An Improved Algorithm for the Distance Constrained *p*-Center Location Problem with Mutual Communication on Tree Networks

Arie Tamir
School of Mathematical Sciences, Tel Aviv University, Ramat-Aviv, Haim Levanon St., 69978 Tel-Aviv, Israel

In a recent article Averbakh and Berman present an $O(p^3\sqrt{(\log\log p)}(\log p) + n^2)$ serial algorithm to solve the distance constrained *p*-center location problem with mutual communication on a tree network with *n* nodes. In this note we suggest two simple modifications leading to the improved (subquadratic in *n*), $O(p^3\sqrt{(\log\log p)}(\log p) + p(n + p)\log^2(n + p))$ complexity bound. We also present a new $O(p^2 n \log n \log(n + p))$ algorithm for the discrete version of this problem. © 2004 Wiley Periodicals, Inc. NETWORKS, Vol. 44(1), 38–40 2004

Keywords: multifacility location; *p*-center; tree networks

1. INTRODUCTION

Given is an undirected connected tree graph $T = (V, E)$. Each edge $e \in E$ has a positive edge length, l_e. An edge is an image of a closed interval under a continuous bijective mapping, that is, a Jordan arc. However, for our purposes an edge $e = (v_r, v_s)$ is identified with an interval of length l_e so that we can refer to its interior points. An interior point is identified by its distances along the edge (interval) from the two nodes v_r and v_s. Let $A(T)$ denote the continuum set of points on the edges of T. We also view $A(T)$ as a connected set, which is the union of $|E|$ intervals, that is, we assume without loss of generality that $A(T)$ is embedded in the plane. The edge lengths induce a distance function on $A(T)$. For any pair of points $x, y \in A(T)$, we let $d(x, y)$ denote the length of $P(x, y)$, the unique simple path in $A(T)$ connecting x and y. $A(T)$ is a metric space with respect to the above distance function. We refer to $A(T)$ as the tree network induced by T and the edge lengths $\{l_e | e \in E\}$.

Viewing $V = \{v_1, \ldots, v_n\}$ as the set of customers, or *existing* facilities, we consider the problem of locating p servers, or *new* facilities, $X = \{x_1, \ldots, x_p\}$ in $A(T)$. The servers must satisfy distance constraints (upper bounds) with respect to each other and to customer sites. The objective is to minimize the maximum weighted distance between pairs of servers and pairs of servers and customers, subject to the above distance constraints. This problem is called the *distance-constrained* p*-center problem with mutual communication* on $A(T)$, and it is formally defined in [1, 6, 7, 13] as follows.

Given are four sets of nonnegative reals, $\{a_{i,j} : i = 1, \ldots, n;\ j = 1, \ldots, p\}$, $\{a'_{i,j} : i = 1, \ldots, n;\ j = 1, \ldots, p\}$, $\{b_{j,k} : j = 1, \ldots, p;\ k = 1, \ldots, p\}$, and $\{b'_{j,k} : j = 1, \ldots, p;\ k = 1, \ldots, p\}$.

1.1. The Constrained p*-Center Problem with Mutual Communication*

Minimize λ, subject to,

$$a'_{i,j}d(v_i, x_j) \le 1, \quad i = 1, \ldots, n; \quad j = 1, \ldots, p,$$

$$a_{i,j}d(v_i, x_j) \le \lambda, \quad i = 1, \ldots, n; \quad j = 1, \ldots, p,$$

$$b'_{j,k}d(x_j, x_k) \le 1, \quad j = 1, \ldots, p; \quad k = 1, \ldots, p,$$

$$b_{j,k}d(x_j, x_k) \le \lambda, \quad j = 1, \ldots, p; \quad k = 1, \ldots, p.$$

The *unconstrained version,* studied in [7, 13], corresponds to the case where

$$a'_{i,j} = 0, \quad \text{for } i = 1, \ldots, n; \quad j = 1, \ldots, p, \quad \text{and}$$
$$b'_{j,k} = 0, \quad \text{for } j = 1, \ldots, p; \quad k = 1, \ldots, p.$$

