The Online Median Problem*

Ramgopal R. Mettu   C. Greg Plaxton

Abstract

We introduce a natural variant of the (metric uncapacitated) k-median problem that we call the online median problem. Whereas the k-median problem involves optimizing the simultaneous placement of k facilities, the online median problem imposes the following additional constraints: the facilities are placed one at a time; a facility cannot be moved once it is placed, and the total number of facilities to be placed, k, is not known in advance. The objective of an online median algorithm is to minimize the competitive ratio, that is, the worst-case ratio of the cost of an online placement to that of an optimal offline placement. Our main result is a linear-time constant-competitive algorithm for the online median problem. In addition, we present a related, though substantially simpler, linear-time constant-factor approximation algorithm for the (metric uncapacitated) facility location problem. The latter algorithm is similar in spirit to the recent primal-dual-based facility location algorithm of Jain and Vazirani, but our approach is more elementary and yields an improved running time.

*Department of Computer Science, University of Texas at Austin, Austin, TX 78712. This research was supported by NSF Grant CCR–9821053. Email: {`ramgopal`, `plaxton`}`@cs.utexas.edu`. The second author is presently on leave at Akamai Techologies, Inc., Cambridge, MA 02139.

1 Introduction

Suppose we wish to open a new chain of stores in a city with n neighborhoods, and that we have a good estimate of the demand for our product in each neighborhood. In determining where to locate the stores, our high-level strategy is to minimize the *service cost* associated with our configuration of stores, which we define as the demand-weighted average distance from a customer to the nearest store. Our business plan is to start with one store, and then to gradually add new stores as allowed by our profits. (Remark: We will never move a previously established store.) Thus our configuration of stores may change over time, and hence the ratio between the service cost of our configuration and that of an optimal same-size configuration may also change. The goal of the *online median problem* is to choose a site for each new store so that the maximum value of this ratio is minimized. An online median algorithm that guarantees a ratio of at most r is said to achieve a *competitive ratio* of r, or to be r*-competitive*.

The variant of this problem in which the total number of stores to be built, k, is known in advance corresponds to the classic k*-median problem*. The k-median problem is known to be NP-hard and has been studied extensively over several decades (see, e.g., [17] for many pointers to the literature). Recently, Charikar *et al.* presented the first polynomial-time constant-factor approximation algorithm for the k-median problem [3]; even more recently, improved time bounds and approximation factors have been obtained by Charikar and Guha [2] and Jain and Vazirani [11].

Note that the online median problem can be viewed as the offline problem of determining a permutation of the n neighborhoods (specifying the order in which to build our stores) that minimizes the maximum ratio between the service cost of any prefix of the permutation and that of an optimal same-size configuration. We adopt this view throughout the remainder of the paper. Given the existence of constant-factor approximation algorithms for the k-median problem, it is natural to ask whether there is a constant-competitive algorithm for the online median problem. In other words, can we (efficiently) find a permutation of the n neighborhoods such that the service cost of any prefix of the permutation is at most a constant times that of an optimal same-size configuration? Note that, given an arbitrary problem instance, it is not clear *a priori* that such a permutation even exists.

In this paper, we affirm the existence of such a permutation and give a deterministic constant-competitive algorithm for the online median problem. Furthermore, the running time of our algorithm is $O(n^2 + \ell n)$ (where ℓ is the number of bits required to represent each distance), which is linear in the size of the input. While the main contribution of this paper is to identify and solve the online median problem, it worth noting that the k-median problem is a special case of the online median problem. Hence our linear-time online median algorithm is also the first linear-time constant-factor approximation algorithm for the k-median problem. (The best previous running time of $O((n^2 \log n)(\ell + \log n))$ is given in [11].)

An obvious approach to the online median problem is to iteratively choose the point that minimizes the objective function. Greedy strategies of this kind are commonly applied in the design of online algorithms [1, 10]. It turns out, however, that for the online median problem, the simple strategy suggested above has an unbounded competitive ratio. We show that a modification of this strategy that we call *hierarchically greedy* can be used to obtain a constant-competitive linear-time algorithm for the online median problem. We develop this strategy by first considering a simple greedy algorithm for facility location.

1.1 Problem Definitions

Fix a set of points U, a distance function $d : U \times U \to \mathbb{R}$, and nonnegative functions $f, w : U \to \mathbb{R}$. We assume throughout that d is a metric, that is, d is nonnegative, symmetric, satisfies the triangle inequality, and $d(x, y) = 0$ iff $x = y$. For the online median problem, it will prove to be useful to consider a slightly more general class of distance functions in which the triangle inequality is relaxed to the following "λ-

approximate" triangle inequality, where $\lambda \geq 1$: For any sequence of points $x_0, \ldots, x_m$ in U, $d(x_0, x_m) \leq \lambda \cdot \sum_{0 \leq i < m} d(x_i, x_{i+1})$. We refer to such a distance function as a λ-***approximate metric***. We let $n = |U|$, and define a subset of U to be a ***configuration*** iff it is nonempty. For any point x and configuration X, we define $d(x, X)$ as $\min_{y \in X} d(x, y)$.

