A Note on Minimum-Cost Coverage by Aligned Disks

Chan-Su Shin∗

Abstract

In this paper, we consider a facility location problem to find a minimum-cost coverage of \( n \) point sensors by disks centered at a fixed line. The cost of a disk with radius \( r \) has a form of a non-decreasing function \( f(r) = r^\alpha \) for any \( \alpha \geq 1 \). The goal is to find a set of disks under \( L_p \) metric such that the disks are centered on the x-axis, their union covers the \( n \) points, and the sum of the cost of the disks is minimized. Alt et al. [1] presented an algorithm in \( O(n^4 \log n) \) time for any \( \alpha > 1 \) under any \( L_p \) metric. We present a faster algorithm for this problem in \( O(n^2 \log n) \) time for any \( \alpha > 1 \) and any \( L_p \) metric.

1 Introduction

We consider geometric facility location problems of finding \( k \) disks whose union covers a set \( P \) of input points with the minimum cost. A center of the disk of radius \( r \) is often modeled as a base station(server) of transmission radius \( r \) and an input point as a sensor(client), so we assume the cost of the disk to be \( r^\alpha \) for some real value \( \alpha \geq 1 \). Thus the goal is to minimize \( \sum_i r^{\alpha}(D_i) \) where the disks \( D_i \) covering \( P \) have radius \( r(D_i) \). Alt et al. [1] presented a number of results on several problems related in this context. Among them, we focus on a restricted version in which the centers of the disks are restricted to be on a fixed line, simply saying x-axis. When the fixed line is not given, but its orientation is fixed, finding the best line giving the minimum coverage even for \( \alpha = 1 \) is quite hard to compute exactly [1], thus they gave a PTAS approximation algorithm.

Alt et al. [1] presented dynamic programming algorithms for this restricted coverage problem by aligned disks on a fixed line in time \( O(n^2 \log n) \) for \( \alpha = 1 \), and in time \( O(n^4 \log n) \) for any \( \alpha > 1 \) under any \( L_p \) metric for \( 1 \leq p < \infty \). For \( L_\infty \) metric, they presented an \( O(n^3 \log n) \)-time algorithm.

We reinterpret their dynamic programming algorithms together with new observations, then we present improved algorithms in \( O(n^2 \log n) \) time for any \( \alpha > 1 \) and any \( L_p \) metric, and in \( O(n^3) \) time for \( L_\infty \) metric. The number of disks in the optimal covering is automatically determined in the algorithm. If one would want to restrict the number of disks used, say as a fixed \( 1 \leq k \leq n \), then we can find at most \( k \) disks whose union covers the input points with minimum cost in a similar way. Actually we can find such \( k \) disks for all \( 1 \leq k \leq n \) in \( O(n^3 \log n) \) time in total.

The formal definition of the problem is as follows: Given a set \( P = \{p_1, p_2, \ldots, p_n\} \) of \( n \) points in the plane, a real value \( \alpha \geq 1 \) and \( L_p \) metric for some \( p \geq 1 \), find an optimal disks \( D_1, D_2, \ldots, D_k \) with centers \( s_i \) on the x-axis and with radii \( r(D_i) \) whose union covers \( P \) such that the sum of the radii, \( \sum_i r^\alpha(D_i) \) is minimized.

∗School of Electronics and Information Engineering, Hankuk University of Foreign Studies, Korea. Email: cssin@hufs.ac.kr. Supported by National Research Foundation Grant(NRF2011-002827).
2 Geometric properties

We assume that the line where the centers of the disks lie is x-axis. As mentioned in [1], we assume that all points in P lie above or on the x-axis and no two points have the same x-coordinates. If a point p is below the x-axis, we replace it with a new point p' mirroring p with respect to the x-axis, then we get the same optimal covering. If p is directly above p', then any disk containing p always contains p', so we can simply discard p from P. Thus from now on we assume that the points of P have nonnegative and distinct x-coordinates, and they are indexed from left to right. Finally we assume the points of P are in the general position, i.e., no three or more points lie on the boundary of a disk with centers on the x-axis.

We also notice that the optimal covering is not unique, so we assign the lexicographic order to the optimal covering, the set of the disks according to x-coordinates of their centers. Then we consider only the leftmost optimal covering $D = \{D_1, D_2, \ldots, D_k\}$ with centers in increasing order on x-axis.

