JOURNAL OF ALGORITHMS 27, 259–268 (1998)
ARTICLE NO. AL970921

An $O(\log^* n)$ Approximation Algorithm for the Asymmetric p-Center Problem

Rina Panigrahy*

Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

and

Sundar Vishwanathan†

Department of Computer Science and Engineering, Indian Institute of Technology, Mumbai 400076, India

Received April 29, 1996; revised January 12, 1997

The input to the asymmetric p-center problem consists of an integer p and an $n \times n$ distance matrix D defined on a vertex set V of size n, where d_{ij} gives the distance from i to j. The distances are assumed to obey the triangle inequality. For a subset $S \subseteq V$ the radius of S is the minimum distance R such that every point in V is at a distance at most R from some point in S. The p-center problem consists of picking a set $S \subseteq V$ of size p to minimize the radius. This problem is known to be NP-complete.

For the symmetric case, when $d_{ij} = d_{ji}$, approximation algorithms that deliver a solution to within 2 of the optimal are known. David Shmoys, in his article [11], mentions that nothing was known about the asymmetric case. We present an algorithm that achieves a ratio of $O(\log^* n)$. © 1998 Academic Press

1. INTRODUCTION

The p-center problem is a canonical problem of the "facility location" type. Imagine that you are given a map of a city, along with the time it takes to reach point x from point y, for all important pairs of points x, y in the city. You have to decide where to place p facilities, say p fire

* E-mail: rinap@mit.edu.
† E-mail: sundar@cse.iitb.ernet.in.

0196-6774/98 $25.00

stations, so that any important point is reachable quickly from at least one of these fire stations. We will assume that the fire stations have to be located at one of the important points in the city. Because of the distributions in traffic density, or perhaps because of one-way streets, it is very likely that the time it takes to go from x to y is very different from the time it takes to travel from y to x. This is precisely the asymmetric p-center problem. If, however, the time taken to travel from x to y is the same as the time taken to travel from y to x, then it becomes an instance of the symmetric p-center problem or simply the p-center problem.

More formally, the input to the (asymmetric) p-center problem consists of an $n \times n$ distance matrix D and a positive integer p. An entry d_{ij} in the distance matrix defines the distance from point i to point j (in the example, the time taken to reach destination j from point i). We will assume that the distances obey the triangle inequality, that is, $d_{ij} + d_{jk} \geq d_{ik}$, for all i, j, k. For a set of points S, the radius of S is the minimum distance R such that every point is within a distance R of some point in S. As output to the p-center problem we require a set of p points, called the *centers*, with minimum radius.

The decision version of this problem has an additional parameter, a positive real number R. The question one asks is whether one can find a set C of p points, the centers, such that every other point is at a distance at most R from some point in C.

This problem is known to be NP-complete [5]. In fact, it remains NP-complete even when the distance matrix D is symmetric and when the distances are restricted to be either 1 or 2. It is then natural to ask how good a solution one can find in polynomial time. Before we discuss the substantial amount of work that has already been done, we introduce the notion of an *approximation algorithm* and related terminology. See [5] for more details.

An approximation algorithm for a problem, loosely speaking, is an algorithm that runs in polynomial time and produces an "approximate solution" to the problem. Let Π be a minimization problem. Let $\mathscr{A}$ be an algorithm for Π. For an instance I of Π, let $\mathscr{A}(I)$ denote the value of the output of $\mathscr{A}$ on input I and let $OPT(I)$ denote the value of the optimum solution. We define the absolute performance ratio for $\mathscr{A}$ to be $\inf\{r \geq 1: A(I)/OPT(I) \leq r$ for instances $I\}$. We will say that an algorithm is an α-approximation algorithm if it always delivers a solution to within a factor α of the optimum.

Hochbaum and Shmoys [6] give a 2-approximation algorithm for the p-center problem when D is symmetric. Dyer and Frieze [4] describe a different 2-approximation algorithm for the same problem. Hsu and Nemhauser [7] prove that unless P = NP, this is the best possible. This lower bound, however, does not hold for specific metrics, such as, for example, the L_2 norm. Slightly worse lower bounds, however are known. The exact complexity in these cases remains an interesting open problem.

We refer the reader to the article by Shmoys [11] for an excellent account of the current status of this problem (and many others!). His concluding remarks on the p-center problem are relevant to this work and we reproduce them below.

"A natural generalization of the problem is to relax the restriction that the distance matrix be symmetric. This turns out to be a non-trivial generalization and essentially nothing is known about the performance guarantee for this extension."

