LOCATION PROBLEMS IN GRAPHS

by

MARGIT GRÖGER

6723 Szeged, Juharfás u. 5/A.

(Received July 8, 1983)

Introduction

Several problems in discrete optimization consist in finding the optimal location of centers in a network. There are several different optimality criteria: we speak of the p-center problem if the objective function to be minimized is the maximum of the distance of each vertex and the center closest to it. In the p-median problem we consider sum instead of maximum. In this paper we discuss the p-center problem.

As the problem is shown to be NP-complete it is reasonable to look for heuristics. We consider a heuristic of the local search type i.e. given an approximate solution we try to improve it by searching a certain neighbourhood (see Papadimitriou [4]). The algorithm works quite well in practice. On the other hand its worst-case behaviour is bad: we show it does not guarantee optimality for any fixed value of the parameters and that the ratio to the optimal solution is not bounded by any constant. We also give an example of the phenomenon that enlarging the neighbourhoods searched does not lead necessarily to a better solution.

The other possible approach is to consider efficiently solvable special cases. Efficient algorithms are known for trees (see Hakimi – Kariv [2]). Applying an algorithm for the p-center problem we give a solution to the following modified problem: find p-centers with the restriction that each center must belong to a previously specified subtree of the network, where the subtrees are assumed to be disjoint. The natural interpretation is that it may be required that some regions must contain a center.

Without the assumption of disjointness there is no polynomial algorithm known to solve the problem. We conjecture it to be NP-complete.

1. Definitions

A graph $G = (V, E)$ is meant to be undirected, without loops and multiple edges. We assume that graphs are connected.

Definition 1. A *weighted network* is a triple $H = (G, a, s)$, where G is a graph, $a: E \to R^+$ is the *edge-weight function*, $s: V \to R^+ \cup \{0\}$ is the *vertex-weight function*. A network is *unweighted* if s is constant (in this case we may assume $s \equiv 1$).

The distance of $v, w \in V$ in a graph with edge-weights is denoted by $d(v, w)$.

Definition 2. Let $H = (G, a, s)$ be a weighted network, $V_1 \subset V$, $v \in V$. We define the *generalized distance functions*

$$d(v, V_1) := \min \{d(v, w) : w \in V_1\}$$

$$r(V, V_1) := \max \{s(w) \cdot d(w, V_1) : w \in V\}$$

Definition 3. $V_1 \subset V$, $|V_1| = p$ is a *p-center* of H if

$$r(V, V_1) = \min \{r(V, W) : W \subset V, |W| = p\},$$

and in this case $r(V, V_1)$ is the *p-radius* of H.

Definition 4. Let $H = (G, a, s)$ be a network. $V_1 \subset V$, $|V_1| = p$ is r-optimal if for every $V_2 \subset V$, $|V_2| = p$, $|V_1 \cap V_2| \geq p - r$

$$r(V, V_1) \leq r(V, V_2)$$

holds. We define

$$E_{r,p}(H) := \frac{\max \{r(V, V_1) : V_1 \text{ is } r\text{-optimal}\}}{r_p(H)},$$

where $r_p(H)$ denotes the p-radius of H.

2. A heuristic algorithm for the p-center problem

Input: an unweighted network $H = (G, a)$, p, r.
Output: $V_1 \subset V$, $|V_1| = p$ r-optimal solution and the p-radius belonging to V_1.
Algorithm: $V_1 := \{1, \ldots, p\}$, FIX $:= \{1, \ldots, p-r\}$,
CHANGE $:= \{p-r+1, \ldots, p\}$.

For FIX $\subset V_1$, $|\text{FIX}| = p - r$ *do*
  For $Y \subset V - \text{FIX}$, $|Y| = r$ *do*
    if $r(V, \text{FIX} \cup Y) < r(V, V_1)$ *then*
      CHANGE $:= Y$
      $V_1 :=$ FIX $\cap$ CHANGE
  end
end

It is straightforward to modify this algorithm for the case of weighted networks.