Let $\alpha \leq 1$ and let $r(D_i)$ denote the radius of $D_i$. We call $r^\alpha(D_i)$ the cost of the disk $D_i$. For a while, let us consider $L_p$ metric only for $1 \leq p < \infty$. Let $\partial R$ denote the boundary of a closed region $R$. We denote by $t_i$ the highest point(or apex) of $\partial D_i$, and by $a_i$ and $b_i$ the left and right intersection points of $\partial D_i$ with the x-axis, respectively. Let $B$ be the union of disks in $D$. Then the following facts hold; the first one is mentioned also in [1].

Fact 1 [1] For each $1 \leq i \leq k$, the apex $t_i$ of $D_i$ appears on $\partial B$.

Let us consider $\partial D_i \cap \partial B$, i.e., the circular arc of $\partial D_i$ which appears on $\partial B$. By Fact 1, $t_i$ must be contained on the arc, so the arc is divided into the left and right subarcs at $t_i$. Then we have the following fact.

Fact 2 For each $1 \leq i \leq k$, $\partial D_i \cap \partial B$ must contain either one point of $P$ at the apex $t_i$ or two points of $P$, one on the left subarc and the other on the right subarc of $\partial D_i \cap \partial B$.

Proof. It is obvious that there must be at least one point of $P$ on $\partial D_i \cap \partial B$. Otherwise we can shrink $D_i$ to get a smaller cost until $\partial D_i$ contains some point. Also if one of the left and right arc has no points, then we can shrink $D_i$ while keeping the point on the one subarc until some point lie either on the apex $t_i$ or on the other subarc containing no points. This contradicts to the optimality. 

For each $1 \leq i < k$, we define $\ell_i$ as a vertical line between $D_i$ and $D_{i+1}$; if $D_i$ intersects $D_{i+1}$, then $\ell_i$ is a vertical line through intersections $\partial D_i \cap \partial D_{i+1}$, otherwise $\ell_i$ is an arbitrary vertical line between $b_i$ and $a_{i+1}$. For convenience, we define $\ell_0$ and $\ell_k$ as vertical lines passing through $a_1$ and $b_k$, respectively.

Let $P_i$ be a subset of points of $P$ lying between $\ell_{i-1}$ and $\ell_i$ for $1 \leq i \leq k$. Then we know that $P_i$ contains at least one point by Fact 2 and they are pairwise disjoint and their union is the same as the whole set $P$. Let $C_i$ be the smallest axis-centered disk containing $P_i$. Clearly $\{C_1, \ldots, C_k\}$ is a covering for $P$. We have the following lemma.

Lemma 1 $\sum_{1 \leq i \leq k} r^\alpha(C_i) = \sum_{1 \leq i \leq k} r^\alpha(D_i)$.

Proof. Since $\{D_1, \ldots, D_k\}$ is the optimal covering for $P$, it holds that $\sum_i r^\alpha(D_i) \leq \sum_i r^\alpha(C_i)$. For each $1 \leq i < k$, $P_i$ is contained in $P \cap D_i$, thus $r^\alpha(C_i) \leq r^\alpha(D_i)$. Since $f(r) = r^\alpha$ is a
nondecreasing function for $\alpha \geq 1$, $\sum_i r^\alpha(C_i) \leq \sum_i r^\alpha(D_i)$, which completes the lemma.

The above lemma means that there is a vertical partition of $P$ into $P_i$'s such that the smallest disks containing $P_i$'s are the optimal disks for $P$. Using this lemma, we can derive a fast dynamic programming algorithm.

3 Dynamic programming algorithm

Alt et al. [1] defined a pinned disk (or circle) as the leftmost smallest axis-centered disks enclosing some fixed subset of points, so the pinned disk contains at least one point on its boundary. The disk $C_i$ defined in Lemma 1 is a pinned disk. It is obvious that the optimal covering $D$ is a subset of such pinned disks. In [1], the dynamic programming algorithm chooses pinned disks with minimum cost from all $O(n^2)$ pre-computed pinned disks, satisfying the feasibility condition that no other points of $P$ lie above the chosen pinned disks. This step causes to take the total $O(n^4 \log n)$ time. But Lemma 1 tells us there must be a partition $P_1, \ldots, P_k$, separated by vertical lines, such that a set of the smallest disks containing $P_i$ is indeed an optimal covering for $P$. Thus we simply go through the input points from left to right, not through the pinned disks, and compute the smallest disk $C_i$ enclosing $P_i$ instead of checking the feasibility condition.

