x_{ij} \geq 1, \; j = 1, \ldots, n; \tag{5.1}$$

$$\sum_{i=j+1}^{n+1} x_{ji} + \sum_{i=1}^{j-1} x_{ij} \leq r, j = 1, \ldots, n; \tag{5.2}$$

$$\sum_{i=1}^{n} x_{in+1} = p; \tag{5.3}$$

$$x_{ij} \leq x_{in+1}, \; i = 1, \ldots, n; \; j = 1, \ldots, n; \tag{5.4}$$

$$x_{ij} \in \{0, 1\}, \; i = 1, \ldots, n, \; j = 1, \ldots, n+1. \tag{5.5}$$

Constraint (5.1) indicates that every vertex represents a heliostat assigned to a microprocessor or a microprocessor assigned to the central computer. Constraint (5.2)

constrains the microprocessor to be connected to at most r heliostats. Constraint (5.3) yields p total microprocessors connected to the central median at $n + 1$, and (5.4) forces the heliostats to be connected to places of microprocessors. Constraint (5.5) ensures that a heliostat is either connected to another heliostat or not.

Problem 5 is a binary integer linear program with $(n + 1)^2$ variables. This can be solved exactly by a direct branch-and-bound procedure, or it may be solved by the simplex method by relaxing constraint (5.5) to $0 \leq x_{ij} \leq 1$. In the latter case, an exact solution may be found by further branching on the values of fractional x_{ij}. The number of additional steps may be small, as reported in Revelle and Swain [19]. This approach is used in Section 4. In the following, we will denote this method as Algorithm 1.

When n is large and the linear programming solution has many fractional values, branch-and-bound procedures may prove inefficient in finding a solution. In this case, some heuristic procedure may be necessary.

One such procedure that we will use here is due to Teitz and Bart[22]. We have extended this method to the capacitated case here in which every median must be connected to r vertices. The algorithm proceeds by starting with a given set of p medians and of vertex assignments to those medians. The first phase of the algorithm is to determine whether switching two vertices assignments can lead to an overall reduction in the total distance.

For this algorithm, we use a different definition for the value of the p-median assignment. For a specific instance I of the problem, we let:

$$\bar{\sigma}(V_p) = \sum_{v_i \in V_p} \left(\sum_{v_j \in \bar{V}_i} d(v_i, v_j) \right), \tag{6}$$

where $\bar{V}_i$ is the subset of r vertices connected to the median v_i, and the medians are $v_{\tau(1)}$, $v_{\tau(2)}, \ldots, v_{\tau(p)}$. This definition is used to explicitly incorporate the capacity constraint on microprocessor connections.

ALGORITHM 2.

Step 1. Assign vertices to p medians such that each median is connected to at most r vertices. Let the median set be $V_p = \{v_{\tau(1)}, v_{\tau(2)}, \ldots, v_{\tau(p)}\}$. Order all vertices v_i, $i \leq n$, $v_i \notin V_p$ by $v_{i_1}, v_{i_2}, \ldots, v_{i_{n-p}}$. Let $j = 1$. Let $k = 1$. Let $V_p'' = V_p$.

Step 2. Replace median $v_{\tau(k)}$ by v_{i_j} to form a new median set $V_p' = (V_p - \{v_{\tau(k)}\}) \cup \{v_{i_j}\}$. Let $k^* = 0$.

(a) Allocate each vertex $v_l \in V$ to the closest median vertex $v_q \in V_p'$ that is connected to less than r vertices and calculate $\bar{\sigma}(V_p')$.

(b) If $\bar{\sigma}(V_p') < \bar{\sigma}(V_p'')$, then, V_p' becomes the best median set found by replacing a median in V_p by v_{i_j}, so let $V_p'' = V_p'$ and $k^* = k$.

(c) If $k < p$, then let $k = k + 1$ and repeat. Else ($k = p$), let $V_p = V_p''$, $k = 1$, and if $k^* > 0$, let $v_{i_j} = v_{\tau(k^*)}$.

(d) If $j < n - p$, let $j = j + 1$ and repeat. Else ($j = n - p$); if V_p has changed since Step 1, go to Step 1, else, go to Step 3.

Step 3. For every pair of non-median vertices, (v_i, v_j), $i \neq j$, such that $v_i \in \bar{V}_k$ and $v_j \in \bar{V}_l$, $k \neq l$: If $d(v_i, v_l) + d(v_j, v_k) < d(v_i, v_k) + d(v_j, v_l)$, then interchange v_i and v_j, that is, let

(a) $\bar{V}_k = (\bar{V}_k - \{v_i\}) \cup \{v_j\}$,

(b) $\bar{V}_l = (\bar{V}_l - \{v_j\}) \cup \{v_i\}$.