We consider three computational problems: k-median, online median, and facility location. For the k-median and online median problems, the ***cost*** of a configuration, which we denote as $cost(X)$, is defined to be $\sum_{x \in U} d(x, X) \cdot w(x)$. The input to the k-median problem is (U, d), w, and an integer k, $0 < k \leq n$. The output is a minimum-cost configuration of size k. The input to the online median problem is (U, d) and w. The output is a total order on U. We define the competitive ratio of such an ordering as the maximum over all k, $0 < k \leq n$, of the ratio of the cost of the configuration given by the first k points in the ordering to that of an optimal k-median configuration. We define the ***competitive ratio*** of an online median algorithm as the supremum, over all possible choices of the input instance (U, d) and w, of the competitive ratio of the ordering produced by the algorithm.

For the facility location problem, the ***cost*** of a configuration, denoted $cost(X)$, is defined as the sum of $\sum_{x \in X} f(x)$ and $\sum_{x \in U} d(x, X) \cdot w(x)$. The input to the facility location problem is (U, d), f, and w. The output is a minimum-cost configuration.

1.2 Previous Work

There has been much prior work on the facility location and k-median problems. In this paper we focus on the metric versions of these problems; for recent work and pointers to the literature on the general (non-metric) facility location and k-median problems, see [19]. The first constant-factor approximation algorithm for facility location is due to Shmoys *et al.* [18] and is based on rounding the (fractional) solution to a linear program. Chudak [4] gives an LP-based $(1 + 2/e)$-approximation algorithm for facility location. This was the best constant factor known until the recent work of Charikar and Guha [2], which establishes a slightly lower approximation ratio of 1.728. The first constant-factor approximation for the k-median problem was recently given by Charikar *et al.* [3] and is also LP-based. That work follows a sequence of bicriteria results utilizing LP-based techniques [15, 16]. (These bicriteria results produce a configuration of size $O(k)$ with cost at most a constant factor times that of an optimal configuration of size k.) Jain and Vazirani [11] give the first nearly linear-time combinatorial algorithms for the facility location and k-median problems, achieving approximation ratios of 3 and 6, respectively. While the latter algorithms are combinatorial, the primal-dual approach used in their analysis is based on linear programming theory. (See [6] for an excellent introduction to the primal-dual method.)

Strategies based on local search and greedy techniques for facility location and the k-median problem have previously been studied. The work of Korupolu *et al.* [12] shows that a simple local search heuristic proposed by Kuehn and Hamburger [14] yields both a constant-factor approximation for the facility location problem and a bicriteria approximation for the k-median problem [12]. Guha and Khuller [7] showed that greedy improvement can be used as a postprocessing step to improve the approximation guarantee of certain facility location algorithms. Guha and Khuller also provide the best lower bound known of 1.463 on the approximation ratio for this problem. More recently, Charikar and Guha [2] achieved the best approximation ratio known for facility location by combining a local search heuristic with the best LP-based algorithm known. Charikar and Guha also give a 4-approximation for the k-median problem by building on the techniques of Jain and Vazirani [11].

1.3 Contributions

Algorithms for problems in discrete location theory arise in many practical applications; see [5, 17], for example, for numerous pointers to the literature. Given that many of these problems are NP-hard, it is desirable to develop fast approximation algorithms. As mentioned above, it is not uncommon for approximation algorithms to be based on a greedy approach. In this paper, we show that greedy strategies yield a fast constant-factor approximation algorithm for the facility location problem and a fast constant-competitive algorithm for the online median problem.

We give a linear-time algorithm for the facility location problem that achieves an approximation ratio of 3. The main idea of the algorithm is to compute and use the "value" of balls about every point in the metric space. In retrospect, the idea of value is implicit in the work of Jain and Vazirani [11]. We make this idea explicit and use the values of balls to make greedy choices. Additionally, our algorithm is faster than the Jain-Vazirani algorithm by a logarithmic factor.

While a simple greedy algorithm yields a constant-factor approximation bound for the facility location problem, it appears that a more sophisticated approach is needed to obtain a constant-factor approximation guarantee for the k-median problem, let alone a constant-competitiveness result for the online median problem. For example, in Section 3 we show that perhaps the most natural greedy approach to the k-median (resp., online median) problem leads to an unbounded approximation (resp., competitive) ratio.

Our main result is a linear-time constant-competitive algorithm for the online median problem. We achieve this result using a "hierarchically greedy" approach. The basic idea behind this approach is as follows: Rather than selecting the next point in the ordering based on a single greedy criterion, we greedily choose a region (the set of points lying within some ball) and then recursively select a point within that region. Thus, the choice of point is influenced by a sequence of greedy criteria addressing successively finer levels of granularity.

1.4 Outline

The rest of this paper is organized as follows. In Section 2, we present our facility location algorithm and prove that it achieves a constant approximation ratio. In Section 3, we present our online median algorithm and prove that it is constant-competitive. Section 4 offers some concluding remarks.

2 Facility Location

The following definitions are used throughout the present section as well as Section 3.

- For any nonnegative integer m, let $[m]$ denote the set $\{i \mid 0 \leq i < m\}$.
- A ***ball*** A is a pair (x, r), where the ***center*** x of A, denoted $center(A)$, belongs to U, and the ***radius*** r of A, denoted $radius(A)$, is a nonnegative real.
- Given a ball $A = (x, r)$, we let $Points(A)$ denote the set $\{y \in U \mid d(x, y) \leq r\}$. However, for the sake of brevity, we tend to write A instead of $Points(A)$. For example, we write "$x \in A$" and "$A \cup B$" instead of "$x \in Points(A)$" and "$Points(A) \cup Points(B)$", respectively.
- The ***value*** of a ball $A = (x, r)$, denoted $value(A)$, is $\sum_{y \in A}(r - d(x, y)) \cdot w(y)$.
- For any ball $A = (x, r)$ and any nonnegative real c, we define cA as the ball (x, cr).

