Incremental Algorithms for Facility Location and k-Median[⋆]

Dimitris Fotakis

Department of Mathematical, Physical and Computational Sciences,
Aristotle University of Thessaloniki, 54006 Thessaloniki, Greece.
Computer Technology Institute, Patras, Greece.
Email: `fotakis@auth.gr`

Abstract. In the incremental versions of Facility Location and k-Median, the demand points arrive one at a time and the algorithm must maintain a good solution by either adding each new demand to an existing cluster or placing it in a new singleton cluster. The algorithm can also merge some of the existing clusters at any point in time. We present the first incremental algorithm for Facility Location with uniform facility costs which achieves a constant performance ratio and the first incremental algorithm for k-Median which achieves a constant performance ratio using $O(k)$ medians.

1 Introduction

The model of incremental algorithms for data clustering is motivated by practical applications where the demand sequence is not known in advance and the algorithm must maintain a good clustering using a restricted set of operations which result in a solution of hierarchical structure. Charikar et al. [3] introduced the framework of incremental clustering to meet the requirements of data classification applications in information retrieval.

In this paper, we consider the incremental versions of metric Facility Location and k-Median. In *Incremental k-Median* [5], the demand points arrive one at a time. Each demand must be either added to an existing cluster or placed in a new singleton cluster upon arrival. At any point in time, the algorithm can also merge some of the existing clusters. Each cluster is represented by its median whose location is determined at the cluster's creation time. When some clusters are merged with each other, the median of the new cluster must be selected among the medians of its components. The goal is to maintain a solution consisting of at most k clusters/medians which minimize the total assignment cost of the demands considered so far. The assignment cost of a demand is its distance from the median of the cluster the demand is currently included in.

[⋆] This work was partially supported by the FET programme of the EU under contract numbers IST-1999-14186 (ALCOM-FT) and IST-2001-33116 (FLAGS). Part of this work was done while the author was at the Max-Planck-Institut für Informatik, Saarbrücken, Germany.

In *Incremental Facility Location*, demand points arrive one at a time and must be assigned to either an existing or a new facility upon arrival. At any point in time, the algorithm can also merge a facility with another one by closing the first facility and re-assigning all the demands currently assigned to it to the second facility. The objective is to maintain a solution which minimizes the sum of facility and assignment costs. The facility cost only accounts for the facilities currently open, while the assignment cost of a demand is its distance from the facility the demand is currently assigned to.

Both problems are motivated by applications of clustering in settings that the demands arrive online (e.g., clustering the web, data mining) and the algorithm must maintain a good solution without resorting to complete re-clustering. To avoid cluster proliferation, the algorithm is allowed to merge existing clusters. In Incremental k-Median, the number of clusters is fixed in advance. On the other hand, Incremental Facility Location provides a model for applications where the number of clusters cannot be determined in advance due to limited a priori information about the demand sequence. Hence, we introduce a uniform facility cost enforcing the requirement for a relatively small number of clusters.

We evaluate the performance of incremental algorithms using the *performance ratio* [3], whose definition is essentially identical to that of the *competitive ratio* (e.g., [2]). An incremental algorithm achieves a performance ratio of c if for all demand sequences, the algorithm's cost is at most c times the cost of an optimal offline algorithm, which has full knowledge of the demand sequence, on the same instance. We also let n denote the total number of demands.

Comparison to Online and Streaming Algorithms. Similarly to online algorithms, incremental algorithms commit themselves to irrevocable decisions made without any knowledge about future demands. More specifically, when a new demand arrives, the algorithm may decide to add the demand to an existing cluster or merge some clusters with each other. These decisions are irrevocable because once formed, clusters cannot be broken up. However, we are not aware of any simple and natural notion of irrevocable cost that could be associated with the decision that some demands are clustered together.

Incremental k-Median can be regarded as the natural online version of k-Median. Nevertheless, Incremental Facility Location is quite different from the problem of Online Facility Location (OFL) introduced in [10]. In particular, OFL is motivated by network design applications where re-configurations are expensive and often infeasible. Thus, the decisions of opening a facility at a particular location and assigning a demand to a facility are irrevocable as well as the corresponding costs. On the other hand, Incremental Facility Location is motivated by clustering applications where merging existing clusters is not only feasible but also desirable and the important restriction is that existing clusters should not be broken up. Hence, only the decision that some demands are clustered together is irrevocable. As a result, the competitive ratio is $\Theta(\frac{\log n}{\log\log n})$ for OFL [6] and constant for Incremental Facility Location.