Go to Step 4.

Step 4. STOP. V_p is an approximate set of p-medians, and $\bar{V}_{\tau(1)}, \bar{V}_{\tau(2)}, \ldots, \bar{V}_{\tau(p)}$ are partitions of vertices to those medians. Let $\sigma_2(I) = \bar{\sigma}(V_p)$ for an instance I of the problem.

This algorithm results in essentially a λ-optimal assignment of medians. Further λ-optimal solutions as in Lin's[14] procedures for the travelling salesman problem may also be found, but they require substantial increases in computational effort. The heuristic algorithm above requires $(n - p)\,p$ steps for each pass through Step 2 and $(n - p)^2$ steps in

Step 3. The algorithm must terminate since at every cycle an improved solutions is considered, and since there are a finite number of possible starts for Step 2. The number of these possible starts is $\binom{n}{p}$, the number of possible median sets.

The main idea of the third method is to reduce the given initial problem on the complete weighted graph $G(V, E)$ to the solution of an approximate variant of the p-median problem on a tree.

Let $T(V, \bar{E})$ be a minimum spanning in $G(V, E)$, which may be constructed using Prim's algorithm[18].

Let $(v_i, v_j) \in \bar{E}$. Removing the edge (v_i, v_j) from T divides the tree T into 2 components, $T_i(V_i, \bar{E}_i)$ and $T_j(V_j, \bar{E}_j)$ such that

$$v_i \in V_i, v_j \in V_j, \quad \text{and} \quad V_i \cup V_j = V.$$

PROPOSITION 1 (11)

If $|V_i| \geq |V_j|$, then the median v^* of tree T belongs to V_i.

Proof. In a tree T, distance between two vertices is calculated as the sum of the lengths of arcs along the shortest path from one vertex to another. We denote this distance between two vertices v_k and v_l as $d_T(k, l)$ and use this in the definition (3) of a median. Now, if $v^* = v_{j^*} \in V_j$, then

$$\begin{aligned}
\sigma(v_{j^*}) &= \sum_{v_k \in V_i} d_T(k, j^*) + \sum_{v_l \in V_j} d_T(l, j^*) \\
&= \sum_{v_k \in V_i} d_T(k, i) + |V_i| d_T(i, j^*) \\
&\quad + \sum_{v_l \in V_j} d_T(l, j^*) \\
&\geq \sum_{v_k \in V_i} d_T(k, i) + |V_j| d_T(i, j^*) \\
&\quad + \sum_{v_l \in V_j} d_T(l, j^*) \\
&= \sigma(v_i),
\end{aligned}$$

where we have used that every path from $v_k \in V_i$ to v_{j^*} must pass through v_i. This shows v_i is always at least as good as v_{j^*} and is strictly better if $|V_i| > |V_j|$.

In the following algorithm, we use the postorder traversal of T defined below.

DEFINITION ([1], p. 54)

Let T be a tree having root r with sons $v_1, v_2, \ldots, v_k$, $k \geq 0$. In the case $k = 0$, the tree consists of the single vertex r. A *postorder traversal of* T is defined recursively as:

(1) Visit in postorder the subtrees with roots, $v_1, v_2, \ldots, v_k$, in that order,

(2) Visit the root r.

ALGORITHM 3

Step 1. Construct the minimum spanning stree T of the graph $G(V, E)$ using Prim's algorithm[18]. Vertex v_{n+1} will be the root of T.

Step 2. Employing the postorder transversal of T, assign each vertex $v_i \in T$ a weight $W(v_i)$ which equals the number of its descendants, in other words, the number of vertices lying on all paths from v_i to the leaves. Allocate the first median v_1^* to the vertex v_{n+1}. Let $l = 1$ and $S = \phi$.

Step 3. Remove the edges adjacent to V_l^* and let $v_{j1}, v_{j2}, \ldots, v_{jk}$ be the vertices formerly adjacent to v_l^*. Include in S the set of subtrees, $T_1, T_2, \ldots, T_k$, with roots $v_{j1}, v_{j2}, \ldots, v_{jk}$.

Step 4. Choose from S, the subtree T_0^* such that:

$$W(v_{j_0}) = \max_{\forall T_i \in S} \{W(v_{j_i})\}.$$

Let $S = S - T_0^*$.

Step 5. In subtree T_0^*, find the vertex V_α such that

$$\Delta(v_\alpha) = \min_{\forall v_i \in T_0^*} \{\Delta(v_i)\}$$

where

$$\Delta(v_i) = |2W(v_i) - W(v_{j_0})|. \tag{7}$$

v_α is then the vertex with the most nearly equal number of ancestors and descendants.

Step 6. Allocate the median V_{l+1}^* to the vertex v_α. If $l < p$, then let $l = l + 1$ and go to Step 3.