In this paper we give a bound of $O(\log^* n)$.

A closely related problem is one where the radius is fixed and one has to minimize the number of centers. This problem can be seen to be equivalent to the well-studied set cover problem. In fact, we do give this equivalence in Section 3 and exploit it in our algorithm.

The asymmetric case does turn out to be harder in other problems, too. The most celebrated such example is the traveling salesman problem. While for the symmetric case a $3/2$-approximation algorithm is known [5], for the asymmetric case the best known algorithm only achieves a factor of $O(\log n)$.

2. PRELIMINARIES

Our notation is standard. Consider n points with pairwise distance matrix $D = \langle d_{ij} \rangle$. Occasionally, for convenience, we will also use $d(i, j)$ instead of d_{ij}. For a point v, $I_d^+(v)$ will denote the set $\{x\colon d_{vx} \le d\}$ and $I_d^-(v)$ will denote the set $\{x\colon d_{xv} \le d\}$. We will say that a center v covers a point x within R if $d_{vx} \le R$. We will also say that v R-covers x. Extending this notation, we will say that a set of centers C covers a set A within R if for every $a \in A$ there is a $c \in C$ that covers a within R.

The set cover problem is the following:

Instance. Set $S, \mathscr{F}$, a collection of subsets of S, and a positive integer k.

Question. Does there exist an $\mathscr{F}' \subseteq \mathscr{F}$, $|\mathscr{F}'| \le k$, such that every element of S occurs at least once in some set in $\mathscr{F}'$?

The performance of the greedy heuristic for the set cover problem has been analyzed by many people [3, 9, 8]. The following theorem is well known.

THEOREM 1. *Assume that the optimal solution to the set cover problem above has a value p. Then the greedy algorithm that chooses a set which covers* *the maximum number of elements at each stage outputs a cover of size at most* $p(1 + \ln(n/p))$, *where* $|S| = n$.

We sketch two simple proofs. The first was told to the authors by Jaikumar Radhakrishnan. It is easy to see that when one picks the set that covers the most number of elements, the number of as yet uncovered elements drops by a factor of $(1 - 1/p)$. Hence, after having picked $p \ln(n/p)$ sets, the number of uncovered elements is at most p. The result follows.

For a more discrete proof, notice that when there are between kp and $(k+1)p$ elements yet to be covered, each set that we pick contains at least k elements. Hence these p elements are covered using at most p/k sets. A simple summation yields that the total number of sets picked is (roughly) at most $pH(n/p)$.

3. THE ALGORITHM

We will assume that we know the optimum radius R. This is not a serious problem since there are just $O(n^2)$ possible values for R and we can run our algorithm for each of these values and choose the best solution.

Motivation and Techniques

We begin with some motivation and a brief sketch of one ingredient of the algorithm. We follow this up with a detailed description.

One crucial ingredient of the algorithm is the repeated use of the greedy set cover heuristic. We first indicate how this is useful for our problem. Along the way we mention that this gives an $O(\log n)$ approximation algorithm for our problem.

So suppose that the optimum radius R is given. Consider the related problem of finding the minimum number of points needed to R-cover V. This problem, as we mentioned in the Section 2, can be formulated as a set cover problem. The elements correspond to the points in V. For every point x, we add a set consisting of elements such that the corresponding points are covered by x. These are points that are at a distance at most R from x.

It is easy to see that a set cover corresponds to a set of points that cover V and vice versa. Using the greedy set cover heuristic, we can find a set S of at most $p \log(|V|/p)$ points that R-cover V. Now, suppose we find q points that R'-cover S. Then these q points $R + R'$-cover V.

To find such points to cover S, we again invoke the set cover heuristic. We now have an instance of a set cover on $|S|$ elements and $|V|$ sets. We have a set for each point v of V, consisting of points in S that are covered by v. Applying the greedy heuristic, we now get at most $p \log(|S|/p) = p \log\log(|V|/p)$ points that R-cover S and hence $2R$-cover V. Note that though we now only have a $2R$-cover of V, we have drastically reduced the number of centers.

We can apply this procedure repeatedly until the number of centers falls to p. It can be shown, for instance, through an easy extension of the proof given in the next section, that for this to happen the number of iterations is at most $O(\log p + \log^* n)$. Notice that this immediately gives an $O(\log n)$ approximation algorithm for the problem. We will, however, be interested in the fact that if the process is iterated $O(\log^* n)$ times we get $2p$ points that $O(R \log^* n)$-cover V.

A Formal Description

During the course of the algorithm we would have picked some vertices as centers. These would cover some vertices. The set A of vertices that have not yet been covered will be called *active*.