Theorem 1. *For every* $p \geq 2$, $1 \leq r < p$ *there exists an unweighted network* $H = (G, a)$ *and a subset* $V_1 = \{v_1, \ldots, v_p\} \subset V$ *s.t.*

(i) V_1 *is not a* p*-center of* H,
(ii) V_1 *is* r*-optimal,*
(iii) V_1 *is not* $r+1$ *-optimal.*

Furthermore, H *can be chosen to be a tree.* □

Proof. Consider the following unweighted network (where $a = 1$).

$$V = \{1, 2, \ldots, 3p\},$$
$$E = \{(1, 2), (2, 3), \ldots, (3p-1, 3p)\}.$$

The unique p-center of H is

$$A = \{2, 5, 8, \ldots, 3p-4, 3p-1\}.$$

Indeed

(i) If $1 \leq y_1 < y_2 < \ldots < y_p \leq 3p$ and $Y = \{y_1, y_2, \ldots, y_p\}$,
then for $i \in V - Y$: $d(i, Y) \geq 1$ and so $r(V, Y) = \max_{i \in V} d(i, Y) \geq 1$.
(ii) $r(V, A) = 1$.

A is unique. Suppose Y is also a p-center. Then for every $i \in V - Y$ $d(i, Y) = 1$. Hence

$$y_1 \leq 2$$
$$y_{t+1} - y_t \leq 3 \quad (1 \leq t \leq p-1)$$
$$y_p \geq 3p - 1.$$

So on the one hand

$$y_p - y_1 \geq 3p - 3,$$

on the other hand

$$y_{t+1} - y_t \leq 3 \quad (1 \leq t \leq p-1).$$

Summing these inequalities

$$y_p - y_1 \leq 3p - 3.$$

Hence $y_p - y_1 = 3p - 3$, thus equality must hold in all these inequalities. Finally let $V_r := \{1, 4, 7, \ldots, 3r+1, 3r+5, \ldots, 3p-4, 3p-1\}$. Then we have $r(V, V_r) = 2$. As the only subset of vertices with a smaller radius is A and $A \cap V_r = \{3r+5, \ldots, 3p-1\}$, thus A contains $r+1$ elements not belonging to V_r. □

Next we consider the question whether there is any bound for the ratio of the value of the solution given by the algorithm and the optimal value.

Theorem 2. *For* $1 \leq r \leq p-2$

$$\sup \{E_{r,p}(H): H \text{ is a network}\} = \infty. \quad \square$$

Proof. As an $r+k$-optimal solution is r-optimal as well (if $k>0$) we have $E_{r,\,p}(H) \geq E_{p-2,\,p}(H)$. Thus it is sufficient to show

$$\sup \{E_{p-2,\,p}(H): H \text{ is a network}\} = \infty.$$

Let p be fixed, and K be an arbitrary real number. Consider the network

$$V = \{1, 2, \ldots 4p-3\}$$

$$E = \{(1,\ 2k-1):\ k = 2,\ 3,\ \ldots,\ 2p-1\} \cup$$

$$\cup \{(k,\ k+1):\ k = 2,\ \ldots,\ 4p-3\}$$

$$a\,(1, 4k-1) = 1 \qquad (k = 1, \ldots, p-1)$$

$$a\,(1, 4k+1) = K \qquad (k = 1, \ldots, p-1)$$

$$a\,(4k, 4k+1) = 1 \qquad (k = 1, \ldots, p-1)$$

$$a\,(4k+1, 4k+2) = 1 \qquad (k = 1, \ldots, p-1)$$

$$a\,(4k-2, 4k-1) = K \qquad (k = 1, \ldots, p-1)$$

$$a\,(4k-1, 4k) = K \qquad (k = 1, \ldots, p-1)$$

The unique p-center of this network is

$$V_0 = \{1, 5, 9, 13, \ldots, 4p-3\}$$

and $r(V, V_0) = 1$.
It can be shown that $V_1 = \{1, 3, 7, 11, \ldots, 4p-5\}$ is $p-2$-optimal. Indeed, for any set W with a p-radius smaller than K must satisfy

$$W \cap \{1, 3, 7, 11, \ldots, 4p-5\} \neq \emptyset \qquad (d(1, W) < K)$$

$$W \cap \{4k, 4k+1, 4k+2\} \neq \emptyset \qquad (d(4k+1, W) < K,\ k = 1, 2, \ldots, p-1)$$