Step 7. Let $\{v_1^*, v_2^*, \ldots, v_{p+1}^*\}$ be the set of medians found. Again, using postorder traversal of T, connect any noncentral median with any vertex v_j which has not yet been considered and is a descendant of the median.

Step 8. Connect any vertex not yet considered to the nearest noncentral median which has less than r adjacent vertices. Connect all noncentral medians with the central median, v_1^*.

Example

Consider the minimum spanning tree in Fig. 1 constructed in Step 1 where $p = 3$ and $r = 1$. In Step 2, weights are assigned as

i	$W(v_i)$
2	1
3	0
4	2

and $v_1^* = v_7$.

For $l = 1$, in Step 3, let $v_{j1} = v_2$, $v_{j2} = v_3$, $v_{j3} = v_4$. In Step 4, $v_{j0} = V_4$. In Step 5, $\Delta(v_i)$ is calculated as

i	$\Delta(v_i)$
4	2
5	0
6	2

so $v_\alpha = v_5$ and $v_2^* = v_5$.

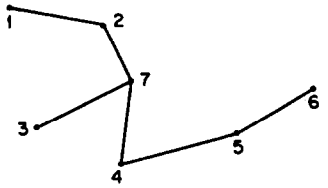

Fig. 1. Original minimum spanning tree.

For $l = 2$, in Step 3, let $v_{j1} = v_2$, $v_{j2} = v_3$. Next, v_2 is chosen in Step 4, and $\Delta(v_i)$ is calculated as

i	$\Delta(v_i)$
1	1
2	1,

and $v_a = v_1$ or v_2. Let $v_3^* = v_2$.

For $l = 3$, only v_3 and its subtree (the vertex v_3) are left, hence $v_4^* = v_3$.

In Step 7, v_1 is assigned to $v_3^* = v_2$ and v_6 is assigned to $v_2^* = v_5$ since they are the only descendants of the medians given. In Step 8, v_4 is connected to $v_4^* = v_3$ since $v_2^* = v_5$ is full. Vertices v_2, v_3, v_5 are then connected to v_7, resulting in Fig. 2.

It should be noted that the algorithm given above sequentially solves the following three problems (a) the construction of a minimum spanning tree, (b) the sequential allocation of p medians on this tree, and (c) the connection of exactly r vertices with every noncentral median. It is impossible to construct an exact polynomial algorithm for solving (a) and (c) because the corresponding problem of constructing a minimum spanning tree with bounded vertex degree is NP-complete ([10], p. 206).

3. ESTIMATION OF TIME COMPLEXITY AND ACCURACY OF THE ALGORITHMS

We first estimate the maximum number of elementary operations required by each algorithm for solving any instance of our problem. For Algorithm 1, the time complexity is not polynomially bounded because it has been shown that the simplex method may take an exponential number of steps to reach an optimum [13]. The branch-and-bound procedure may also involve an exponential number of steps.

Algorithm 2 can involve as many steps as the number of possible median sets, $\binom{n}{p}$. Since this grows exponentially with n for fixed r, Algorithm 2 is also not polynomially bounded. We note, however, that these bounds apply to worst case behavior and that average case behavior may be much better.

We will now estimate the time complexity of Algorithm 3 step by step. Step 1 may be completed in $0(n^2)$ since Prim's algorithm for spanning trees has the estimation $0(n^2)$. Step 2 requires $0(m)$ operations where $m = 0(n^2)$ for a complete graph. Step 3 consists of $0(p \cdot c_1)$ operations where c_1 is a constant and $p = n/r + 1$. Steps 4 and 5 in the long run require $0(p \cdot n)$ operations. In fact, one application of Steps 4 and 5 will take $0(n)$ operations, if we construct a list of the vertices in which each vertex is assigned the weight of the subtree which includes that vertex and the corresponding value of Δ. Step 6 requires $0(p \cdot c_2)$ operations where c_2 is a constant. Steps 7 and 8 may be done in $0(n^2)$ operations. Combining these estimates, we obtain the common estimation of $0(n^2)$.

Algorithm 1 using the branch-and-bound procedure guarantees an optimal solution, but Algorithms 2 and 3 provide only approximate solutions. We next consider the accuracy of those algorithms.