Let $A$ be an array in which $A[i]$ stores the minimum cost for a subset $\{p_i, p_{i+1}, \ldots, p_n\}$. The minimum cost for the whole set $\{p_1, \ldots, p_n\}$ will be stored at $A[1]$. If we denote by $D(\{p_i, \ldots, p_j\})$ the smallest disk containing $\{p_i, \ldots, p_j\}$, then we have the following recurrence relation:

$$A[i] = \begin{cases} \infty & \text{if } i > n, \\ \min_{i \leq j \leq n} \{A[j+1] + r^\alpha(D(\{p_i, \ldots, p_j\}))\} & \text{if } 1 \leq i \leq n. \end{cases}$$

The key step is to compute $D(\{p_i, \ldots, p_j\})$ fast. We can do this in amortized $O(\log n)$ time maintaining the intersection of the $x$-axis with the farthest Voronoi diagram (FVD) in a dynamic way. For a fixed $i$, $A[i]$ is computed in $O(n \log n)$ time, so the total time to compute $A[1]$ becomes $O(n^2 \log n)$.
As in Figure 1, the intersection of the farthest Voronoi diagram for \( \{p_1, \ldots, p_j\} \) with the \( x \)-axis partitions the \( x \)-axis into intervals \( I_1, I_2, \ldots, I_{m(i,j)} \) from the left to the right, where \( I_l \) is a half-open interval \( I_l := [x_{l-1}, x_l) \), where \( x_0 = -\infty \) and \( x_{m(i,j)} = +\infty \). Each interval \( I_l \) is a collection of the points from which the farthest point of \( \{p_1, \ldots, p_j\} \) is the same. We denote by \( p(I_l) \) the farthest point from any \( x \in I_l \). Then a disk centered at some point \( x \in I_l \) and with radius \( |xp(I_l)| \) encloses all the points of \( \{p_1, \ldots, p_j\} \).

Let \( D(I_l) \) be the smallest disk enclosing \( \{p_1, \ldots, p_j\} \) whose center lies in \( I_l \). We have two cases. For a case which \( \partial D(I_l) \) has one point at its apex, the point is indeed \( p(I_l) \) and the center of \( D(I_l) \) has the same \( x \)-coordinate as that of \( p(I_l) \). For the other case, \( \partial D(I_l) \) should have two points, so the center of \( D(I_l) \) must be on \( x_{l-1} \), the left endpoint of \( I_l \), but the radius of \( D(I_l) \) is defined as \( \infty \).

To store such intervals, we use a balanced search tree \( T \) [3]. We store at its leaves the intervals \( I_1, \ldots, I_{m(i,j)} \) with their corresponding radii from left to right. Each internal node \( v \) of \( T \) stores the minimum one among the radii in the leaves of the subtree rooted at \( v \). Then the radius stored at the root of \( T \) is the radius of the smallest disk enclosing \( \{p_1, \ldots, p_j\} \). We can insert a new interval into \( T \) and delete an interval from \( T \) both in \( O(\log n) \) time.

For a fixed \( i \), we now construct the intervals \( I_1, \ldots, I_{m(i,j)} \) for all \( i \leq j \leq n \) incrementally from \( j = i \) to \( j = n \). For \( j = i \), there is only one interval. We start with this interval, and update the interval set by adding the points one by one from \( p_{i+1}, \ldots, p_n \). We now explain how we update \( T \) for \( \{p_i, \ldots, p_{j-1}\} \) when \( p_j \) is inserted.

We know that the interval for \( p_j \) must appear because \( p_j \) is the rightmost point among \( p_i, \ldots, p_{j-1} \), and moreover the interval should be the leftmost one, i.e., its left endpoint must be \( x_0 = -\infty \). When the interval for \( p_j \) is inserted into \( T \), several consecutive intervals in \( T \) from the left should be removed from \( T \) or replaced with a shorter interval in \( T \). To identify such intervals, we need the following basic properties on the farthest Voronoi diagram.

**Lemma 2** For \( \{p_1, \ldots, p_j\} \) under any \( L_p \) metric, \( 1 \leq p < \infty \), the intersection of the \( x \)-axis with the farthest Voronoi diagram for \( \{p_1, \ldots, p_j\} \) has the properties: (1) The interval for \( p_j \) is connected, and (2) for any two consecutive intervals \( I \) and \( J \) where \( I \) is in the left of \( J \) on the \( x \)-axis, then \( p(I) > p(J) \), where \( p(I) > p(J) \) means the \( x \)-coordinate of \( p(I) \) is larger than that of \( p(J) \).