2.1 Algorithm

In the first step of the following algorithm, we assume for the sake of convenience that there is at least one point x such that $w(x) > 0$. (The problem is trivial otherwise.) The output of the algorithm is the configuration Z_n, which we also refer to as Z. Remark: The indexing of the sets Z_i has been introduced solely to facilitate the analysis.

- For each point x, determine an associated ball $A_x = (x, r_x)$ such that $value(A_x) = f(x)$.
- Determine a bijection $\varphi : [n] \to U$ such that $r_{\varphi(i-1)} \leq r_{\varphi(i)}$, $0 < i < n$.
- Let $B_i = (x_i, r_i)$ denote the ball $A_{\varphi(i)}$, $0 \leq i < n$. Let $Z_0 = \emptyset$.
- For $i = 0$ to $n-1$: If $Z_i \cap 2B_i = \emptyset$ then let $Z_{i+1} = Z_i \cup \{x_i\}$; otherwise, let $Z_{i+1} = Z_i$.

We now sketch a simple linear-time implementation of the above algorithm. For each point x, the associated radius r_x can be computed in $O(n)$ time. (This is essentially a weighted selection problem.) Thus the first step requires $O(n^2)$ time. The second step involves sorting n values and can be accomplished in $O(n \log n)$ time. The running time for the third step is negligible. Each iteration of the fourth step can be easily implemented in $O(n)$ time, for a total of $O(n^2)$ time.

2.2 Approximation Ratio

In this section we establish the following theorem.

Theorem 1 *For any configuration X, $cost(Z) \leq 3 \cdot cost(X)$.*

Proof: Immediate from Lemmas 2.3 and 2.7 below. ■

Lemma 2.1 *For any point x_i, there exists a point x_j in Z such that $j \leq i$ and $d(x_i, x_j) \leq 2r_i$.*

Proof: If there is no such point x_j with $j < i$, then $Z_i \cap 2B_i$ is empty, and so x_i belongs to Z. ■

Lemma 2.2 *Let x_i and x_j be distinct points in Z. Then $d(x_i, x_j) > 2 \cdot \max\{r_i, r_j\}$.*

Proof: Assume without loss of generality that $j < i$. Thus $r_i \geq r_j$. Furthermore, $d(x_i, x_j) > 2r_i$ since x_j belongs to Z_i and $Z_i \cap 2B_i$ is empty. ■

For any point x and any configuration X, let

$$charge(x, X) \quad = \quad d(x, X) + \sum_{x_i \in X} \max\{0, r_i - d(x_i, x)\}.$$

Lemma 2.3 *For any configuration X, $\sum_{x \in U} charge(x, X) \cdot w(x) = cost(X)$.*

Proof: Note that

$$\begin{aligned} \sum_{x \in U} charge(x, X) \cdot w(x) &= \sum_{x_i \in X} \sum_{x \in B_i} (r_i - d(x_i, x)) \cdot w(x) + \sum_{x \in U} d(x, X) \cdot w(x) \\ &= \sum_{x_i \in X} value(B_i) + \sum_{x \in U} d(x, X) \cdot w(x), \end{aligned}$$

which is equal to $cost(X)$ since $value(B_i) = f(x_i)$. ■

Lemma 2.4 *Let x be a point, let X be a configuration, and let x_i belong to X. If $d(x, x_i) = d(x, X)$ then $charge(x, X) \geq \max\{r_i, d(x, x_i)\}$.*

Proof: If x does not belong to B_i, then $charge(x, X) \geq d(x, x_i) > r_i$. Otherwise, $charge(x, X) \geq d(x, x_i) + (r_i - d(x, x_i)) = r_i \geq d(x, x_i)$. ■

Lemma 2.5 *Let x be a point and let x_i belong to Z. If x belongs to B_i, then $charge(x, Z) \leq r_i$.*

Proof: By Lemma 2.2, there is no point x_j in Z such that $i \neq j$ and x belongs to B_j. The claim now follows from the definition of $charge(x, Z)$, since $d(x, Z) \leq d(x, x_i)$. ■

Lemma 2.6 *Let x be a point and let x_i belong to Z. If x does not belong to B_i, then $charge(x, Z) \leq d(x, x_i)$.*

Proof: The claim is immediate unless there is a point x_j in Z such that x belongs to B_j. If such a point x_j exists, then Lemmas 2.2 and 2.5 imply $d(x_i, x_j) > 2 \cdot \max\{r_i, r_j\}$ and $charge(x, Z) \leq r_j$, respectively. The claim now follows since $d(x, x_i) \geq d(x_i, x_j) - d(x, x_j) > 2r_j - r_j = r_j$. ■

Lemma 2.7 *For any point x and configuration X, $charge(x, Z) \leq 3 \cdot charge(x, X)$.*

Proof: Let x_i be some point in X such that $d(x, x_i) = d(x, X)$. By Lemma 2.1, there exists a point x_j in Z such that $j \leq i$ and $d(x_i, x_j) \leq 2r_i$.