Let I be an instance of our problem and let $\sigma_1(I) = \sigma_{\text{opt}}(I)$, the optimal solution value obtained using the exact algorithm 1. An instance of this problem will be defined as a set

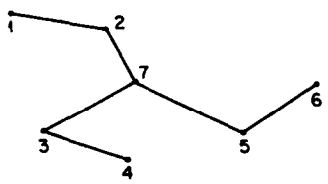

Fig. 2. Median locations at 2, 3, 5 and 7.

of given vertex locations on a predetermined size field. Let $\sigma_2(I)$ and $\sigma_3(I)$ be the values of approximate solutions found using Algorithms 2 and 3. The problem in the remainder of this section is to determine the absolute performance ratios ([10], p. 128), which are determined by the following formulas:

$$R_2 = \inf\{r_2 \geq 1: R_2(I) = \frac{\sigma_2(I)}{\sigma_1(I)} \leq r_2 \text{ for all possible instances } I\},$$
$$R_3 = \inf\{r_3 \geq 1: R_3(I) = \frac{\sigma_3(I)}{\sigma_1(I)} \leq r_3 \text{ for all possible instances } I\}. \tag{8}$$

We first present a lower bound estimate of the optimal solution, $\sigma_1(I)$. For a problem with n vertices and $p = (n/r + 1)$ medians, the number of connections between the medians and all vertices may be calculated by the following formula:

$$n - p = n - \frac{n}{r+1} = \frac{nr}{r+1} \tag{9}$$

Let $\mathcal{T}$ be the set of all possible spanning trees for instance I. Let us denote the minimum edge length in all such trees by

$$\rho_{\min} = \min_{\forall (x_i, x_j) \in T \subset \mathcal{T}} \{d(x_i, x_j)\}, \tag{10}$$

and let the maximum edge distance be

$$\rho_{\max} = \max_{\forall (x_i, x_j) \in T \in \mathcal{T}} \{d(x_i, x_j)\}. \tag{11}$$

It is clear that

$$\sigma_1(I) \geq \frac{nr}{r+1} \rho_{\min}. \tag{12}$$

An optimal solution to the problem is not guaranteed by Algorithm 2, but the value of the solution may be bounded. At termination of the algorithm, no vertex may be substituted for another and lead to a lower value. We let $(v_1^*, v_2^*, \ldots, v_p^*)$ be the median set in the optimal solution. Since none of these vertices which are not in the median set of the approximate solution may be substituted for a median in the approximate solution and lead to a lower total distance, we obtain

$$\sum_{j \in V_i^1} d(v_i^j, v_i^1) + d(v_k^*, v_{l(k)}^1) \leq \sum_{j \in V_i^1} d(v_i^j, v_k^*) + d(v_i^1, v_{l(k)}^1) \tag{13}$$

where $v_k^* \in V_{l(k)}^1$ in the approximate problem. This inequality holds for every vertex $v_i^j \in V_{k(j)}^*$ in the optimal solution.

We take a sum of the l.h.s. of (13) for v_i^j for all i and j to find

$$\sum_{i=1}^{p} \sum_{j \ni v_i, j \in V_i^1} \left\{ \sum_{j \ni v_i^1 \in V_i^1} d(v_i^j, v_i^1) + d(v_i^j, v_i^1) + d(v_{k(j)}^*, v_{l(k)}^1) \right\}$$
$$= r\sigma_2(I) + \sum_{i=1}^{p} \sum_{j} d(v_{k(j)}^*, v_{l(k)}^1) \geq r\sigma_2(I) + r \cdot p \cdot d_{\min}. \tag{14}$$

where $\sigma_2(I)$ is the value obtained in this instance by this approximate algorithm, $n = p(r+1)$ is assumed, and $d_{\min} = \min_{v_i, v_j} d(v_i, v_j)$.

For the r.h.s. of (13), we obtain:

$$\sum_{i=1}^{p}\sum_{j}\left(\sum_{j\neq j} d(v_i^j, v_{k(j)}^*) + d(v_j, v_{k(j)}^*) + d(v_i^1, v_{k(k)}^1)\right) \leq \sum_{i=1}^{p}\sum_{j}\left(\sum_{j\neq j} d(v_i^j, v_i^1)\right.$$

$$\left. + \sum_{j\neq j} d(v_i^1, v_{k(j)}^*) + d(v_j, v_{k(j)}^*) + d(v_i^1, v_{k(k)}^1)\right) \leq (r-1)\cdot\sigma_2(I) + \sigma_1(I) + r\cdot p\cdot d_{\max}, \quad (15)$$

where $d_{\max} = \max_{v_i, v_j} d(v_i, v_j)$. (14) and (15) give us the following.

PROPOSITION 2

The solution, $\sigma_2(I)$ obtained by Algorithm 2, is bounded by

$$\sigma_2(I) \leq \sigma_1(I) - rp(d_{\max} - d_{\min}), \quad (16)$$

where $\sigma_1(I)$ is the exact solution to (5).

Proof. This follows directly from (14) and (15).