(the intersection of p disjoint sets with W is nonempty and $|W| = p$) Thus $|W \cap V_1| = 1$. Hence changing any $p-2$ elements of V_1 we cannot get a radius smaller than K. Thus $\sup\limits_{H} E^c_{p-2,\,p}(H) \geq K$ for any $K>0$. □

It is natural to suppose that for a network H and p fixed, we get a better approximate solution of we increase the value of r i.e. we change more vertices in each step. (In the terminology of local search algorithms (see [4]) increasing the value of r means that we are searching a larger neighbourhood at each step.) The following example shows that this intuition is false.

Example. Let H be the unweighted network given on Figure 1. (the edge-weights are given encircled). We want to find a 6-centrum of H starting from $V_1 = \{1, 2, \ldots, 6\}$. If $r = 2$ we get the 2-optimal solution {1, 2, 3, 9, 10, 13} with radius 6. If $r = 1$ we get the 1-optimal solution {1, 2, 10, 12, 14, 19} with radius 5.

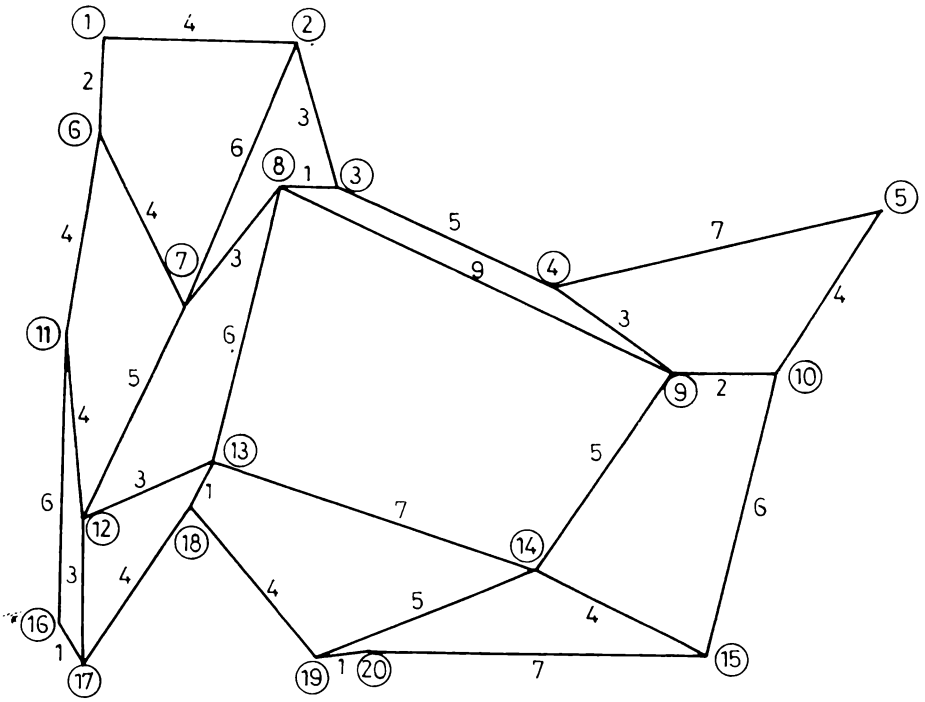

Figure 1.

3. A version of the p-center problem for trees

Let be given a network the graph of which is a tree and p disjoint subtrees. We want to locate p centers by placing one center in each subtree optimally. Formally the problem is the following.