If x belongs to B_j, then $charge(x, Z) \leq r_j$ by Lemma 2.5. The claim follows since $j \leq i$ implies $r_j \leq r_i$ and Lemma 2.4 implies $charge(x, X) \geq r_i$.

If x does not belong to B_j, then $charge(x, Z) \leq d(x, x_j)$ by Lemma 2.6. Thus $charge(x, Z) \leq d(x, x_i) + d(x_i, x_j) \leq d(x, x_i) + 2r_i$. The claim now follows by Lemma 2.4, since the ratio of $d(x, x_i) + 2r_i$ to $\max\{r_i, d(x, x_i)\}$ is at most 3. ■

3 Online Median Placement

In the previous section, we found that a simple greedy algorithm yields interesting results for the facility location problem. The most obvious greedy algorithm for the online median problem is to select as the next point in the ordering the one that minimizes the objective function. Unfortunately, this algorithm gives an unbounded competitive (resp., approximation) ratio for the online median (resp., k-median) problem. To see this, consider an instance consisting of $n > 3$ points, one "red" and the rest "blue", such that the following conditions are satisfied: the red point has weight 0; each blue point has weight 1; the distance from the red point to any blue point is 1, and the distance between any pair of distinct blue points is 2. The aforementioned greedy algorithm chooses the red point first in the ordering, since that gives a cost of $n - 1$ while choosing any other point gives a cost of $2n - 4$. But then the ratio for a configuration of size $n - 1$ is unbounded since the greedy cost is 1 and the optimal cost is 0. (This example also shows that no online median algorithm can achieve a competitive ratio below $2 - \frac{2}{n-1}$.)

We show that a more careful choice of the point, which we call hierarchically greedy, works well. Let Δ (resp., δ) denote the largest (resp., smallest) distance between two distinct points in the metric space. We define a certain ball about each point, and select a ball A of maximum value. But rather than simply choosing the center of ball A as the next point in the ordering, we apply the approach recursively to select a point within a region defined by A. At each successive level of recursion, we consider geometrically smaller balls about the remaining candidate points. Within $O(\log \frac{\Delta}{\delta})$ levels of recursion, we arrive at a ball containing a single point, and we return this point as the next one in the ordering. Note that whereas the greedy algorithm discussed in the previous paragraph makes a single greedy choice to select a point, the hierarchically greedy algorithm makes $O(\log \frac{\Delta}{\delta})$ greedy choices per point.

Throughout this section, let λ, α, β, and γ denote real numbers satisfying the following inequalities.

$$\lambda \geq 1 \tag{1}$$
$$\alpha > 1+\lambda \tag{2}$$
$$\beta \geq \frac{\lambda(\alpha-1)}{\alpha-1-\lambda} \tag{3}$$
$$\gamma \geq \left(\frac{\alpha^2\beta+\alpha\beta}{\alpha-1}+\alpha\right)\lambda \tag{4}$$

The online median algorithm of Section 3.1 below makes use of the following additional definitions. A ***child*** of a ball (x, r) is any ball $(y, \frac{r}{\alpha})$ where $d(x, y) \leq \beta r$. For any point x, let $isolated(x, \emptyset)$ denote the ball $(x, \max_{y \in U} d(x, y))$. For any point x and configuration X, let $isolated(x, X)$ denote the ball $(x, d(x, X)/\gamma)$. For any nonempty sequence ϱ, we let $head(\varrho)$ (resp., $tail(\varrho)$) denote the first (resp., last) element of ϱ.

3.1 Algorithm

Let $Z_0 = \emptyset$. For $i = 0$ to $n-1$, execute the following steps:

- Let σ_i denote the singleton sequence $\langle A \rangle$ where A is a maximum value ball in $\{isolated(x, Z_i) \mid x \in U \setminus Z_i\}$.
- While the ball $tail(\sigma_i)$ has more than one child, append a maximum value child of $tail(\sigma_i)$ to σ_i.
- Let $Z_{i+1} = Z_i \cup \{center(tail(\sigma_i))\}$.

The output of the online median algorithm is a collection of point sets Z_i such that $|Z_i| = i$, $0 \leq i \leq n$, and $Z_i \subseteq Z_{i+1}$, $0 \leq i < n$. Note that it is sufficient for an implementation of the algorithm to maintain the ball $tail(\sigma_i)$, as opposed to the entire sequence σ_i. The sequence σ_i has been introduced in order to facilitate the analysis.

We discuss two implementations of the online median algorithm in Section 3.4. The first implementation has a slightly superlinear running time. The second implementation runs in linear time, but assumes a (linear) preprocessing phase in which all distances are rounded down to the nearest integral power of λ. (Note that for the preprocessing phase to be well-defined, we require $\lambda > 1$.) If the input distance function is a metric, it is straightforward to see that such rounding produces a λ-approximate metric.

3.2 Competitive Ratio

Before proceeding with the analysis, we introduce a number of additional definitions.

- Let z_i denote the unique point in $Z_{i+1} \setminus Z_i$, $0 \leq i < n$.
- For any configuration X and set of points Y, let $cost(X, Y) = \sum_{y \in Y} d(y, X) \cdot w(y)$.
- For any configuration X, we partition U into $|X|$ sets $\{cell(x, X) \mid x \in X\}$ as follows: For each point y in U, we choose a point x in X such that $d(y, X) = d(y, x)$ and add y to $cell(x, X)$.
- For any configuration X, point x in X, and set of points Y, we define $in(x, X, Y)$ as $cell(x, X) \cap isolated(x, Y)$ and $out(x, X, Y)$ as $cell(x, X) \setminus in(x, X, Y)$.