The absolute performance ratio for Algorithm 2 may be obtained by:

$$R_2 = \inf_I \left\{\frac{\sigma_2(I)}{\sigma_i(I)}\right\} \leq 1 + \frac{rp(d_{\max} - d_{\min})}{\frac{nr}{r+1}\cdot\rho_{\min}} = 1 + \frac{d_{\max} - d_{\min}}{\rho_{\min}}, \quad (17)$$

using (16), (12), and $p = n/(r+1)$. We note that for a field of dimensions a by b, $d_{\max} \leq \sqrt{a^2+b^2}$, so R^2 is bounded by a constant depending on the field size.

We will now find the estimation for $\sigma_3(I)$.

PROPOSITION 3

After allocating p medians to the set of vertices any subtree obtained from Algorithm 3 will include no more than:

$$\alpha = \frac{n}{2^k} \text{ vertices,}$$

where

$$k = \lfloor \log_2(p+1) \rfloor .$$

Proof. We will proceed by induction on p. For $p = 1$, then $k = 1$, $\alpha = n/2$. Suppose that the proposition holds for $p = l$ and consider the case, $p = l + 1$. Two cases are possible:

(1) $\lfloor \log_2(l+1) \rfloor = \lfloor \log_2(l+2) \rfloor$,
(2) $\lfloor \log_2(l+2) \rfloor = \lfloor \log_2(l+1) \rfloor + 1$.

Let $l^* = \min\{l\}$ such that $\lfloor \log_2(l^*+1) \rfloor = \lfloor \log_2(l+1) \rfloor$. After having allocated l^* medians, we will have l^*+1 subtrees. It is clear that in Case 1 above there will exist at least one subtree with $n/2^k$ vertices where $k = \lfloor \log_2(l+1) \rfloor$, and, in Case 2, all subtrees will be the same size, in other words, every subtree will have no more than $n/2^k$ vertices, where $k = \lfloor \log_2(l+1) \rfloor + 1$. ■

It follows from Proposition 2 that the maximum path length in such a subtree will be no greater than

$$\left(\frac{n}{2^k} - 1\right)\rho_{\max},$$

where

$$k = \lfloor \log_2(p+1) \rfloor .$$

Let $T_1, T_2, \ldots, T_s$ be the sequence of subtrees of the minimum spanning tree T with corresponding roots $v_{j_1}, v_{j_2}, \ldots, v_{j_s}$ adjacent with the central median v^*_{p+1} and such that $w(v_{j_1}) \geq w(v_{j_2}) \geq \ldots \geq w(v_{j_s})$. Then, possibly, starting with some t, $2 \leq t \leq s$, each of the subtrees T_l, where $t \leq l \leq s$, will not have a median.

Let

$$b = \sum_{l=t}^{s} w(v_{j_l}). \tag{18}$$

It then follows from Proposition 3 that any subtree resulting from Algorithm 3 will include no more than $(n - b)/2^k$ vertices, where $k = \lfloor \log_2(p+1) \rfloor$. Without loss of generality, let $p = 2^m$, where $m \geq 1$. (If not, p may be decreased to the nearest power of 2.) Then

$$\frac{n-b}{2\lfloor \log_2(p+1) \rfloor} \leq \frac{n-b}{p} = \frac{(n-b)(r+1)}{n}. \tag{19}$$

In order to obtain an upper bound estimate of $\sigma_3(I)$, it is necessary to investigate the most "unsuitable" instance of the problem for Algorithm 3. Obviously, this will occur when $t = 2$, that is, when all noncentral medians are allocated to a single tree, T_1 (see Fig. 3). It is clear that in this case each of the subtrees $T_2, T_3, \ldots, T_s$ will contain at most $(n-b)(r+1)/n$ vertices. The most "unsuitable" structure for subtrees, $T_1, T_2, \ldots, T_s$, will be a tree, that contains only one leaf (i.e. a path).

Let us denote by $\sigma_{31}(I)$ the total distance from the vertices of subtree T_1 to the corresponding nearest medians (Step 7 of Algorithm 3). Let $\sigma_{32}(I)$ be the total distance from the vertices of subtrees $T_2, T_3, \ldots, T_k$, to the corresponding nearest medians of subtree T_1 (Step 8 of Algorithm 3). Then

$$\sigma_3(I) = \sigma_{31}(I) + \sigma_{32}(I). \tag{20}$$

Let v_j be one of the vertices of subtrees, $T_2, T_3, \ldots, T_k$, and let v_i^* be the nearest median in subtree T_1 to vertex v_j. Then, from the triangle inequality in Euclidean space, it follows

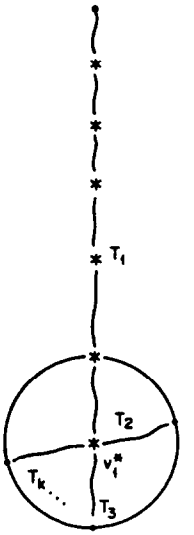

Fig. 3. An example of the worst case tree for Algorithm 3. Stars are used to show the median allocations. The circle bounds the area of allocation of vertices of subtrees, $T_2, T_3, \ldots, T_k$.