Polynomial serial algorithms for the unconstrained version are presented in [7, 13]. The first polynomial algorithm for the constrained model is given in [6]. Its complexity is $O(pn(n + p\log n) + p^3\log p)$. An improved serial algorithm of $O(n^2 + p^3\sqrt{(\log\log p)}(\log p))$ complexity has been recently presented in [1]. Its complexity reduces further to $O[n^2 + p^3\sqrt{(\log\log p/\log p)}]$, when applied to

Received June 2003; accepted March 2004
Correspondence to: A. Tamir; e-mail: atamir@post.tau.ac.il
DOI 10.1002/net.20012
Published online in Wiley InterScience (www.interscience.wiley.com).

the unconstrained version. Parallel NC-algorithms are also presented in [1].

In comparison, consider the classical weighted p-center problem, where there is no mutual communication between the p servers, and each customer is served by its respective nearest server. (All servers are identical and provide the same services.) The complexity of the best known algorithms to solve the classical model on tree networks is subquadratic in n [16, 17].

Motivated by the cases where the number of existing facilities, (customers), is significantly larger than the number of new facilities, (servers), our goal is to obtain algorithms for the above model with mutual communication, whose complexity is also subquadratic in n.

In this short technical note we observe that the preprocessing phase of the algorithm in [1] can be modified to improve the complexity of this algorithm to $O(p(n + p)\log^2(n + p) + p^3\sqrt{(\log\log p)(\log p)})$. We also present a new algorithm that solves the discrete version of the constrained model in $O(p^2 n \log n \log(n + p))$ time.

2. THE MODIFIED ALGORITHM

The algorithm in [1] is a serial parametric implementation of a parallel location scheme (PLS), following the general ideas in [14, 15]. The algorithm consists of a two-step auxiliary stage, and a three-step main procedure.

Step 0.1, the first step of the auxiliary stage, is based on computing distances on a general graph with p nodes. (The data of the graph is unrelated to the edge lengths and topology of the underlying tree T.) The total effort spent in executing Step 0.1 in [1] is $O(p^3\sqrt{(\log\log p)(\log p)} + p(n + p)\log^3 p)$. (This bound is attained by implementing the sophisticated methods in [20]; see the Remark on page 11 in [1].)

In Step 0.2, the second step of the auxiliary stage, the matrix of the distances between the nodes of T is computed in $O(n^2)$ time.

Finally, based on the assumption that the distance between any pair of nodes of T is now available in $O(1)$ time, the implementation of the main procedure takes $O(p(n + p)\log^2(n + p))$ time. (We do not suggest any changes in the three-step main procedure. Therefore, for the sake of brevity, we see no reason to introduce the notation and terminology required for the presentation of the main procedure. Instead, we refer the reader to [1] for the long and detailed description of the implementation of this procedure.)

To improve the total complexity, we suggest two modifications.

First, we note that the complexity of Step 0.1 can be improved to $O(p^3\sqrt{(\log\log p)(\log p)} + p(n + p)\log^2 p)$, by implementing the idea in Application (4) in [5].

Second, Step 0.2 is replaced by the following step:

Without loss of generality suppose that the underlying tree T is rooted at the node v_1. In $O(n)$ time compute the distances $\{d(v_1, v_i) : i = 1, \ldots, n\}$. Then use the $O(n)$ algorithm in [11, 12] to preprocess the rooted tree T and build the data structure to compute nearest common ancestors. (See also the more recent algorithms in [2–4, 10, 18, 19].) With this structure, given a pair of nodes, v_i, v_j, we can find their nearest common ancestor, say v_k, in $O(1)$ time. We then have $d(v_i, v_j) = d(v_1, v_i) + d(v_1, v_j) - 2d(v_1, v_k)$.

Because the distance between any pair of nodes of T is now available in $O(1)$ time, the implementation of the main procedure will still take $O(p(n + p)\log^2(n + p))$ time.