Our algorithm will have two phases. It is in phase 2 that we will repeatedly use the greedy set heuristic. For phase 2 we will need the following theorem.

THEOREM 2. *Suppose you are given $A \subseteq V, p, R$, so that A has an R-cover of size p. Also assume that there is a set $\mathscr{C}_1$ of vertices that R-cover $V \setminus A$. Then one can find in polynomial time a set of $2p$ vertices $\mathscr{C}_2$ that together with $\mathscr{C}_1$ cover A (and hence V) to within a radius of $O(R \log^* |A|)$.*

A discussion of a special case of the theorem, with $A = V$, was discussed previously in the section on motivation and techniques.

Here is the algorithm to prove theorem 2.

Algorithm Recursive Cover (A, p, R)

{We assume that there are p vertices that can cover A to within a radius of R.}

$A_0 \leftarrow A$.

$i \leftarrow 0$.

As long as $|A_i| > 2p$ repeat the following three steps:

1. Construct the following instance of set cover, $\langle S, \mathscr{F} \rangle$. Set $S = A_i$. There is one set $X \in \mathscr{F}$ for each point $x \in V$. X consists of all points in A_i, R-covered by x.

2. We now run the greedy set cover heuristic to get a set A'_{i+1} of points that R-cover A_i. $A_{i+1} \leftarrow A'_{i+1} \cap A$.

3. $i \leftarrow i + 1$.

Output A_i.

End Recursive Cover

Analysis. We argue by induction on i that $A_i \cup \mathscr{C}_1$ covers A to within a radius of $2iR$. Notice that A'_{i+1} covers A_i to within R. Let A_{i+1} R-cover H_i and let $B_{i+1} = A'_{i+1} \setminus A_{i+1}$ R-cover D_i, where $D_i = A_i \setminus H_i$. We note that $\mathscr{C}_1$ covers B_{i+1} to within R and hence covers D_i to within $2R$. Hence $A_{i+1} \cup \mathscr{C}_1$ $2R$-covers A_i and, by induction, $(2iR + 2R)$-covers A.

Also, $|A_{i+1}|$ is at most $O(p \log(|A_i|/p))$ and hence, by induction, at most $O(p \log^{(i)}(|A|/p))$. (We use the fact that $1 + \ln x \leq \log_a x$ for small enough a and large enough x.) This ensures that the size falls to $2p$ in $O(\log^*(|A|))$ iterations. We can continue using this procedure, but that is inefficient. It takes a further $\log p$ iteration to decrease the size to p.

It would seem that once we get down to $2p$ vertices, perhaps, we can work with these efficiently and get a cover. But this does not seem possible. We will use RecursiveCover; but as a *last step* of our algorithm.

To describe our algorithm, we need the following notion. A vertex u will be called a *center capturing vertex* (CCV) if $\Gamma_R^-(v) \subseteq \Gamma_R^+(v)$. It is easy to see that if a vertex v is a CCV then either it is a center or there is a center in $\Gamma_R^+(v)$.

We begin with an informal description of the algorithm.

There are two phases to the algorithm: the find or halve phase and the augment phase. In the first phase we repeatedly look for a CCV u, and include it in our cover as long as we can find one. We also remove from the active set every vertex covered within a radius of $2R$ from u.

We enter the second phase if we are left with some vertices to cover, none of which are CCVs. In other words, we are left with a nonempty set of active vertices A, none of which is a CCV. The combinatorial crux of this work is in proving that in such a case, roughly, if p vertices R-cover the vertices in A, then there exist $p/2$ vertices that $5R$-cover them. Hence, in a way, we have set ourselves up for an application of Recursive-Cover.

Here are the formal details.

Algorithm Approximate p-Center
Input is the distance matrix, the number of centers to pick p, and the optimum radius R.
The Find or Halve phase
$A \leftarrow V$
Repeat the following steps as long as there is a CCV, $v \in A$, and $p > 0$.

1. $\mathscr{C}_1 \leftarrow \mathscr{C}_1 \cup \{v\}$

{we include v as a center}

2. $A \leftarrow A \setminus \Gamma_{2R}^+(v)$

{we remove from the active set vertices covered to within a radius of $2R$ by v}

3. $p \leftarrow p - 1$

{note that the remaining vertices can be covered with one less center}

End the Find or Halve phase
The Augment phase
Set $A' \leftarrow A \setminus \Gamma_{5R}^{+}(\mathscr{C}_1)$.
If $|A'| \neq \varnothing$ then run *RecursiveCover*$(A', p/2, 5R)$ to get centers $\mathscr{C}_2$ of size p that together with $\mathscr{C}_1$ covers A within a radius of $O(R \log^* |A'|)$.
End the Augment phase
Output $\mathscr{C} = \mathscr{C}_1 \cup \mathscr{C}_2$.
End Algorithm

4. ANALYSIS

This section is devoted to the proof of the following theorem.