Incremental algorithms also bear a resemblance to one-pass streaming algorithms for clustering problems (e.g., [9] for a formal definition of the streaming model of computation). In case of streaming algorithms, the emphasis is on space and time efficient algorithms which achieve a small approximation ratio by ideally performing a single scan over the input. A streaming algorithm for k-Median is not restricted in terms of the solution's structure or the set of operations available. On the other hand, incremental algorithms must maintain a good hierarchical clustering by making irrevocable decisions, but they should only run in polynomial time. Nevertheless, all known incremental algorithms for clustering problems can be either directly regarded as or easily transformed to efficient one-pass streaming algorithms (e.g., [3, 8, 5, 4] and this paper).

Previous Work. Charikar et al. [3] presented incremental algorithms for k-Center which achieve a constant performance ratio using k clusters. Charikar and Panigrahy [5] presented an incremental algorithm for Sum k-Radius which achieves a constant performance ratio using $\mathrm{O}(k)$ clusters. They also proved that any deterministic incremental algorithm for k-Median which maintains at most k clusters must have a performance ratio of $\Omega(k)$. Determining whether there exists an incremental algorithm which achieves a constant performance ratio using $\mathrm{O}(k)$ medians is suggested as an open problem in [5].

In [8, 4], they are presented one-pass streaming algorithms for k-Median which assume that n is known in advance. For k much smaller than n^ϵ, the algorithms of Guha et al. [8] achieve a performance ratio of $2^{\mathrm{O}(1/\epsilon)}$ using n^ϵ medians and run in $\mathrm{O}(nk\,\mathrm{poly}(\log n))$ time and n^ϵ space. The algorithm of Charikar et al. [4] achieves a constant performance ratio with high probability (whp.) using $\mathrm{O}(k \log^2 n)$ medians and runs in $\mathrm{O}(nk \log^2 n)$ time and $\mathrm{O}(k \log^2 n)$ space.

The algorithms of [10, 6, 1] for OFL are the only known incremental algorithms for Facility Location. In OFL [10], the demands arrive one at a time and must be irrevocably assigned to either an existing or a new facility upon arrival. In [10], a randomized $\mathrm{O}(\frac{\log n}{\log\log n})$-competitive algorithm and a lower bound of $\omega(1)$ are presented. In [6], the lower bound is improved to $\Omega(\frac{\log n}{\log\log n})$ and a deterministic $\mathrm{O}(\frac{\log n}{\log\log n})$-competitive algorithm is given. In [1], it is presented a simpler and faster deterministic $\mathrm{O}(2^d \log n)$-competitive algorithm for d-dimensional Euclidean spaces.

The lower bounds of [10, 6] hold only if the decision of opening a facility at a particular location is irrevocable. Hence, they do not apply to Incremental Facility Location. However, the lower bound of [6] implies that every algorithm which maintains $\mathrm{o}(k \log n)$ facilities must incur a total *initial* assignment cost of $\omega(1)$ times the optimal cost, where the initial assignment cost of a demand is its distance from the first facility the demand is assigned to. Therefore, every algorithm treating merge as a black-box operation cannot approximate the optimal assignment cost within a constant factor unless it uses $\Omega(k \log n)$ facilities (e.g., the algorithm of [4]). In other words, to establish a constant performance ratio, one needs a merge rule which can provably *decrease* both the algorithm's facility and assignment costs.

Contribution. We present the first incremental algorithm for metric Facility Location with uniform facility costs which achieves a constant performance ratio. The algorithm combines a simple rule for opening new facilities with a novel merge rule based on distance instead of cost considerations. We use a new technique to prove that a case similar to the special case that the optimal solution consists of a single facility is the dominating case in the analysis. This technique is also implicit in [6] and may find applications to other online problems. To overcome the limitation imposed by the lower bound of [6], we establish that in the dominating case, merge operations also decrease the assignment cost.