To conclude, the total complexity of the modified algorithm is $O(p^3\sqrt{(\log\log p)(\log p)} + p(n + p)\log^2 p + n + p(n + p)\log^2(n + p))$, which amounts to $O(p^3\sqrt{(\log\log p)(\log p)} + p(n + p)\log^2(n + p))$. The complexity reduces to $O[p^3\sqrt{(\log\log p/\log p)} + p(n + p)\log^2(n + p)]$ for the unconstrained version. (We note that the modification in Step 0.1 was mentioned already in [1]. However, the authors did not use it there because it would not improve the overall complexity bound of their algorithm. We have demonstrated above that this modification is quite useful when coupled together with our modified version of Step 0.2. Note that if we apply only the modification in Step 0.2, but not the one suggested for Step 0.1, the total complexity will be $O(p^3\sqrt{(\log\log p)(\log p)} + p(n + p)\log^3 p + p(n + p)\log^2(n + p))$. The latter complexity bound is strictly inferior, for example, when $c_1 n^{\alpha} \le p \le c_2 n^{1/2}$ for some appropriate constants c_1, c_2, and α.)

3. THE DISCRETE MODEL

In the constrained p-center problem with mutual communication defined above, the new facilities $X = \{x_1, \ldots, x_p\}$ can be established anywhere in $A(T)$. By the *discrete* version of the model we refer to the case where X must be a multisubset of nodes. We note that the results in [1], as well as the modifications presented above, can easily be adapted to the discrete version without affecting the complexity bounds. Specifically, the discrete model can also be solved in $O(p^3\sqrt{(\log\log p)(\log p)} + p(n + p)\log^2(n + p))$ time.

We now present a new solution approach that is applicable only to the discrete case.

In the discrete case we can easily identify a set of polynomial cardinality containing λ^*, the optimal value of the parameter λ. Specifically, λ^* is an element of the set $\Lambda = \Lambda_1 \cup \Lambda_2$, where

$$\Lambda_1 = \{a_{i,k} d(v_i, v_j) : i, j = 1, \ldots, n; k = 1, \ldots, p\},$$

and

$$\Lambda_2 = \{b_{k,l} d(v_i, v_j) : i, j = 1, \ldots, n; k, l = 1, \ldots, p\}.$$

From the definition, λ^* is the smallest value of $\lambda \in \Lambda$, such that there exists a multiset of nodes $X = \{x_1, \ldots, x_p\}$ satisfying all the constraints,

$$a'_{i,j}d(v_i, x_j) \leq 1, \quad i = 1, \ldots, n; \quad j = 1, \ldots, p,$$

$$a_{i,j}d(v_i, x_j) \leq \lambda, \quad i = 1, \ldots, n; \quad j = 1, \ldots, p,$$

$$b'_{j,k}d(x_j, x_k) \leq 1, \quad j = 1, \ldots, p; \quad k = 1, \ldots, p,$$

$$b_{j,k}d(x_j, x_k) \leq \lambda, \quad j = 1, \ldots, p; \quad k = 1, \ldots, p.$$

To test the feasibility of a given value of λ, we can use the $O(p(n + p))$ procedure in [7], adapted to the discrete case.

The cardinality of Λ is clearly $O(p^2n^2)$, but there is an efficient way to search over Λ without explicitly generating it. First, from the results in [9, 17] we conclude that the set Λ_1 can be represented as the union of $O(pn \log n)$ subsets, (monotone columns), such that for each value of t, the t-largest element in each one of these subsets can be obtained in $O(1)$ time. The total time to obtain this compact representation is $O(pn \log n)$, [9].

Let $D = \{d(v_i, v_j) : i, j = 1, \ldots, n\}$. For $k, l = 1, \ldots, p$, define the set $b_{k,l}D$ by

$$b_{k,l}D = \{b_{k,l}d(v_i, v_j) : i, j = 1, \ldots, n\}.$$

Thus, the set Λ_2 can now be represented as the union of $O(p^2)$ sets, where each one of them is obtained from D by multiplying all its elements by a fixed scalar. Moreover, from the results in [9, 17] we conclude that for each value of t, the t-largest element in $b_{k,l}D$ can be found in $O(n \log n)$ time.

Finally, to search λ^* in Λ, we apply the search procedure in [8], using the above representation of Λ as the union of $O(p^2 + pn \log n)$ subsets.

