An $O(p^2 log^2 n)$ Algorithm for the Unweighted p-Center Problem on the Line

Nimrod Megiddo and Arie Tamir

Raymond and Beverly Sackler
Faculty of Exact Sciences
School of Mathematical Sciences
Tel Aviv University
Ramat-Aviv, 69978 Israel

Let $V = \{v_1, ..., v_n\}$ be a set of points (customers) on the real line, where $v_1 < v_2 < ... < v_n$. Each point v_i, $i = 1, ..., n$, is associated with a positive weight w_i. The p-center problem is to locate p points (centers) on the line in order to minimize the maximum weighted distance of the customers to their respective nearest centers. Formally the problem is to

$$Minimize \max_{1\le i\le n} \min_{1\le j\le p} \{w_i|v_i - x_j|\}, \tag{1}$$

where $x_1, ..., x_p$ are real points.

By the discrete version of the problem we refer to the case where the points $x_1, ..., x_p$ are restricted to be in the set V. An optimal solution to the (discrete) p-center problem is called a (discrete) p-center. If all the weights are equal the above problem is called the unweighted p-center problem. There are several efficient algorithms to solve the above problem. Megiddo and Tamir (1983) presented an $O(n \log^2 n)$ algorithm for the problem, and Megiddo, Tamir, Zemel and Chandrasekaran (1981) gave an $O(n \log n)$ algorithm for the discrete version. The former can also be implemented in $O(n \log n)$ time by applying the modified search procedure in Cole (1987). Recently, Frederickson discovered an ingeniuos approach leading to an $O(n)$ algorithm for solving the unweighted p-center problem and its discrete version.

The above bounds are uniform and independent of p, the number of points to be selected. Since in most applications p is significantly smaller than n, we were motivated to find an algorithm whose complexity is sublinear in n. The cases where $p = 1, 2$ can easily be solved in $O(\log n)$ time. In this note we consider the case of a general p, and present an $O(p^2 \log^2 n)$ algorithm for the unweighted problems. (This algorithm was originally presented in an unpublished report in 1981.)

We assume that the sequence $v_1 < v_2 < \ldots < v_n$, is given by a linear array. Consider the unweighted version of (1), and suppose without loss of generality that $p < n$. Let r_p denote the optimal objective value. Given a positive real r we let $p(r)$ denote the smallest number of points (centers) needed in order to ensure that the distance of any point (customer) v_i, $i = 1, \ldots, n$, to its nearest center is at most r. We call r feasible for problem (1) if $p(r) \leq p$. In particular, r_p is the smallest feasible value. We start by presenting a simple $O(p \log n)$ algorithm for testing feasibility and then use it to find a p-center. (For convenience we define $v_{n+1} = \infty$.)

The Feasibility Test.

Given is a positive real r.

Step 0: Set $j = 1$, $p(r) = 0$, and $X = \emptyset$.

Step 1: Use a binary search to find a point v_i, $i \geq j$, such that $v_i \leq v_j + 2r < v_{i+1}$. Increase $p(r)$ by 1. Also augment the midpoint of the interval $[v_j, v_i]$ to the set X.

Step 2: If $p(r) > p$, stop: r is not feasible. If $i = n$, stop: r is feasible. Otherwise, set $j = i + 1$, and go to Step 1.

The effort to execute Step 1 is $O(\log n)$, and since the feasibility test has at most $p + 1$ iterations its complexity is clearly $O(p \log n)$.

We now present the algorithm for solving the unweighted p-center problem.

The p-Center Algorithm.

Step 0: Set $j = 1, k = 0$, $R_p = |v_n - v_1|/2$, and $X_0 = \emptyset$.

Step 1: Use a binary search, combined with the feasibility test, to find a point v_i, $i \geq j$, such that $|v_i - v_j|/2$ is not feasible but $|v_{i+1} - v_j|/2$ is feasible. Increase k by 1. If $k > p$, stop: R_p is the optimal value. Otherwise, set $R_p = \text{Min}\{R_p, |v_{i+1} - v_j|/2\}$. Let x_k be the midpoint of the interval $[v_j, v_i]$. Define $X_k = X_{k-1} \cap \{x_k\}$.

Step 2: If $i = n$, stop: R_p is the optimal value. Otherwise, set $j = i + 1$, and go to Step 1.

The effort to execute Step 1 is $O(p \log^2 n)$ since we have $O(logn)$ phases in the binary search, where each phase requires the feasibility test to resolve the query. The algorithm iterates at most $p + 1$ times, and therefore its total complexity is $O(p^2 \log^2 n)$.

The validity of the algorithm follows from the following argument. At each iteration k the recorded value of R_p is an upper bound on the optimal value r_p. Moreover, if the optimal value is smaller than R_p , then there is an optimal solution where the first k centers are established at the k points in X_k. The algorithm outputs the optimal value r_p. To find the optimal p-center apply the feasibility test with $r = r_p$. The resulting set X contains a p-center.

A similar procedure can be adapted to solve the discrete version of the unweighted model.

The Feasibility Test for the Discrete Case.

Given is a positive real r.

Step 0: Set $j = 1$, $p(r) = 0$, and $X = \emptyset$.

Step 1: Use a binary search to find a point v_i, $i \geq j$ such that $v_i \leq v_j + r < v_{i+1}$.

Increase $p(r)$ by 1. Also augment the point v_i to X. Then use a binary search to find a point v_t, $t \geq i$, such that $v_t \leq v_i + r < v_{t+1}$.

Step 2: If $p(r) > p$, stop: r is not feasible. If $t = n$, stop: r is feasible. Otherwise, set $j = t + 1$ and go to Step 1.

The Discrete p-Center Algorithm.

Step 0: Set $j = 1, k = 0$, $R_p = |v_n - v_1|$, and $X_0 = \emptyset$.

Step 1: Use a binary search, combined with the feasibility test, to find a point v_i, $i \geq j$, such that $|v_{i+1} - v_j|$ is feasible but $|v_i - v_j|$ is not. Increase k by 1. If $k > p$, stop: R_p is the optimal value. Use a binary search, combined with the feasibility test, to find a point v_t, $t \geq i$, such that $|v_{t+1} - v_i|$ is feasible but $|v_t - v_i|$ is not. Set $R_p = \ Min\ \{R_p, |v_{i+1} - v_j|, |v_{t+1} - v_i|\}$. Define $X_k = X_{k-1} \cap \{v_i\}$.

Step 2: If $t = n$, stop: R_p is the optimal value. Otherwise, set $j = t + 1$, and go to Step 1.

References

1) R. Cole, "Slowing down sorting networks to obtain faster sorting algorithms," *J. ACM* 34 (1987) 200-208.

2) G.N. Frederickson, "Optimal algorithms for partitioning trees and locating p-centers in trees," Technical Report, Department of Computer Science, Purdue University, 1990.

3) N. Megiddo and A. Tamir, Unpublished Report 1981.

4) N. Megiddo and A. Tamir, "New results on p-center problems," *SIAM J. on Computing* 12 (1983) 751-758.

4) 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. on Computing* 10 (1981) 328-337.