Using the algorithm for Facility Location as a building block, we obtain the first incremental algorithm for k-Median which achieves a constant performance ratio using $\mathrm{O}(k)$ medians, thus resolving the open question of [5]. In other words, we improve the number of medians required for a constant performance ratio from $\mathrm{O}(k \log^2 n)$ to $\mathrm{O}(k)$. Combining our techniques with the techniques of [4], we obtain a randomized incremental algorithm which achieves a constant performance ratio whp. using $\mathrm{O}(k)$ medians and runs in $\mathrm{O}(nk^2 \log^2 n)$ time and $\mathrm{O}(k^2 \log^2 n)$ space. This algorithm can also be regarded as a time and space efficient one-pass streaming algorithm for k-Median.

The proofs and the technical details omitted from this extended abstract due to space constraints can be found in the full version of this paper [7].

Notation. We only consider unit demands and allow multiple demands to be located at the same point. For Incremental Facility Location, we restrict our attention to the special case of uniform facility costs, where the cost of opening a facility, denoted by f, is the same for all points. We also use the terms facility, median and cluster interchangeably.

A metric space $\mathcal{M} = (M, d)$ is usually identified by its point set M. The distance function d is non-negative, symmetric, and satisfies the triangle inequality. For points $u, v \in M$ and a subspace $M' \subseteq M$, $d(u, v)$ denotes the distance between u and v, $D(M')$ denotes the diameter of M', and $d(M', u)$ denotes the distance between u and the nearest point in M'. It is $d(\emptyset, u) = \infty$. For a point $u \in M$ and a non-negative number r, $\mathrm{Ball}(u, r) \equiv \{v \in M : d(u, v) \leq r\}$.

2 An Incremental Algorithm for Facility Location

The algorithm Incremental Facility Location - IFL (Fig. 1) maintains its *facility configuration* F, its *merge configuration* consisting of a *merge ball* $\mathrm{Ball}(w, m(w))$ for each facility $w \in F$, and the set L of *unsatisfied demands*.

A demand becomes unsatisfied, i.e., is added to the set L, upon arrival. Each unsatisfied demand holds a *potential* always equal to its distance to the nearest existing facility. If the unsatisfied neighborhood B_u of a new demand u has accumulated a potential of βf, a new facility located at the same point with u opens. Then, the demands in B_u become satisfied and lose their potential. Hence, the notion of unsatisfied demands ensures that each demand contributes to the facility cost at most once.

Each facility $w \in F$ maintains the set $\mathrm{Init}(w)$ of the demands *initially assigned* to w (namely, the demands assigned to w upon arrival and not after a merge operation), its *merge radius* $m(w)$ and the corresponding *merge ball*

```
Let $x$, $\beta$, and $\psi$ be constants.
$F \leftarrow \emptyset$;  $L \leftarrow \emptyset$;
For each new demand $u$:
    $L \leftarrow L \cup \{u\}$;  $r_u \leftarrow d(F,u)/x$;
    $B_u \leftarrow \mathrm{Ball}(u, r_u) \cap L$;
    $\mathrm{Pot}(B_u) = \sum_{v \in B_u} d(F,v)$;
    if $\mathrm{Pot}(B_u) \geq \beta f$ then
        Let $w'$ be the location of $u$;
        open($w'$);  $L \leftarrow L \setminus B_u$;
        for each $w \in F \setminus \{w'\}$ do
            if $d(w,w') \leq m(w)$ then
                merge($w \rightarrow w'$);
    Let $w$ be the facility in $F$ closest to $u$;
    update_merge_radius($m(w)$);
    initial_assignment($u$, $w$);
```

```
open($w'$)
    $F \leftarrow F \cup \{w'\}$;  $\mathrm{Init}(w') \leftarrow \emptyset$;
    $C(w') \leftarrow \emptyset$;  $m_1(w') \leftarrow 3\,r_u$;
merge($w \rightarrow w'$)
    $F \leftarrow F \setminus \{w\}$;
    $C(w') \leftarrow C(w') \cup C(w)$;
update_merge_radius($m(w)$)
    $m_2(w) = \max\{r :$
$|\mathrm{Ball}(w, \frac{r}{\psi}) \cap (\mathrm{Init}(w) \cup \{u\})| \cdot r \leq \beta f\}$;
    $m(w) = \min\{m_1(w), m_2(w)\}$;
initial_assignment($u$, $w$)
    $\mathrm{Init}(w) \leftarrow \mathrm{Init}(w) \cup \{u\}$;
    $C(w) \leftarrow C(w) \cup \{u\}$;
```

Fig. 1. The algorithm Incremental Facility Location – IFL .

