

# Faster Dynamic Programming Algorithms for Facility Location

Jonathan Lenchner\*

## 1 Introduction

We study the following 1D and 1.5D problems: Given a set of  $m$  server locations  $\{t_1, \dots, t_m\}$ , lying along the  $x$ -axis and specified in increasing order, and  $n$  client locations  $\{p_1, \dots, p_n\}$ , also specified in increasing order along the  $x$ -axis and either located themselves on the  $x$ -axis (1D), or in the plane (1.5D), and a cost of broadcasting to client locations  $f(r)$  which is a function of the broadcast radius, we seek to determine the servers to broadcast from, and radii to broadcast to, so as to minimize the total cost of reaching all clients. For cost functions we consider separately that of linear and supralinear cost, and since constants and lower order terms are irrelevant to order of magnitude analysis we consider  $f(r) = r^\alpha$  for  $\alpha = 1$  or  $\alpha > 1$ . Whether in 1D or 1.5D we say covering by disks or circles, despite the fact that 1D “disks” are really intervals. We distinguish the case in 1.5D where the server locations are pre-specified along the  $x$ -axis from the case where we are free to pick the server locations anywhere we chose, though still along the  $x$ -axis. We refer to the former as the discrete problem, and the latter as the continuous problem.

## 2 Background

The 1D problem for  $\alpha \geq 1$  was shown to have an exact  $O(n^2m)$  solution by Lev-Tov and Peleg [3] and a linear time 4-approximation for the case  $\alpha = 1$ . In [1], Alt et al. improved on the constant factor, giving a linear time 3-approximation and an  $O(m + n \log m)$  time 2-approximation. Zolotarev [4] cleverly extended the Lev-Tov and Peleg solution to also solve the discrete 1.5D problem in time  $O(n^2m)$ . Finally, the full discrete 2D problem has been studied in [3], [2] and [1] where various PTAS and NP-hardness results are given. In [1], Alt et al. studied the continuous 1.5D problem, giving an  $O(n^2 \log n)$  exact solution for  $\alpha = 1$  and an  $O(n^4 \log n)$  solution for  $\alpha > 1$ .

## 3 Results

In this work we give an exact  $O(n^2 + nm)$  time algorithm for the 1D problem for any  $\alpha \geq 1$  thus improving on the Lev-Tov and Peleg [3] exact solution. We also show how to adapt the Zolotarev (Lev-Tov and Peleg) discrete solution to obtain an exact  $O(n^4)$  time solution to the continuous problem for  $\alpha > 1$  thus shaving a  $\log n$  factor off the solution given by Alt et al. Our 1D algorithm is a variation on the algorithm for solving the “continuous” 1.5D problem for the case  $\alpha = 1$  in [1].

We begin by considering the problem in 1D for arbitrary  $\alpha \geq 1$ . Lev-Tov and Peleg’s algorithm works by filling out a matrix  $Cost[i][j]$  which gives the minimum cost of covering the first  $j$  clients by the first  $i$  servers. The matrix is filled out one row at a time. The first row is filled out by noting that

$$Cost[1][j] = \max(dist(t_1, p_1)^\alpha, dist(t_1, p_j)^\alpha),$$

and the remaining rows are filled out by observing that  $Cost[i][j]$  is the minimum over (i)  $Cost[i-1][j]$ , (ii)  $dist(t_i, p_j)^\alpha + Cost[i-1][k-1]$  where  $p_k$  is the left-most client captured by the disk of radius  $dist(t_i, p_j)$  about server  $t_i$ , and (iii)  $dist(t_i, p_l)^\alpha + Cost[i-1][l-1]$  for all  $l$  such that  $1 \leq l \leq j-1$  and  $dist(t_i, p_l) > dist(t_i, p_j)$ .

Our solution uses just a singly indexed array. We consider just “pinned” disks, which are disks with center at some server  $t_i$  and radius extending out to some client  $p_j$ . Clearly an optimal solution will consist entirely of pinned disks. As in [1], call the “owner” of a disk the right-most client contained in the disk. There are  $nm$  pinned disks and furthermore they can all be stored along with their left and right-most contained client as a set of  $n$  arrays, where each array contains the disks for which client  $p_i$  is owner. We can compute the disks centered at a given server, together with their right-most and left-most clients, in increasing order of their radii, by simply locating where the server is in the sorted list of clients and splitting that list at the server and merging both sublists in linear time. This takes  $O(n)$  time per server, hence  $O(nm)$  time overall.