that:

$$d(v_j, v_i^*) < d(v_j, v_{p+1}^*) + d(v_{p+1}^*, v_i^*). \tag{21}$$

From Step 5 of Algorithm 3, it follows that every noncentral median will be connected with no more than $\lceil b/p \rceil - \lceil b(r+1)/n \rceil$ vertices. This may be roughly estimated by $\lceil b(n+1)/n \rceil \leq 2b(r+1)/n$.
Then

$$\sigma_{32}(I) \leq \sum_{\forall v_j \in \bigcup_{i=2}^{k} T_i} d(v_j, v_{p+1}^*) + \sum_{i=1}^{p} \frac{2b(r+1)}{n} \cdot d(v_{p+1}^*, v_i^*). \tag{22}$$

Let us estimate the r.h.s. term in (22). It is clear that any of the subtrees $T_2, T_3, \ldots, T_k$, has no more than $(n-b)(r-1)/n$ vertices. Furthermore, the worst structure for any of these subtrees is a path. Using these facts, we have

$$\sum_{\forall v_i \in \bigcup_{i=2}^{k} T_i} d(v_j, v_{p+1}^*) \leq \frac{b \cdot \rho_{\max}}{\frac{(n-b)(r+1)}{n}} \left[1 + 2 + \cdots + \frac{(n-b)(r+1)}{n} - 1\right]$$

$$= \frac{b \cdot n \cdot \rho_{\max}}{(n-b)(r+1)} \cdot \frac{(n-b)^2(r+1)^2}{2n^2} = \frac{b(n-b)(r+1)}{2n} \rho_{\max}. \tag{23}$$

$$\sum \frac{2b\,(r+1)}{n} \cdot d(v_{p+1}^*, v_i^*) \leq \frac{2b(r+1)}{n} \cdot \frac{(n-b)\,(r+1)}{n} \cdot \rho_{\max} \left[1 + 2 + \cdots + \frac{n}{r+1}\right]$$

$$= \frac{2b \cdot (r+1)}{n} \cdot \rho_{\max} \cdot \frac{(n+r+1)}{2(r+1)^2} \cdot \frac{(n-b)\,(r+1)}{n} \cdot n$$

$$= \frac{b \cdot (n-b)(n+r+1)}{r} \cdot \rho_{\max}. \tag{24}$$

Using (23) and (24), we may estimate $\sigma_{32}(I)$ by:

$$\sigma_{32}(I) \leq \frac{b(n-b)(r+1)}{2n} \rho_{\max} + \frac{b(n-b)(n+r+1)}{n} \rho_{\max}. \tag{25}$$

We may also obtain an estimate for $\sigma_{31}(I)$ by:

$$\sigma_{31}(I) \leq \frac{n}{r+1} \cdot \rho_{\max} \cdot \left(1 + 2 + \cdots + \frac{(n-b)\,(r+1)}{n} - 1\right)$$

$$= \frac{n}{r+1} \cdot \frac{(n-b)^2\,(r+1)^2}{2n} \cdot \rho_{\max} = \frac{(n-b)^2\,(r+1)}{2n} \rho_{\max}. \tag{26}$$

Using (25) and (26), the following estimate of $\sigma_3(I)$ is obtained.

$$\sigma_3(I) \leq \frac{(n-b)^2(r+1)}{2n} \rho_{\max} + \frac{b(n-b)(r+1)}{2n} \rho_{\max} + \frac{b(n-b)(n+r+1)}{n} \rho_{\max}$$

$$- \frac{n-b}{2n} \rho_{\max} [(n-b)(r+1) + b(r+1) + 2b(n+r+1)]$$

$$- \frac{n-b}{2n} \rho_{\max} [nr + n + 2bn + 2br + 2b]$$

$$- \frac{n-b}{2n} \rho_{\max} [n(2b + r + 1) + 2b(r+1)] \tag{27}$$

Using (27) and (12), we may find the ratio R_3 by

$$R_3 = \inf\left\{\frac{\sigma_3(I)}{\sigma_1(I)}\right\} \leq \frac{(n-b)[b(2b+r+1)+2b(r+1)]\cdot(r+1)}{2n\cdot n\cdot r}\cdot\frac{\rho_{\max}}{\rho_{\min}}$$

$$= \frac{\rho_{\max}}{\rho_{\min}}\cdot\frac{(n-b)(r+1)[n(2b+r+1)+2b(r+1)]}{2n^2\cdot r}. \tag{28}$$