Given $H = (G, a, s)$, a network, where G is a tree, $G_1, \ldots, G_p$ disjoint subtrees of G.

Find $A = \{v_1, \ldots v_p\}$ with v_i a vertex of G_i $1 \leq i \leq p$ s.t. $d(V, A) = \min\{d(V, B)\colon B = \{w_1, \ldots w_p\}, w_i \text{ is a vertex of } G_i \text{ for } 1 \leq i \leq p\}$.

The algorithm performs binary search over the possible values of the radius r, and for a given value of r it locates centers by proceeding towards the root from the leaves of the tree, controlling distance and the subtrees given simultaneously. (The principle of processing the tree "inwards" is similar to the algorithm of Kariv-Hakimi [2] for p-centers.)

Given a network, we choose a root arbitrarily, and denote by v a vertex of it.

Notation: $n(v)$ – the level of v (the level of the root is 0),
$F(v)$ – the ancestor of v.
$H(v)$ – the subtree with root v.

We shall use the following variables (with their intended meaning):

$J(v)$ – the set of vertices assigned to v (the assignment of vertices is explained later),
$k(v)$ the number of vertices in $J(v)$,
$T(v)$ – the vector of distances of v and the vertices assigned to v,
$S(v)$ – the vector of weights of the vertices assigned to v,
$C(v)$ – the center already selected that is nearest to v,
$c(v)$: $= d(v, C(v))$,
r – the actual value of the supposed radius,
N – the actual level processed,
v – the actual vertex processed.

The algorithm.

1. Perform binary search over the set

$$\{d(v, w)\cdot s(v) : v, w \text{ vertices of } G,\ v \neq w\}.$$

2. (Initial values) For all $v \in V$:

$$J(v) := \{v\}$$
$$k(v) := 1$$
$$T(v; 1) := 0$$
$$S(v; 1) := s(v)$$
$$c(v) := \infty$$
$$N := \max\{n(v) : v \in V\}.$$

3. If all vertices on level N are processed then:

if $N = 0$, then go to 1.
else $N := N-1$

else $v :=$ an arbitrary unprocessed vertex on level N.

4. If v is the root of a subtree G_i then if G_i does not contain yet a center then v is assigned as center and $c(F(v)) := \min\{c(F(v)), d(v, F(v))\}$, go to 3.

5. Let $I_1 := \{t : (c(v)+T(v; t))\cdot S(v; t) \leq r\}$.

6. If $|I_1| = k(v)$ then $c(F(v)) := \min\{c(F(v)), c(v)+d(v, F(v))\}$ go to 3.

Else if there is a $t \in \{1, \ldots, k(v)\} - I_1$ with
$[d(v, F(v))+T(v; t)]\cdot S(v; t) > r$ then

if v is a vertex of a subtree G_i and G_i does not contain yet a center then v is assigned as center and

$c(F(v)) := \min\{c(F(v)), d(v, F(v))\}$, go to 3.

If v is a vertex of a subtree G_i and G_i contains a center already or v is not a vertex of any subtree G_i then go to 1. (There is no solution for r)

7. (Assignment to $F(v)$)

$c(F(v)) := \min\{c(F(v)), c(v)+d(v, F(v))\}$

$k(F(v)) := k(F(v))+k(v)-|I_1|$

for $t \in \{1, 2, \ldots, k(v)\} - I_1$:

$S(F(v); .) := S(v; t)$

$T(F(v); .) := T(v; t)+d(v, F(v))$

and go to 3.

Theorem 3. *The algorithm is correct.* □

Proof. It is sufficient to show that in the case when the algorithm does not find a solution for a given r, there is no solution for this r.