With this preprocessing the algorithm fills out the singly indexed array  $Cost[i]$ . After each iteration through the main loop,  $Cost[k]$  for  $1 \leq k \leq i$  contains the minimum cost of covering clients up to  $p_k$  by disks owned by clients up to  $p_i$ . The pseudocode is given in Figure 1. The

---

\*IBM T. J. Watson Research Center, Yorktown Heights, NY 10598.  
lenchner@us.ibm.com.

```

MINIDCOVER :
for every pinned circle  $C$ 
  find the leftmost and rightmost points enclosed by  $C$ 
 $Cost[0] \leftarrow 0$ 
for  $i \leftarrow 1$  to  $n$ 
   $Cost[i] \leftarrow \infty$ 
  for each pinned circle  $C$  owned by  $p_i$ 
     $p_j \leftarrow$  leftmost point enclosed by  $C$ 
     $Cost[i] \leftarrow \min\{Cost[i], Cost[j-1] + radius(C)^\alpha\}$ 
   $k \leftarrow 1$ 
  while  $i > k$  &  $Cost[i-k] > Cost[i]$ 
     $Cost[i-k] \leftarrow Cost[i]$ 
     $k \leftarrow k+1$ 
return  $Cost[n]$ 

```

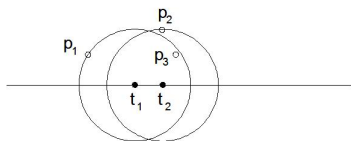
**Figure 1.** The 1D dynamic programming algorithm.

critical portion of this algorithm is the back-tracking steps that take place in the while loop at the end of each iteration. Backtracking is required to maintain the fact not just that  $Cost[i]$  contains the minimum cost of covering clients up to  $p_i$  with disks owned by clients up to  $p_i$ , but also that  $Cost[k]$  likewise contains the minimum cost of covering clients up to  $p_k$  with disks owned by clients up to  $p_i$  for  $1 \leq k \leq i$ . Without backtracking we could be in a position where  $Cost[i] < Cost[j]$  even though  $i > j$  and we later have to use this erroneous value of  $Cost[j]$ .

To check that at each step  $i$ , the invariants  $Cost[k]$ ,  $1 \leq k \leq i$  are properly maintained, we need verify that an optimal solution never contains two disks with the same owner. If  $C$  and  $C'$  were two disks in OPT with the same owner let  $C$  be the disk which extends furthest to the left (or either of the two disks if there is a tie). Then  $C$  contains all of the clients in  $C'$  so  $C'$  is superfluous. It follows that OPT can never contain two disks with the same owner and so MinIDCover is correct.

The two steps following for each pinned circle  $C$  owned by  $p_i$  get executed  $nm$  times - once for each pinned circle. However, for each of the  $n$  iterations of the outer loop, we may have to make up to  $n$  backtracking assignments. Hence the total running time is  $O(nm + n^2)$ .

In 1.5D, the above algorithm cannot be made to work, as is easiest to see for  $\alpha > 1$  where it is possible for two disks in OPT to have the same owner. See Figure 2.



**Figure 2.** An example for which OPT contains two disks with the same owner ( $p_3$ ) for sufficiently large  $\alpha$ .

In 1.5D the Zolotarev (Lev-Tov and Peleg) algorithm

requires a bit more work to fill in the first row:

$$Cost[1][j] = \max_{1 \leq k \leq j} (dist(t_1, p_1)^\alpha, dist(t_1, p_k)^\alpha),$$

and then to fill in  $Cost[i][j]$  we take the min over (i)  $Cost[i-1][j]$ , (ii)  $dist(t_i, p_j)^\alpha + Cost[i-1][k]$  where  $p_k$  is the right-most client to the left of  $p_j$  not captured by the disk of radius  $dist(t_i, p_j)$  about server  $t_i$ , and (iii)  $dist(t_i, p_l)^\alpha + Cost[i-1][k_l]$  for all  $l$  such that  $1 \leq l \leq j-1$  and  $dist(t_i, p_l) > dist(t_i, p_j)$  where  $p_{k_l}$  is the right-most client to the left of  $p_j$  not captured by the disk of radius  $dist(t_i, p_l)$  about server  $t_i$ . There is some work required to show that this algorithm is correct and that it can still run in time  $O(n^2m)$ . A naive implementation would require an additional  $O(n)$  time in the inner loop to determine “the right-most client to the left of  $p_j$  not captured by the disk of radius  $dist(t_i, p_l)$  about server  $t_i$ .”

We easily transform the above solution into a solution to the continuous 1.5D problem. Given a set of  $n$  clients in the continuous problem, it is easy to see that any disk in OPT will be determined by either one or two “pinning” clients which it will pass through. Taking all such choices from the  $n$  clients leads to  $O(n^2)$  candidate server locations. Zolotarev’s discrete algorithm with these servers and the original  $n$  clients then yields a solution to the continuous problem in time  $O(n^2m) = O(n^4)$ .

I thank Hervé Bronnimann and Joe Mitchell for helpful conversations.

## References

- [1] H. Alt, E. Arkin, H. Bronnimann, J. Erickson, S. Fekete, C. Knauer, J. Lenchner, J. Mitchell, and K. Whittlesey. Minimum-cost coverage of point sets by disks. *Proceedings of the 22nd Annual Symposium on Computational Geometry (SCG06)*, pages 449–458, 2006.
- [2] V. Bilò, I. Caragiannis, C. Kaklamanis, and P. Kanellopoulos. Geometric clustering to minimize the sum of cluster sizes. In *Proceedings of the 13th European Symposium on Algorithms*, volume 3669 of *Lecture Notes in Computer Science*, pages 460–471. Springer, 2005.
- [3] N. Lev-Tov and D. Peleg. Polynomial time approximation schemes for base station coverage with minimum total radii. *Computer Networks*, 47(4):489–501, 1985.
- [4] A. Zolotarev. On a client-server geometric cover by disks. Master’s thesis, Polytechnic University, June 2006.