This search procedure has $O(\log|\Lambda|)$ iterations. The effort in each iteration is dominated by the time to compute some t-largest element in each one of the above $O(p^2 + pn \log n)$ subsets, and the $O(p(n + p))$ time needed to implement the feasibility test algorithm in [7]. Thus, the total effort spent in each iteration is $O(p^2n \log n + pn \log n + p(n + p))$.

Because $|\Lambda| = O(n^2p^2)$ the number of iterations is $O(\log(n + p))$. We conclude that λ^* can be identified in $O(p^2n \log n \log(n + p))$ time.

To summarize, we have presented two solution methods for the discrete model, with $O(p^3\sqrt{(\log \log p)(\log p)} + p(n + p)\log^2(n + p))$, and $O(p^2n \log n \log(n + p))$ complexity bounds, respectively. Note that the second method (associated with the latter bound) should be preferred when n, the number of customers, is relatively smaller than p, the number of servers, for example, when $p\sqrt{(\log \log p/\log p)} \geq c_3n \log n$ for some constant c_3.

REFERENCES

[1] I. Averbakh and O. Berman, Parallel NC-algorithms for multifacility location problems with mutual communication and their applications, Networks 40 (2002), 1–12.

[2] M.A. Bender and M. Farach-Colton, The LCA problem revisited, Proceedings of the 4th Latin American Theoretical Informatics Symposium (LATIN), 2000, pp. 88–94.

[3] O. Berkman and U. Vishkin, Recursive star-tree parallel data structure, SIAM J Comput 22 (1993), 221–242.

[4] O. Berkman and U. Vishkin, Finding level-ancestors in tree, J Comput System Sci 48 (1994), 214–230.

[5] R. Cole, Slowing down sorting networks to obtain faster sorting algorithms, J ACM 34 (1987), 200–208.

[6] E. Erkut, R.L. Francis, and A. Tamir, Distance constrained multifacility minimax location problems on tree networks, Networks 22 (1992), 37–54.

[7] R.L. Francis, T.J. Lowe, and H.D. Ratliff, Distance constraints for tree network multifacility location problem, Operat Res 26 (1978), 570–596.

[8] G.N. Frederickson and D.B. Johnson, The complexity of selection and ranking in $X + Y$ and matrices with sorted columns, J Comput System Sci 24 (1982), 197–208.

[9] G.N. Frederickson and D.B. Johnson, Finding k-th paths and p-centers by generating and searching good data structures, J Algorithms 4 (1983), 61–80.

[10] H.N. Gabow, J.L. Bentley, and R.E. Tarjan, Scaling and related techniques for geometry problems, Proceedings of the 16th ACM Symposium on Theory of Computing (STOC), 1984, pp. 135–143.

[11] D. Harel, A linear time algorithm for the lowest common ancestors problem, Proceedings of the 21st IEEE Symposium on Foundations of Computer Science (FOCS), 1980, pp. 308–319.

[12] D. Harel and R.E. Tarjan, Fast algorithms for finding nearest common ancestors, SIAM J Comput 13 (1984), 338–355.

[13] A. Kolen, Tree network and planar location theory, Center for Mathematics and Computer Science, Amsterdam, The Netherlands, 1986.

[14] N. Megiddo, Combinatorial optimization with rational objective functions, Math Operat Res 4 (1979), 414–424.

[15] N. Megiddo, Applying parallel computation algorithms in the design of serial algorithms, J ACM 30 (1983), 852–865.

[16] N. Megiddo and A. Tamir, New results on p-center problems, SIAM J Comput 12 (1983), 751–758.

[17] N. Megiddo, A. Tamir, E. Zemel, and R. Chandrasekaran, An $O(n \log^2 n)$ algorithm for the k-th longest path in a tree with applications to location problems, SIAM J Comput 10 (1981), 328–337.

[18] P. Powel, A further improved LCA algorithm, Technical Report TR90-01, University of Minneapolis, 1990.

[19] B. Shieber and U. Vishkin, On finding lowest common ancestors: Simplification and parallelization, SIAM J Comput 17 (1988), 1253–1262.

[20] T. Takaoka, A new upper bound on the complexity of the all shortest path problem, Informat Process Lett 43 (1992), 195–199.