From (23) it follows that if we have $b = 0$, then:

$$R_3 \leq \frac{\rho_{\max}}{\rho_{\min}}\cdot\frac{(r+1)n(r+1)}{2nr} = \frac{\rho_{\max}}{\rho_{\min}}\frac{(r+1)^2}{2r}$$

that is

$$R_3 \leq \frac{\rho_{\max}}{\rho_{\min}}\frac{(r+1)^2}{2r}, \tag{29}$$

where $\rho_{\max}/\rho_{\min}$ is a constant that depends on the size of the field. We observe that R_2 and R_3 are both bounded by constants when b is a constant but that R_3 may be $0(n)$ if $b = 0(n)$. This case is extremely unlikely for a field with equally distributed heliostats. It should also be noted that Algorithm 3 has the advantage of a polynomial bound in time complexity.

4. COMPUTATIONAL RESULTS

A set of randomly generated problems was used to test the algorithms. A FORTRAN code, SWTCHR, was written to perform Algorithm 2, and another FORTRAN code, MINST, was written for Algorithm 3. Both of these codes were compiled on the non-optimized standard FORTRAN-G compiler on The University of Michigan's Amdahl 470/V8 computer. All processing was also done on this machine. The linear program of Algorithm 1 was executed by the versatile MINOS package[16].

The linear program in (5) requires $(n+1)^2$ variables and more than $(n+1)^2$ rows. As

Table 1. Solutions for $n = 28$ heliostats

Problem Number	Linear Program			Algorithm 2			Algorithm 3		
	CPUs	Iterations	Value	CPUs	Value	% of LP Opti.	CPUs	Value	% of LP Opti.
1	34.7	606	6095.7	.31	6306.4	103	.16	6528.1	107
2	32.5	559	6129.5	.32	6350.7	104	.16	7642.3	125
3	38.3	680	5839.7	.32	8301.0	144	.15	7618.6	130
4	30.4	529	5992.3	.32	6793.6	113	.16	6931.1	116
5	37.7	629	5818.1	.32	6716.8	115	.18	8577.3	147
6	37.0	651	5426.7	.32	6005.3	111	.16	7723.1	142
7	40.8	754	5894.4	.31	6656.1	113	.14	7925.4	134
8	30.0	519	5230.3	.31	6346.4	121	.14	5964.6	114
9	45.0	765	5691.5	.31	7935.5	139	.14	8424.3	148
10	32.9	487	5711.5	.31	6382.0	112	.15	7348.8	129
Average	35.9	618	5783.0	.32	6788.4	118	.15	7469.4	129

n increases, the problem size becomes prohibitively large and finding a solution becomes extremely expensive. For this reason, tests of all three algorithms were limited to small n. The results in Table 1 are for problems in a 1000 m × 1000 m field in which $n = 28$ heliostats have randomly been placed uniformly in the field. For these examples, the microprocessor capacity was set at ten and three microprocessors were located within the field. The resulting linear program for these examples included 842 rows, 785 columns and 3921 nonzero elements. The CPU times given exclude input and solution output. Problems 2, 5, 6, 7 and 10 had some noninteger variables in their solution. These problems required additional processing through branch-and-bound to obtain an optimal all-integer result. The times reported for these problems do not include that additional processing.

These problems were also solved by Algorithms 2 and 3 and the results were compared with the optimal linear programming solution. Algorithm 2 was started with a randomly chosen median allocation. The average accuracy of Algorithm 2 on the first set of problems was better than that of Algorithm 3 but its time requirement was greater.

Table 2 contains results for 100 random heliostat sites and for a capacity of 20 at each of 5 microprocessors. These problems are excessively large for an efficient linear program solution, hence the results are compared with the length of the minimum spanning tree instead of the linear program solution. Algorithm 2 again provides better average results but it requires consistently more time.

It should be noted that the majority of calculation time is spent determining distances between pairs of heliostats. Since this operation only needs to be performed once, a combination of Algorithms 3 and 2 may prove quite useful in practice. In finding the minimum spanning tree in Algorithm 3, distances may be calculated and stored out of core. The solution from Algorithm 3 may be used as an initialization for Algorithm 2 and the distances may be recovered without additional computation.