- For any configuration X and set of points Y, we define $in(X,Y)$ as $\cup_{x\in X} in(x,X,Y)$ and $out(X,Y)$ as $U \setminus in(X,Y)$.

In this section we present our main result, Theorem 2 below. In order to minimize the competitive ratio of $2\lambda(\gamma+1)$ implied by the theorem, we set λ to 1, set α to $2+\sqrt{3}$ and set β and γ to the right-hand sides of Equations (3) and (4), respectively. We thereby establish a competitive ratio of slightly below 30 for the online median problem. In Section 3.4 we describe a linear-time implementation of the online median algorithm for which the parameter λ is required to be strictly greater than 1. The degradation in the competitive ratio that results by setting λ greater than 1 can be made arbitrarily small by choosing λ sufficiently close to 1.

Theorem 2 *For any configuration X, $cost(Z_{|X|}) \leq 2\lambda(\gamma+1)\cdot cost(X)$.*

Proof: Let $Y = in(X, Z_{|X|})$ and let $Y' = out(X, Z_{|X|}) = U \setminus Y$. Note that $cost(X) = cost(X,Y) + cost(X,Y')$ and $cost(Z_{|X|}) = cost(Z_{|X|}, Y) + cost(Z_{|X|}, Y')$. Thus the theorem follows immediately from Lemmas 3.2, 3.4, and 3.5 below. ∎

Lemma 3.1 *For any configuration X, point x in X, and point y in $out(x, X, Z_{|X|})$, $d(y, Z_{|X|}) \leq \lambda(\gamma+1)\cdot d(y,X)$.*

Proof: Let $isolated(x, Z_{|X|}) = (x, r)$. Note that $d(x,y) > r$. Also, by the definition of $isolated(x, Z_{|X|})$, there is a point z in $Z_{|X|}$ such that $d(x,z) = \gamma r$. Hence $d(y,z) \leq \lambda[d(x,y) + d(x,z)] = \lambda[d(x,y) + \gamma r] < \lambda[d(x,y) + \gamma \cdot d(x,y)] = \lambda(\gamma+1)\cdot d(x,y) = \lambda(\gamma+1)\cdot d(y,X)$. The claim follows since $d(y,z) \geq d(y, Z_{|X|})$. ∎

Lemma 3.2 *For any configuration X,*

$$cost(Z_{|X|}, out(X, Z_{|X|})) \quad \leq \quad \lambda(\gamma+1)\cdot cost(X, out(X, Z_{|X|})).$$

Proof: Summing the inequality of Lemma 3.1 over all y in $out(x, X, Z_{|X|})$, we obtain

$$cost(Z_{|X|}, out(x, X, Z_{|X|})) \quad \leq \quad \lambda(\gamma+1)\cdot cost(X, out(x, X, Z_{|X|})).$$

The claim now follows by summing the above inequality over all x in X. ∎

Lemma 3.3 *For any configuration X and point x in X,*

$$cost(Z_{|X|}, in(x, X, Z_{|X|})) \quad \leq \quad \lambda(\gamma+1)[cost(X, in(x, X, Z_{|X|})) + value(isolated(x, Z_{|X|}))].$$

Proof: Assume that $isolated(x, Z_{|X|}) = (x, r)$. Note that $d(x,y) = \gamma r$ for some y in $Z_{|X|}$. Thus, for any z in $isolated(x, Z_{|X|})$, $d(y,z) \leq \lambda[d(y,x) + d(x,z)] \leq \lambda(\gamma+1)r$. It follows that $cost(Z_{|X|}, in(x, X, Z_{|X|}))$ is at most $\lambda(\gamma+1)$ times

$$\begin{aligned}
\sum_{z\in in(x,X,Z_{|X|})} r\cdot w(z) \quad &\leq \quad \sum_{z\in in(x,X,Z_{|X|})} d(x,z)\cdot w(z) + \sum_{z\in isolated(x,Z_{|X|})} (r - d(x,z))\cdot w(z) \\
&= \quad cost(X, in(x, X, Z_{|X|})) + value(isolated(x, Z_{|X|})).
\end{aligned}$$

∎

Lemma 3.4 *For any configuration X and point x in X,*

$$cost(Z_{|X|}, in(X, Z_{|X|})) \quad \leq \quad \lambda(\gamma+1)[cost(X, in(X, Z_{|X|})) + \sum_{x \in X} value(isolated(x, Z_{|X|}))].$$

Proof: The claim follows by summing the inequality of Lemma 3.3 over all x in X. ■

Our main technical lemma is stated below. The proof is given in the next subsection.

Lemma 3.5 *For any configuration X, $\sum_{x \in X} value(isolated(x, Z_{|X|})) \leq cost(X)$.*

3.3 Proof of Lemma 3.5

In this section we establish our main technical lemma, Lemma 3.5.