THEOREM 3. *Algorithm Approximate p-Center outputs a set of size p that covers V to within a radius of* $O(R \log^* |V|)$.

Consider each execution of the *find or halve* phase. Suppose v is a CCV. Then either v is a center or there is some center in $\Gamma_R^{+}(v)$. To see this, consider the center c that covers v. Since v is a CCV, $c \in \Gamma_R^{+}(v)$. So at the end of the find or halve phase it is clear that every vertex not in A is $2R$-covered by vertices in $\mathscr{C}_1$. We prove next that if we reach the augment phase then there exist $p/2$ vertices that together with $\mathscr{C}_1$ $5R$-cover A'. (Note that the value of p now is that specified at the end of the find or halve phase. Hence, for instance, if this value of p is 0 then all vertices must be covered at the end of the first phase.)

We begin with a simple combinatorial lemma.

LEMMA 4. *Let* $D = \langle U, F \rangle$ *be a digraph. Then there is a subset* $W \subseteq U$, $|W| \leq |U|/2$ *such that every vertex within degree at least* 1 *is reachable in at most two steps from some vertex in* W.

Proof. (Induction on $|U|$). The basis, $n = 2$, is trivial. Also note that if the arc set F is empty we are done. For the inductive set, pick any vertex u with nonzero outdegree. Remove u and its neighbors. Since u has nonzero outdegree we reduce the size of the graph by at least 2. Hence, by induction, there is a set of size at most $(n-2)/2$, which works for the remaining graph. Now, the only vertices in the original graph not covered by this set could be u, some of its neighbors, or perhaps vertices at distance 2 from u. This last case could occur if the indegree of some vertex at distance 2 from u fell to 0 while removing neighbors of u. Hence adding u to this set gives us a set of the required size for the original graph. ∎

The next lemma is crucial to the working of the algorithm. We note that as long as we keep finding CCVs, we keep removing centers. The next lemma explains why the augment phase works.

LEMMA 5. *Consider a subset of the vertex set A that has an R-cover consisting of p vertices. Assume that we have already picked a set $\mathscr{C}_1$ of vertices to $2R$-cover the vertices in $V \setminus A$. Suppose there are no CCVs in A. Then there are $p/2$ vertices $\mathscr{C}_2$ that together with $\mathscr{C}_1$ $5R$-cover A.*

Proof. Let $x_1, \ldots, x_p$ be the centers that form a R-covering of A. We partition these centers into three types:

1. centers in $V \setminus A$,
2. centers $x \in A$ such that $\Gamma_{2R}^-(x) \cap (V \setminus A) \neq \emptyset$,
3. centers $x \in A$ such that $\Gamma_{2R}^-(x) \cap (V \setminus A) = \emptyset$.

We note that the centers outside A are already $2R$-covered by the set $\mathscr{C}_1$ of vertices that we have picked. Hence the vertices that these centers cover are covered by vertices in $\mathscr{C}_1$ within a radius of $3R$.

Consider any vertex covered by some center z of type 2. Since z is of type 2 there is a $u \in V \setminus A$ such that $d_{uz} \leq 2R$. But u is $2R$-covered by $\mathscr{C}_1$. Hence $\mathscr{C}_1$ $5R$-covers the points covered by z.

Summarizing, $\mathscr{C}_1$ $5R$-covers all points except points covered by centers of type 3. Hence, for the proof of the lemma, we need to prove the existence of $p/2$ points that $5R$-cover the points R-covered by centers of type 3.

Without loss of generality, let $x_1, \ldots, x_q$ be the centers of type 3 and $x_{q+1}, \ldots, x_s$ be the centers of type 2. Since the vertices $x_1, \ldots, x_q$ are not CCVs, there are vertices $y_1, \ldots, y_q \in V$ (perhaps not all distinct) such that $d_{y_i x_i} \leq R$ while $d_{x_i y_i} > R$. We note that $y_i \in A$, for otherwise x_i would be a type 2 center. Consider some index $j \leq q$. Since x_j is of type 3, y_j is necessarily covered by some center u in A. Suppose to the contrary that some center $z \in V \setminus A$ covers y_j. Then $d(z, x_j) \leq d(z, y_j) + d(y_j, x_j) \leq 2R$, which contradicts the fact that x_j is a type 3 center. Moreover, this center u is distinct from x_j since $d_{x_j y_j} > R$.