5. CONCLUSION

The problem of minimizing cable connections in a solar power system has been presented. This task was modeled as a p-median problem with additional constraints and three algorithms were presented for its solution. Worst case analysis of the algorithms indicated that one heuristic method was polynomially bounded and that its absolute

Table 2. Solutions for $n = 100$ heliostats

Problem Number	Minimal Spanning Tree Value	Algorithm 2		Multiple of MST	Algorithm 3		Multiple of MST
		CPUs	Value		CPUs	Value	
1	6961.2	10.5	22516.5	3.23	4.2	27679.8	3.98
2	6998.3	10.2	19669.2	2.81	4.1	20321.9	2.90
3	7051.6	10.2	23107.3	3.28	4.1	23310.0	3.30
4	6635.9	10.3	22783.3	3.43	4.1	20603.0	3.10
5	7017.0	10.3	23285.5	3.32	4.1	25748.7	3.67
6	6493.7	10.3	20549.5	3.16	4.1	25649.0	3.95
7	6892.0	10.4	18542.7	2.69	4.1	25773.9	3.74
8	6641.3	10.3	22196.7	3.34	4.1	24504.6	3.69
9	6993.1	10.4	25149.0	3.60	4.1	29803.8	4.26
10	6759.1	10.3	18329.6	2.71	4.1	20978.2	3.10
Average	6844.6	10.3	21612.9	3.16	4.1	24437.3	3.57

performance ratio was, in general, bounded by a constant. Computational results supported the relative efficiency of this polynomial heuristic algorithm.

REFERENCES

1. A. V. Aho, T. E. Hopcroft and J. D. Ullman, *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reading, Massachusetts (1974).
2. G. E. Ahromenko, *et al*; The control of a field of heliostats of solar power plants. *Heliotechnics (USSR)* **4**, 16–22 (1980).
3. N. J. Bigon, Power from the Sun by thermodynamics: French tower type projects. In *Solar Energy: International Progress, Proceedings of the International Symposium Workshop on Solar Energy* (Edited by T. N. Veziroglu) Vol. 3, pp. 1329–1337, 16–22, June 1978, Cairo, Egypt. Pergamon Press, New York (1980).
4. G. Bucci and D. N. Streeter, A methodology for the design of distributed information systems. *CACM* **22**, 233–245 (1979).
5. P. O. Carden, Steering a field of mirrors using a shared computer-based controller. *Solar Energy* **20**, 343–355 (1978).
6. N. Christofides, *Graph Theory, An Algorithmic Approach.* Academic Press, New York (1977).
7. R. C. Enger and H. Weicher, Solar electric generating system resource requirements. *Solar Energy* **23**, 255–261 (1979).
8. L. R. Esau and K. C. Williams, On teleprocessing system design, Part II. *IBM Systems J.* **5**, 142–147 (1966).
9. C. Francia, Pilot plants of solar steam generating stations. *Solar Energy* **12**, 51–64 (1968).
10. M. R. Garey and D. S. Johnson, *Computer and Intractability, A Guide to the Theory of NP-Completeness.* W. H. Freeman & Company, San Francisco (1979).
11. A. J. Goldman, Optimal center location in simple networks. *Transp. Sci.* **5**, 212–221 (1971).
12. R. J. Holl, A central receiver solar power system for remote locations. In *Solar Energy: Int. Progress Proc. of the Int. Symp. Workshop on Solar Energy* (Edited by T. N. Veziroglu) Vol. 3, pp. 1408–1425, 16–22 June 1978, Cairo, Egypt. Pergamon Press, New York (1980).
13. V. Klee and G. Minty, How good is the simplex algorithm? In *Inequalities III* (Edited by O. Shisha), pp. 159–175. Academic Press, New York (1972).
14. S. Lin, Computer solution of the travelling salesman problems. *Bell System Tech. J.* **44**, 2245–2269 (1965).
15. F. W. Lipps and L. L. Vant-Hull, A cellwise method for the optimization of large central receiver systems. *Solar Energy* **20**, 505–516 (1978).
16. B. A. Murtagh and M. A. Saunders, *MINOS User's Guide.* Systems Optimization Laboratory, Technical Report 77-9, Stanford (1977).
17. C. H. Papadimitriou, The complexity of the capacitated tree problem, *Networks* **8**, 217–230 (1978).
18. R. C. Prim, Shortest connection networks and some generalizations. *Bell System Tech. J.* **36**, 1389–1401 (1957).
19. C. S. Revelle and R. W. Swain, Central facilities location, *Geographical Analysis* **2**, 30–42 (1970).
20. Sandia National Labs, *User's Manual for Delson 2: A Computer Code for Calculating the Optical Performance and Optimal system Design.* Report # 8237, Albuquerque (1981).
21. O. T. M. Smith and P. S. Smith, A helioelectric farm. In *Solar Energy: Int. Progress, Proc. of the Int. Symp. Workshop on Solar Energy* (Edited by T. N. Veziroglu), Vol. 3, pp. 1368–1407, 16–22 June 1978, Cario, Egypt. Pergamon Press, New York (1980).
22. M. B. Teitz and P. Bart, Heuristic methods for estimating the generalized vertex median of a weighted graph. *Oper. Res.* **16**, 995–961 (1968).
23. L. Vant-Hull, Process heat from solar energy. *Energy Progress* **2**, 107–110 (1982).