Lemma 3.6 *Let $A = (x, r)$ belong to σ_i. Then $d(x, Z_i) \geq \gamma r$.*

Proof: Let z be a point in Z_i such that $d(x, z) = d(x, Z_i)$. If $A = head(\sigma_i)$ then $A = isolated(x, Z_i)$ and the result is immediate. Otherwise, let $B = (y, s)$ denote the predecessor of A in σ_i and assume inductively that $d(y, Z_i) \geq \gamma s$. Note that $d(x, y) \leq \beta s$ and $s = \alpha r$. Thus $d(x, Z_i) = d(x, z) \geq d(y, z)/\lambda - d(x, y) \geq (\gamma/\lambda - \beta)\alpha r \geq \gamma r$, where the last step follows from Equation (4). ■

Lemma 3.7 *Let $A = (x, r)$ belong to σ_i and let $B = (y, s)$ belong to σ_j. If $i < j$ and $d(x, y) \leq r + s$, then the following claims hold: (i) $radius(head(\sigma_j)) \leq \frac{r}{\alpha}$; (ii) $A \neq tail(\sigma_i)$; (iii) the successor of A in σ_i, call it C, satisfies $value(C) \geq value(head(\sigma_j))$.*

Proof: Let $head(\sigma_j) = (y', s')$. For part (i), we know that $d(y', z_i) \geq \gamma s'$ by Lemma 3.6. Also, we have

$$\begin{aligned} d(y', z_i) &\leq \lambda \left[d(y', y) + d(y, x) + d(x, z_i)\right] \\ &\leq \lambda \left[\beta\left(s' + \frac{s'}{\alpha} + \cdots + \alpha s\right) + s + r + \beta\left(r + \frac{r}{\alpha} + \cdots\right)\right] \\ &\leq \left[\frac{\alpha\beta}{\alpha - 1} \cdot (r + s') + r\right]\lambda. \end{aligned}$$

Combining the two inequalities and applying Equation (4), we obtain

$$\left(\frac{\alpha^2\beta + \alpha\beta}{\alpha - 1} + \alpha\right)\lambda s' \quad \leq \quad \left[\frac{\alpha\beta}{\alpha - 1} \cdot (r + s') + r\right]\lambda.$$

Multiplying through by $(\alpha - 1)/\lambda$ and rearranging, we get $r \geq \frac{\alpha^2\beta + \alpha^2 - \alpha}{\alpha\beta + \alpha - 1} \cdot s' = \alpha s'$, establishing the claim.

For part (ii), note that $d(x, y) \leq r + \frac{r}{\alpha} < \beta r$ by part (i) and Equation (3). Thus A has at least two children; the claim follows.

For part (iii), we use Equations (2) and (3) and part (i) to observe that

$$\begin{aligned} d(x, y') &\leq \lambda \left[d(x, y) + d(y, y')\right] \\ &\leq \lambda \left[r + s + \left(\alpha s + \alpha^2 s + \cdots + s'\right)\beta\right] \\ &\leq \lambda r + \frac{\alpha\beta\lambda}{\alpha - 1} \cdot s' \\ &\leq \lambda r + \frac{\alpha\beta\lambda}{\alpha - 1} \cdot \frac{r}{\alpha} \\ &\leq \left(\frac{\beta}{\alpha - 1} + 1\right)\lambda r, \end{aligned}$$

which is at most βr by Equation (3). It follows that $head(\sigma_j)$ is contained in a child of A. Thus $value(C) \geq value(head(\sigma_j))$. ■

For ease of notation, throughout the remainder of this section we fix a configuration X, and let k denote $|X|$. We now describe a ***pruning procedure*** that takes as input the k sequences σ_i, $0 \leq i < k$, and produces as output k sequences τ_i, $0 \leq i < k$. The sequence τ_i is initialized to σ_i, $0 \leq i < k$. The (nondeterministic) pruning procedure then performs a number of iterations. In a general iteration, the pruning procedure checks whether there exist two balls $A = (x, r)$ and $B = (y, s)$ in distinct sequences τ_i and τ_j, respectively, such that $i < j$ and $d(x, y) \leq r + s$. If not, the pruning procedure terminates. If so, the sequence τ_i is redefined as the proper suffix of (the current) τ_i beginning at the successor of A. Note that part (ii) of Lemma 3.7 ensures that the pruning procedure is well-defined. Furthermore, the procedure is guaranteed to terminate since each iteration reduces the length of some sequence τ_i.

Lemma 3.8 *Let $A = (x, r)$ belong to τ_i and let $B = (y, s)$ belong to τ_j. If $i < j$ then $d(x, y) > r + s$.*

Proof: Immediate from the definition of the pruning procedure. ■

Lemma 3.9 *Each sequence τ_i is nonempty.*

Proof: Immediate from part (ii) of Lemma 3.7 and the definition of the pruning procedure. ■

Lemma 3.10 *Let x be a point and assume that $0 \leq i < j \leq n$. Then*

$$value(isolated(x, Z_i)) \geq value(isolated(x, Z_j)).$$

Proof: Since $Z_i \subseteq Z_j$, $radius(isolated(x, Z_i)) \geq radius(isolated(x, Z_j))$. The claim follows. ■

Lemma 3.11 *Let x be a point and assume that $0 \leq i < k$. Then*

$$value(head(\sigma_i)) \geq value(isolated(x, Z_k)).$$

Proof: If x belongs to Z_i, then $radius(isolated(x, Z_i)) = 0$, so $value(isolated(x, Z_i)) = 0$ and there is nothing to prove. Otherwise, $value(head(\sigma_i)) \geq value(isolated(x, Z_i))$ by the definition of the online median algorithm, and the claim follows by Lemma 3.10. ■

Lemma 3.12 *Let x be a point and assume that $0 \leq i < k$. Then*

$$value(head(\tau_i)) \geq value(isolated(x, Z_k)).$$

Proof: We prove that the claim holds before and after each iteration of the pruning procedure. Initially, $\tau_i = \sigma_i$ and the claim holds by Lemma 3.11. If the claim holds before an iteration of the pruning procedure, then it holds after the iteration by part (iii) of Lemma 3.7. ■

A ball $A = (x, r)$ is defined to be ***covered*** iff $d(x, X) < r$. A ball is ***uncovered*** iff it is not covered.