**Proof.** For the completeness, we prove these properties. A bisector of two points under any \( L_p \) metric is monotone to the \( x \)-axis and the \( y \)-axis, so it intersects the \( x \)-axis only once [4]. To prove the connectedness, we suppose that \( p_j \) has two disjoint intervals \( I \) and \( L \), where \( I \) is to the left of \( L \). There must be one or more intervals between them, denote by \( J \) the interval to the right of \( I \) and by \( K \) the interval to the left of \( L \). Note that \( J \) is not necessarily different with \( K \). Let \( D \) be a smallest disk centered at \( I \cap J \), i.e., the common endpoint of \( I \) and \( J \) which encloses all points in \( \{p_i, \ldots, p_j\} \). Then \( p(I) \) and \( p(J) \) lie on \( \partial D \). Similarly, let \( D' \) be a smallest disk centered at \( K \cap L \) enclosing all the points. Since \( p(I) = p(L) = p_j \), they must be on one of two intersections \( \partial D \cap \partial D' \), clearly the one above the \( x \)-axis. Also the lune \( D \cap D' \) contains all the points in \( \{p_i, \ldots, p_j\} \). This implies that \( p(J) \) must lie on the right boundary arc of the lune. The bisector of \( p(I) \) and \( p(J) \) intersects the \( x \)-axis at \( I \cap J \), thus the points on the \( x \)-axis to the left of \( I \cap J \) is farther to \( p(J) \) than to \( p(I) \), which contradicts that \( I \) is in the left of \( J \). For the second fact, we consider the half-circle of the smallest disk centered at \( I \cap J \) on the \( x \)-axis which passes through \( p(I) \) and \( p(J) \). Since the half-circle intersects with the bisector of \( p(I) \) and \( p(J) \) exactly once, \( p(I) \) should be in the right of \( p(J) \) along the
half-circle. This means \( p(I) > p(J) \) because the half-circle is monotone to the x-axis.

Let \( J = [a, b) \) be the interval of \( p_j \) in the interval set for \( \{p_1, \ldots, p_j\} \). Then we already know that \( a = -\infty \). By Lemma 2 it suffices to find the interval \( I_l \) from the intervals for \( \{p_1, \ldots, p_{j-1}\} \) which intersects with the bisector of \( p_j \) and \( p(I_l) \). Then \( b \) is the intersection of \( I_l \) with the bisector. For this, we do the intersection test from \( l = j - 1 \) to \( l = i \) one by one. Once \( I_l \) is found, we (1) delete the intervals \( I_1, \ldots, I_{l-1} \), which are completely contained in \( J \), from \( T \), (2) insert a new interval \( J \) for \( p_j \), and (3) replace(i.e., delete then insert) \( I_l \) with a part not contained in \( J, I_l \setminus J \). If some interval is removed from \( T \), then it is never inserted again into \( T \). Hence, for a fixed \( i \), we can compute the smallest disks enclosing disks for \( \{p_1, \ldots, p_j\} \) for all \( i \leq j \leq n \) in \( O((n-i) \log n) \) time. In other words, we can compute \( A[i] \) in \( O((n-i) \log n) = O(n \log n) \) for fixed \( i \). The total time of the algorithm is \( O(n^2 \log n) \), and the space is \( O(n) \). The detailed algorithm is summarized below.

**Algorithm 1 MinCostAlignedCoverage(\( P, \alpha \))**

**Input:** A set \( P \) of \( n \) points \( \{p_1, \ldots, p_n\} \) and \( \alpha \geq 1 \).

**Output:** A set of disks \( D = \{D_1, \ldots, D_k\} \) with minimum cost of \( \sum \alpha(D_i) \) which covers \( P \).

1. \( A[n + 1] = \infty \).
2. Initially, \( T \) consists of one interval \((-\infty, +\infty)\) for \( p_n \).
3. for \( i \leftarrow n \) to 1 do
4. \( A[i] = \infty \)
5. for \( j \leftarrow i \) to \( n \) do
6. Find the first interval \( I_l \) in \( T \) such that the bisector \( B \) of \( p_j \) and \( p(I_l) \) intersects \( I_l \) by scanning the intervals in \( T \) one by one from left to right
7. \( J := [-\infty, B \cap I_l] \)
8. Remove intervals \( I_1, \ldots, I_{l-1} \), replace \( I_l \) with \( I_l \setminus J \), and insert \( J \) in \( T \)
9. Let \( r \) be the radius stored at the roof of \( T \), i.e., \( r = r(D(\{p_i, \ldots, p_j\})) \)
10. \( A[i] = \min(A[i], \alpha + A[j + 1]) \)
11. Keep the index \( j \) which gives the minimum cost
12. end for \( j \)
13. end for \( i \)
14. Reconstruct the optimal disk set \( D \) by backtracking the recorded indices
15. return \( A[1] \) and \( D \)