$\mathrm{Ball}(w, m(w))$. w is the only facility in its merge ball. In particular, when w opens, the merge radius of w is initialized to a fraction of the distance between w and the nearest existing facility. Then, if a new facility w' opens in w's merge ball, w is merged with w'. The algorithm keeps decreasing $m(w)$ to ensure that no merge operation can dramatically increase the assignment cost of the demands in $\mathrm{Init}(w)$. More specifically, it maintains the invariant that

$$|\mathrm{Init}(w) \cap \mathrm{Ball}(w, \tfrac{m(w)}{\psi})| \cdot m(w) \leq \beta f \tag{1}$$

After the algorithm has updated its configuration, it assigns the new demand to the nearest facility. We always distinguish between the arrival and the assignment time of a demand because the algorithm's configuration may have changed in between.

If the demands considered by IFL occupy m different locations, IFL can be implemented in $\mathrm{O}(nm|F_{\max}|)$ time and $\mathrm{O}(\min\{n, m|F_{\max}|\})$ space, where $|F_{\max}|$ is the maximum number of facilities in F at any point in time. The remaining of this section is devoted to the proof of the following theorem.

Theorem 1. *For every $x \geq 18$, $\beta \geq \frac{4(x+1)}{x-8}$, and $\psi \in [4,5]$,* IFL *achieves a constant performance ratio for Facility Location with uniform facility costs.*

Preliminaries. For an arbitrary fixed sequence of demands, we compare the algorithm's cost with the cost of the optimal facility configuration, denoted by F^*. We reserve the term (optimal) *center* for the optimal facilities in F^* and the term *facility* for the algorithm's facilities in F. In the optimal solution, each demand u is assigned to the nearest center in F^*, denoted by c_u. We use the clustering induced by F^* to map the demands and the algorithm's facilities to the optimal centers. In particular, a demand u is always mapped to c_u. Similarly, a facility w is mapped to the nearest optimal center, denoted by c_w. Also, let $d^*_u = d(c_u, u)$ be the optimal assignment cost of u, let $\mathrm{Fac}^* = |F^*|f$ be the optimal facility cost, and let $\mathrm{Asg}^* = \sum_u d^*_u$ be the optimal assignment cost.

To refer to the algorithm's configuration at the moment a demand is considered (resp. facility opens), we use the demand's (resp. facility's) identifier as a subscript. We distinguish between the algorithm's configuration at the demand's arrival and assignment time using plain symbols to refer to the former and primed symbols to refer to the latter time. For example, for a demand u, F_u (resp. F'_u) is the facility configuration at u's arrival (resp. assignment) time. Similarly, for a facility w, F_w (resp. F'_w) is the facility configuration just before (resp. after) w opens.

For each facility w, we let $B_w \equiv \mathrm{Ball}(w, d(F_w, w)/x) \cap L_w$ denote the unsatisfied neighborhood contributing its potential towards opening w (i.e., if u is the demand opening w, then $B_w = B_u$). When an existing facility w is merged with a new facility w', the existing facility w is closed and the demands currently assigned to w are re-assigned to the new facility w' (and not the other way around).

Basic Properties. The following lemmas establish the basic properties of IFL.

Lemma 1. *For every facility w mapped to c_w, $d(c_w, w) \leq d(F_w, c_w)/3$.*

Proof Sketch. Each new facility is opened by a small unsatisfied neighborhood with a large potential. If there were no optimal centers close to such a neighborhood, the cost of the optimal solution could decrease by opening a new center.

For a formal proof, we assume that there is a facility w such that $d(c_w, w) > d(F_w, c_w)/3$. Since $B_w \subseteq \mathrm{Ball}(w, d(F_w, w)/x)$, we can show that for each $u \in B_w$, $d(u, w) < \frac{4}{x-4} d^*_u$ and $d(F_w, u) < \frac{4(x+1)}{x-4} d^*_u$. Using $\mathrm{Pot}(B_w) \geq \beta f$, we conclude that for β appropriately large, $f + \sum_{u \in B_w} d(u, w) < \sum_{u \in B_w} d^*_u$, which contradicts to the optimality of F^*. $\square$

Lemma 2. *For every facility w, there will always exist a facility in* $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$ *and every demand currently assigned to w will remain assigned to a facility in* $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$.