Let N denote the depth of the tree: The case $N = 0$ is trivial. In the case $N = 1$ the algorithm does not find a solution only, when

(1) every subtree G_j is a vertex and there is a vertex $v \in V$ with

$s(v) \cdot d(v, \cup G_j) > r$

(2) the depth of a subtree G_{j_0} is 1 (the other subtrees are leaves) and

a) the algorithm assigned a leaf v_1 of G_{j_0} as a center and there is a vertex $v \in V$ with

$s(v) \cdot d(\bigcup_{j \neq j_0} G_j \cup \{v_1\}, v) > r$

b) there is a $v \in V - \cup G_j$ with $s(v) \cdot d(v_0, v) > r$ (v_0 is the root of the original tree)

It is easy to see that in these cases no solution exists.

Suppose there exists a network $H = (G, a, s)$, $p \in N^+$, $r > 0$ and disjoint subtrees $G_1, \ldots, G_p$ s.t. there is a solution of the corresponding problem but the algorithm does not find any. Assume H is such a network with $\sum_{v \in V} n(v)$ minimal. The depth of the tree is ≥ 2. Let $v_1 \in V$ be a vertex of maximal level and $v_2 = F(v_1)$, $v_3 = F(v_2)$.

In the subtree rooted at v_2 certain vertices are selected as centers by the algorithm, others are assigned to v_3. Denote centers by $c_1, \ldots c_t$, vertices assigned to v_3 by $y_1, \ldots, y_s$.

Now let us construct the following network that is obtained from H by deleting the subtree rooted at v_2 and edge (v_2, v_3) and adding new vertices

$\bar{c}_1, \ldots, \bar{c}_t, \bar{y}_1, \ldots, \bar{y}_s$ and edges $(v_3, \bar{c}_1), \ldots, (v_3, \bar{c}_t), (v_3, \bar{y}_1), \ldots, (v_3, \bar{y}_s)$ with edge weights $a(v_3, \bar{c}_i) = d(v_3, c_i)$ $(i = 1, \ldots, t)$, $a(v_3, \bar{y}_i) = d(v_3, y_i)$ $(i = 1, \ldots, s)$ and vertex weights $s(\bar{c}_i) = s(c_i)$ $(i = 1, \ldots, t)$, $s(\bar{y}_i) = s(y_i)$ $(i = 1, \ldots, s)$. (See Fig. 2.)

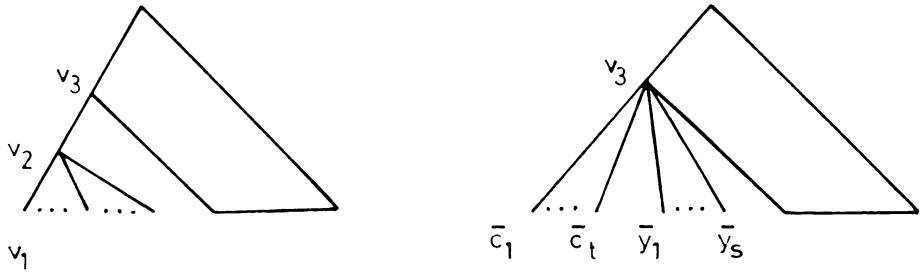

Figure 2.

We define the new subtrees on this network as follows: if $c_i \in G_k$ for some i then $\bar{G}_k = \{\bar{c}_i\}$ for $k = 1, \ldots, p$. Otherwise $\bar{G}_k$ is obtained from G_k by replacing each y_i contained in G_k by $\bar{y}_i$ and connecting it to v_3.

Then the modified network will have a solution and the algorithm will not find any. Further on the level sum of this network is strictly smaller then the original one. □

REFERENCES

[1] *N. Christofides*, Graph Theory – an Algorithmic Approach, Academic Press, New-York, London, San Francisco, 1955.

[2] *O. Kariv* and *S. L. Hakimi*, An algorithmic approach to network location problems, *SIAM J. Appl. Math.*, **37** (3) (1979), 513–521.

[3] *J. Krarup* and *P. M. Pruzan*, Selected Families of Discrete Location Problems, University of Copenhagen, Copenhagen, 1977.

[4] *C, H. Papadimitriou* and *K. Steiglitz*, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Englewood Cliffs (N. J.), 1982.