Lemma 3.13 *For any uncovered ball $A = (x, r)$, $cost(X, A) \geq value(A)$.*

Proof: Note that $cost(X, A) \geq \sum_{y \in A} d(y, X) \cdot w(y) \geq \sum_{y \in A}(r - d(y, x)) \cdot w(y) = value(A)$. ■

Let I denote the set of all indices i in $[k]$ such that some ball in τ_i is covered. We now construct a matching between the sets $[k]$ and X as follows. First, for each i in I, we match i with a point x in X that belongs to the last covered ball in the sequence τ_i. (Note that such a point x is guaranteed to exist by the definition of I. Furthermore, Lemma 3.8 ensures that we do not match the same point with more than one index.) Second, for each i in $[k] \setminus I$ in turn, we match i with an arbitrary unmatched point x in X.

We now construct a function φ mapping each point x in X to an uncovered ball. For each x in X that is matched with an index i in $[k] \setminus I$, we set $\varphi(x)$ to $head(\tau_i)$. For each x in X that is matched with an index i in I, we set $\varphi(x)$ to the successor of the last covered ball in τ_i unless $tail(\tau_i)$ is covered, in which case we set $\varphi(x)$ to the ball $(x, 0)$.

Lemma 3.14 *For any pair of distinct points x and y in X, $\varphi(x) \cap \varphi(y) = \emptyset$.*

Proof: Immediate from Lemma 3.8 and the fact that the ball $(x, 0)$ is contained in $tail(\tau_i)$. ■

Lemma 3.15 *For any point x in X, $value(\varphi(x)) \geq value(isolated(x, Z_k))$.*

Proof: If x is matched with an index i in $[k] \setminus I$, the claim follows by Lemma 3.12. If x is matched with an index i in I, we consider two cases. If $tail(\tau_i)$ is covered, then $x = z_i$ since $tail(\tau_i)$ has exactly one child. The claim follows since $\varphi(x) = isolated(x, Z_k) = (x, 0)$. If $tail(\tau_i)$ is uncovered, then the predecessor of $\varphi(x)$ in τ_i, call it $A = (y, r)$, exists and contains x. It follows that $value(\varphi(x)) \geq value(B)$, where $B = (x, r/\alpha)$ is the child of A centered at x. Let $C = (x, s)$ denote the ball $isolated(x, Z_k)$. Below we complete the proof of the claim by showing that $r/\alpha \geq s$, which implies that $B \supseteq C$ and hence $value(B) \geq value(C)$.

It remains to prove that $r/\alpha \geq s$ in the final case considered above. We have

$$\begin{aligned} d(x, z_i) &\leq \lambda\,[d(x, y) + d(y, z_i)] \\ &\leq \lambda r + \beta\lambda\left(r + \frac{r}{\alpha} + \cdots\right) \\ &\leq \left(1 + \frac{\alpha\beta}{\alpha - 1}\right)\lambda r, \end{aligned}$$

which is less than $\gamma r/\alpha$ by Equation (4). The desired inequality follows since $d(x, z_i) \geq \gamma s$ by the definition of C. ■

Lemmas 3.13, 3.14, and 3.15 together yield a proof of Lemma 3.5.

3.4 Time Complexity

In this section we describe two implementations of the online median algorithm given in Section 3.1. Throughout this section, let ℓ denote the quantity $\log \frac{\Delta}{\delta}$. The first implementation runs in $O((n+\ell) \cdot n \log n)$ time. The second implementation runs in $O(n^2 + \ell n)$ time and assumes an $O(n^2)$-time preprocessing phase in which all distances are rounded down to the nearest integral power of λ. To analyze the running time of the implementations given below, we make use of the following lemma.

Lemma 3.16 *Let $A = (x, r)$ be a child of a ball B in sequence σ_i and let $A' = (x, r')$ be a child of a ball B' in sequence σ_j. If $i < j$ then $r > (\alpha + 1)r'$.*

Proof: First, note that $d(x, z_i) \leq \beta\,(r + r/\alpha + \cdots) \leq \alpha\beta r/(\alpha - 1)$. By Lemma 3.6, $\gamma r' \leq d(x, Z_j) \leq d(x, z_i)$. Combining these inequalities and using Equation (4), we obtain

$$\begin{aligned} r &\geq \frac{(\alpha-1)\gamma}{\alpha\beta} \cdot r' \\ &> \frac{\alpha-1}{\alpha\beta} \cdot \frac{\alpha^2\beta + \alpha\beta}{\alpha - 1} \cdot r' \\ &= (\alpha+1)r'. \end{aligned}$$