We claim that at most $s/2$ points from $x_1, \ldots, x_s$ suffice to $5R$-cover the vertices R-covered by $x_1, \ldots, x_q$. In proof, consider the following auxiliary digraph B on s vertices, say $z_1, \ldots, z_s$. There is an arc from z_i to z_j if and only if y_j is covered by center x_i. Lemma 4 applies and it outputs a subset $\{z_{i_1}, \ldots, z_{i_m}\}$ of size at most $s/2$. We now verify that $x_{i_1}, \ldots, x_{i_m}$ form a $5R$-cover of the vertices R-covered by $x_1, \ldots, x_q$. To see this, consider any vertex u R-covered by a center x_j, $j \leq q$. In the auxiliary digraph B, z_j will have indegree at least 1 since there is some x_i, $i \neq j$, that covers y_j. Hence there is some z_{i_h} such that z_j is reachable in at most two steps from it. We claim that u is $5R$-covered by x_{i_h}. So, consider a path $z_{i_h} \to z_p \to z_j$ in B. (The case when the path is of smaller length is handled analogously.) This means $d(x_{i_h}, y_p) \leq R$ and $d(x_p, y_j) \leq R$. We also know that $d(y_p, x_p) \leq R$ and $d(y_j, x_j) \leq R$. Hence $d(x_{i_h}, x_j) \leq d(x_{i_h}, y_p) + d(y_p, x_p) + d(x_p, y_j) + d(y_j, x_j) \leq 4R$. ∎

The proof of Theorem 3 can now be inferred from the preceding discussion and Theorem 2.

5. CONCLUSION

The interesting question here is if there exists an algorithm with constant approximation ratio for the asymmetric p-center problem. Perhaps a more careful look at the relaxation of the combinatorial structure that occurs when one imposes asymmetry will lead to such an algorithm

However, for the theorist it would be more interesting if this were not the case! This would then furnish the first example of a natural problem whose approximability is a very slowly growing function of the input size. The Feige−Lovasz 2-Prover−1-Round proof systems were used ingeniously by Lund and Yannakakis [10] to prove the hardness of the set cover. Can these proof systems be used again in the present context?

ACKNOWLEDGMENTS

David Shmoys was a great help, with comments, and also for presenting an earlier version of this work.

We are grateful to Ajit Diwan and Howard Karloff for comments which improved the presentation. Also, but for Howard sending us a preprint of Shmoy's article, we would never have worked on this problem. Last, the second author thanks the University of Chicago for letting him use their computational facilities.

REFERENCES

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman, "The Design and Analysis of Computer Algorithms,'' Addison−Wesley, Reading, MA, 1974.
2. B. Bollobas, ''Combinatorics,'' Cambridge Univ. Press, 1986.
3. V. Chvatal, A greedy heuristic for the set covering problem. *Math. Oper. Res.* 4 (1979), 233−235.
4. M. Dyer and A. Frieze, A simple heuristic for the p-center problem, *Oper. Res. Lett.* 3 (1991), 285−288.
5. M. R. Garey and D. Johnson, ''Computers and Intractability: A Guide to the Theory of NP-Completeness,'' Freeman, San Francisco, 1979.
6. D. S. Hochbaum and D. B. Shmoys, A best possible approximation algorithm for the k-center problem, *Math. Oper. Res.* 10 (1985), 180−184.

7. W. L. Hsu and G. L. Nemhauser, Easy and hard bottleneck location problems, *Discrete Appl. Math*. 1 (1979), 209–216.
8. D. S. Johnson, Approximation algorithms for combinatorial problems, *J. Comput. System Sci*. 9 (1978), 256–278.
9. L. Lovasz, On the ratio of optimal integral and fractional covers, *Discrete Math*. 13 (1975), 383–390.
10. C. Lund and M. Yannakakis, On the hardness of approximating minimization problems, *in* "Proceedings of the ACM STOC,'' pp. 286–293, 1993.
11. D. B. Shmoys, Computing near-optimal solutions to combinatorial optimization problems, *in* ''Advances in Combinatorial Optimization'' (W. Cook, L. Lovasz, and P. Seymour, Eds.), Amer. Math. Soc., Providence, pp. 355–397, 1995.
12. R. E. Tarjan, "Data Structures and Network Algorithms,'' SIAM, Philadelphia, 1983.