Proof Sketch. The lemma holds because every time a facility w is merged with a new facility w', the merge radius of w' is a fraction of the merge radius of w. Formally, if w is merged with a new facility w', it must be $d(w, w') \leq m(w)$ and $m(w') \leq \frac{3}{x} d(w, w')$. Hence, $\mathrm{Ball}(w', \frac{x}{x-3} m(w')) \subseteq \mathrm{Ball}(w, \frac{x}{x-3} m(w))$. $\square$

Facility Cost. We distinguish between *supported* and *unsupported facilities*: A facility w is supported if $\mathrm{Asg}^*(B_w) = \sum_{u \in B_w} d^*_u \geq \frac{\beta}{3x} f$, and unsupported otherwise. The cost of each supported facility w can be directly charged to the optimal assignment cost of the demands in B_w. Since each demand contributes to the facility cost at most once, the total cost of supported facilities is at most $\frac{3x}{\beta} \mathrm{Asg}^*$. On the other hand, the merge ball of each unsupported facility w mapped to the optimal center c_w includes a sufficiently large area around c_w. Thus, each unsupported facility w mapped to c_w is merged with the first new facility w' also mapped to c_w because w' is included in w's merge ball. Consequently, there can be at most one unsupported facility mapped to each optimal center. Hence, the total cost of unsupported facilities never exceeds Fac^* and the algorithm's facility cost is at most $\mathrm{Fac}^* + \frac{3x}{\beta}\, \mathrm{Asg}^*$.

Lemma 3. *Let w be an unsupported facility mapped to an optimal center c_w, and let w' be a new facility also mapped to c_w. If w' opens while w is still open, then w is merged with w'.*

Proof Sketch. By Lemma 1, it must be $d(c_w, w') \leq d(F_{w'}, c_w)/3 \leq d(c_w, w)/3$, because w' is mapped to c_w and $w \in F_{w'}$ by hypothesis. We can also prove that $m(w) \geq 3\, d(c_w, w)/2$. This implies the lemma because $d(w, w') \leq d(c_w, w) + d(c_w, w') \leq 4\, d(c_w, w)/3 \leq m(w)$ and w must be merged with w'. □

Assignment Cost. We start with a sketch of the analysis in the special case that F^* consists of a single optimal center, denoted by c. The same ideas can be applied to the case that some optimal centers are very close to each other and isolated from the remaining centers in F^* (cf. *isolated active coalitions*).

Let w be the facility which is currently the nearest facility to c. By Lemma 1, w stops being the nearest facility to c as soon as a new facility w' opens. By Lemma 3, if w is an unsupported facility, it is merged with w'. The main idea is that as long as the algorithm keeps merging existing facilities with new ones (which are significantly closer to c), the algorithm's facility configuration converges rapidly to c and the algorithm's assignment cost converges to the optimal assignment cost. In addition, the rule for opening new facilities ensures that the algorithm's assignment cost for the demands arriving after w's and before w''s opening remains within a constant factor from the optimal cost. A simple induction yields that the assignment cost for the demands arriving as long as existing facilities are merged with new ones (cf. *good inner demands*) remains within a constant factor from the optimal cost.

However, if w is a supported facility, it may not be merged with the new facility w'. Then, the algorithm is charged with an irrevocable cost because the chain of merge operations converging to the optimal center has come to an end. This cost accounts for the assignment cost of the demands whose facility has stopped converging to c (cf. *bad inner demands*). This cost is charged to the optimal assignment cost of the demands in B_w, which is at least $\frac{\beta}{3x} f$ since w is a supported facility.

For the analysis of non-isolated sets of optimal centers (cf. *non-isolated active coalitions*), it is crucial that Lemma 2 implies a notion of distance (cf. *configuration distance*) according to which the algorithm's configuration converges monotonically to each point of the metric space. The analysis is based on the fact that a set of optimal centers can be non-isolated only if its distance to the algorithm's configuration lies in a relatively short interval. This implies that the case dominating the analysis and determining the algorithm's performance ratio is that of isolated sets of optimal centers.

We proceed to formally define the basic notions used in the analysis of the algorithm's assignment cost. In the following, $\lambda = 3x + 2$, $\rho = (\psi + 2)(\lambda + 2)$, and $\gamma = 12\rho$ are appropriately chosen constants.