∎

In the first implementation, for each point x in U, we sort the remaining points by their distance from x. The total sorting time is $O(n^2 \log n)$. Using these sorted arrays, we can compute the value of any given ball in $O(\log n)$ time. We also maintain the distance from x to the nearest point in Z_i. Note that $d(x, Z_{i+1})$ can be determined in constant time given $d(x, Z_i)$ and z_i. The total time to maintain such distances is thus $O(n^2)$. It follows that the first step of each iteration can be implemented in $O(n)$ time. The total time for the second step is $O(\log n)$ times the sum over all balls A appearing in some sequence σ_i, $0 \leq i < n$, of the number of children of A. By Lemma 3.16, it is straightforward to see that the latter sum is $O(\ell n)$, and thus the total time for the second step is $O(\ell n \log n)$. The running time of the third step is negligible. Thus the running time of the first implementation is $O((n + \ell) \cdot n \log n)$, as claimed above.

For the second implementation, note that after the preprocessing phase, there are $O(\ell)$ distinct distances. Thus, for each point x, $O(n+\ell)$ time is sufficient to construct an $O(\ell)$-sized table that can be used to compute the value of any ball (x, r) in $O(1)$ time. It follows that the total time for the second step can be improved to $O(\ell n)$. The running time of the second implementation is therefore $O(n^2 + \ell n)$, which is linear in the size of the input (in bits).

4 Concluding Remarks

We plan to investigate whether the ideas presented in this paper can be applied to other problems. Korupolu *et al.* [13] give an algorithm and an efficient distributed implementation for hierarchical cooperative caching in which the distance function is an ultrametric. We would like to see if the hierarchical greedy strategy can be used or extended to solve the cooperative caching problem in an arbitrary metric space. It would also be interesting to see if the hierarchical greedy strategy admits an efficient distributed implementation for this problem.

This paper has focused on the development of fast deterministic algorithms for the facility location problem and the online median problem. It would be interesting to investigate whether randomization yields sublinear-time constant factor approximation algorithms for problems of this kind. Indyk gives such approximation algorithms for a collection of metric space problems [9]. For the uniform-demand k-median problem, he gives a bicriteria approximation algorithm that uses random sampling and a black-box k-median algorithm. His algorithm has a constant probability of success and runs in $\tilde{O}(nk^3)$[1] time. Assuming the existence of an $\tilde{O}(n^2)$-time bicriteria k-median algorithm, this time bound can be reduced to $\tilde{O}(nk)$ [8]. Recently, we have obtained an $O(n(k + \log n))$-time approximation algorithm for the uniform-demand k-median problem that uses the online median algorithm in this paper as a black box for solving the k-median problem. (Remark: It is not hard to show an $\Omega(nk)$ lower bound for the k-median problem, even in a randomized setting.) The cost of the configuration of size k returned by our algorithm is within a constant factor of optimal with high probability (i.e., arbitrary inverse polynomial failure probability). We would like

[1]The tilde notation omits polylogarithmic factors in n and k.

to see if our $O(n(k + \log n))$-time randomized algorithm for the uniform-demand k-median problem can be modified to handle arbitrary demands while preserving the time bound.

References

[1] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, UK, 1998.

[2] M. Charikar and S. Guha. Improved combinatorial algorithms for facility location and k-median problems. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 378–388, October 1999.

[3] M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 1–10, May 1999.

[4] F. A. Chudak. Improved approximation algorithms for uncapacitated facility location. In R. E. Bixby, E. A. Boyd, and R. Z. Ríos-Mercado, editors, *Integer Programming and Combinatorial Optimization*, volume 1412 of *Lecture Notes in Computer Science*, pages 180–194, Berlin, 1998. Springer.

[5] W. Domschke and A. Drexl. *Location and Layout Planning: An International Bibliography*, volume 238 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, Berlin, 1985.

[6] M. X. Goemans and D. P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, Boston, MA, 1995.

[7] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 649–657, January 1998.

[8] P. Indyk. Sublinear time algorithms for metric space problems. Revised version of conference paper (available at `http://theory.stanford.edu/~indyk`).

[9] P. Indyk. Sublinear time algorithms for metric space problems. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 428–434, May 1999.

[10] S. Irani and Karlin A. R. Online computation. In D. S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, Boston, MA, 1995.

[11] K. Jain and V. V. Vazirani. Primal-dual approximation algorithms for metric facility location and k-median problems. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 2–13, October 1999.

[12] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. In *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–10, January 1998.

[13] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman. Placement algorithms for hierarchical cooperative caching. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 586–595, January 1999.

[14] A. A. Kuehn and M. J. Hamburger. A heuristic program for locating warehouses. *Management Science*, 9:643–666, 1963.

[15] J.-H. Lin and J. S. Vitter. Approximation algorithms for geometric median problems. *Information Processing Letters*, 44:245–249, 1992.

[16] J.-H. Lin and J. S. Vitter. ε-approximations with minimum packing constraint violation. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing*, pages 771–782, May 1992.

[17] P. Mirchandani and R. Francis, editors. *Discrete Location Theory*. Wiley, New York, NY, 1990.

[18] D. B. Shmoys, É. Tardos, and K. Aardal. Approximation algorithms for facility location problems. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 265–274, May 1997.

[19] N. E. Young. K-medians, facility location, and the Chernoff-Wald bound. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, January 2000.