**Algorithm for \( L_\infty \) metric.** Under this metric, the unit disk is an axis-aligned square. As before, we consider only the leftmost optimal covering by the lexicographic order. We can easily see that Fact 1 and Fact 2 can be applied for \( L_\infty \) metric if \( t_i \), the apex of the disk is defined as the upper and right corner of the disk. To use Lemma 2 we define a partition of \( P, P_1, \ldots, P_k \), separated by vertical lines containing right sides of the optimal disks. Then we can also prove in a similar way as the proof in Lemma 2 that the sum of the costs of the smallest squares \( C_i \) containing \( P_i \) is the same as the minimum cost for \( P \). We now compute \( A[i] \) similarly. The key step is to compute the smallest square \( C \) enclosing \( \{p_1, \ldots, p_j\} \) quickly. This square \( C \) is determined by two points; \( p_j \) and one of the points \( p_i \) and the highest point of \( \{p_i, \ldots, p_{j-1}\} \), which can be computed in \( O(1) \) time if we maintain the highest point during
the incremental evaluation. Thus we can compute $A[i]$ in $O(n)$ time. The total time is $O(n^2)$.

**Theorem 1** Given a set $P$ of $n$ points in the plane and a non-decreasing cost function with $\alpha \geq 1$, we can compute an optimal disks centered on the $x$-axis such that the union covers $P$ and the sum of the costs of the disks is minimized in $O(n^2 \log n)$ time for any fixed $L_p$ metric and in $O(n^2)$ time for $L_\infty$ metric.

We can also consider the case when the number of disks used to cover $P$ is given as a fixed value $k$. This case would be required by practical reasons. This can be similarly solved by filling a two dimensional table $A[i][k]$, the minimum cost needed to cover $p_i, \ldots, p_n$ with at most $k$ disks, in $O(kn^2 \log n)$ time. Actually we can find all optimal coverings for any $1 \leq k \leq n$ in the same time.

**Theorem 2** Given a set $P$ of $n$ points in the plane and a non-decreasing cost function with $\alpha \geq 1$, we can compute a collection of all optimal coverings for $P$ such that $P$ is covered by at most $k$ disks for any $1 \leq k \leq n$ and the sum of the costs of the disks is minimized in $O(n^3 \log n)$ time for any fixed $L_p$ metric and in $O(n^3)$ time for $L_\infty$ metric.

### 4 Concluding Remarks

We can consider other disk coverage problems with practical restrictions such as the connectivity constraint. Recently, Chambers et al. [2] investigated a problem of assigning radii to a given set of points in the plane such that the resulting set of disks is connected and the sum of radii, i.e., $\alpha = 1$ is minimized. When we bring such connectivity constraint to our problem for $\alpha \geq 1$, we need to find a “connected” set of disks centered on the $x$-axis whose union covers $n$ input points. When $\alpha = 1$, the smallest disk containing all points is the optimal coverage. However, we can easily show for $\alpha > 1$ that infinitely many disks always guarantee the minimum cost coverage for any input. Thus we should restrict the number of disks used to cover, say $1 \leq k \leq n$. But we have no idea how hard this problem is for a fixed $k$.

### References

[1] Helmut Alt, Esther M. Arkin, Hervé Brönnimann, Jeff Erickson, Sándor P. Fekete, Christian Knauer, Jonathan Lenchner, Joseph S. B. Mitchell, and Kim Whittlesey. Minimum-cost coverage of point sets by disks. In Proceedings of the twenty-second annual symposium on Computational geometry, SCG ’06, pages 449–458, New York, NY, USA, 2006. ACM.

[2] Erin Wolf Chambers, Sándor P. Fekete, Hella-Franziska Hoffmann, Dimitri Marinakis, Joseph S. B. Mitchell, Venkatesh Srinivasan, Ulrike Stege, and Sue Whitesides. Connecting a set of circles with minimum sum of radii. In Proceedings of the 12th international conference on Algorithms and data structures, WADS'11, pages 183–194, Berlin, Heidelberg, 2011. Springer-Verlag.

[3] Mark de Berg, Marc van Kreveld, Mark Overmars, and Otfried Schwarzkopf. Computational Geometry: Algorithms and Applications. Springer-Verlag, second edition, 2000.

[4] D. T. Lee. Two-dimensional voronoi diagrams in the $l_p$-metric. J. ACM, 27:604–618, October 1980.