Configuration Distance. For an optimal center $c \in F^*$ and a facility $w \in F$, *the configuration distance between c and w*, denoted by $g(c, w)$, is $g(c, w) = d(c, w) + \frac{x}{x-3}\, m(w)$. For an optimal center $c \in F^*$, the *configuration distance of*

c, denoted by $g(c)$, is $g(c) = \min_{w \in F}\{g(c, w)\} = \min_{w \in F}\{d(c, w) + \frac{x}{x-3}\, m(w)\}$. The configuration distance $g(c)$ is non-increasing with time and there always exists a facility within a distance of $g(c)$ from c (Lemma 2).

Coalitions. A set of optimal centers $K \subseteq F^*$ with representative $c_K \in K$ forms a *coalition* as long as $g(c_K) > \rho D(K)$. A coalition K becomes *broken* as soon as $g(c_K) \leq \rho D(K)$. A coalition K is *isolated* if $g(c_K) \leq \mathrm{sep}(K)/3$ and *non-isolated* otherwise, where $\mathrm{sep}(K) = d(K, F^* \setminus K)$ denotes the distance separating the optimal centers in K from the remaining optimal centers. Intuitively, as long as K's diameter is much smaller than $g(c_K)$ (K is a coalition), the algorithm behaves as if K was a single optimal center. If the algorithm is bound to have a facility which is significantly closer to K than any optimal center not in K (K is isolated), then as far as K is concerned, the algorithm behaves as if there were no optimal centers outside K.

A *hierarchical decomposition* $\mathcal{K}$ of F^* is a complete laminar set system[1] on F^*. Every hierarchical decomposition of F^* contains at most $2\,|F^*| - 1$ sets. Given a hierarchical decomposition $\mathcal{K}$ of F^*, we can fix an arbitrary representative c_K for each $K \in \mathcal{K}$ and regard $\mathcal{K}$ as a system of coalitions which hierarchically cover F^*. Formally, given a hierarchical decomposition $\mathcal{K}$ of F^* and the current algorithm's configuration, a set $K \in \mathcal{K}$ is *an active coalition* if K is still a coalition (i.e., $g(c_K) > \rho D(K)$), while every superset of K in $\mathcal{K}$ has become broken (i.e., for every $K' \in \mathcal{K}, K \subset K'$, $g(c_{K'}) \leq \rho D(K')$). The current algorithm's configuration induces a collection of active coalitions which form a partitioning of F^*. Since $g(c_K)$ is non-increasing, no coalition which has become broken (resp. isolated) can become active (resp. non-isolated) again.

Let $D_N(K) \equiv \max\{D(K), \frac{1}{3\rho}\,\mathrm{sep}(K)\}$. By definition, K becomes either isolated or broken as soon as $g(c_K) \leq \rho D_N(K)$. Using [6, Lemma 1], we can show that there is a hierarchical decomposition of F^* such that no coalition K becomes active before $g(c_K) < (\rho{+}1)\gamma^2 D_N(K)$. In the following, we assume that the set of active coalitions is given by a fixed hierarchical decomposition $\mathcal{K}$ of F^* such that for every non-isolated active coalition K, $\rho D_N(K) < g(c_K) < (\rho + 1)\gamma^2 D_N(K)$.

Each new demand u is mapped to the unique active coalition containing c_u when u arrives. Each new facility w is mapped to the unique active coalition containing c_w just before w opens. For an isolated active coalition K, we let w_K denote the *nearest facility* to K's representative c_K at any given point in time. In other words, w_K is a function always associating the isolated active coalition K with the facility in F which is currently the nearest facility to c_K.

Final Assignment Cost. Let u be a demand currently assigned to a facility w with merge radius $m(w)$. The *final assignment cost* of u, denoted by $\overline{d}_u$, is equal to $\min\{d(u,w){+}\frac{x}{x-3}\, m(w), (1{+}\frac{1}{\lambda}) \max\{d(c_K, w), \lambda D(K)\}{+}d^*_u\}$ if u is mapped to an isolated active coalition K and w is currently the nearest facility to c_K, and equal to $d(u, w) + \frac{x}{x-3}\, m(w)$ otherwise. If u is currently assigned to a facility w, it will remain assigned to a facility in $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$ (Lemma 2). We can also prove

[1] A set system is *laminar* if it contains no intersecting pair of sets. The sets K, K' form an *intersecting pair* if neither of $K \setminus K'$, $K' \setminus K$ and $K \cap K'$ are empty. A laminar set system on F^* is *complete* if it contains F^* and every singleton set $\{c\}$, $c \in F^*$.

that if u is mapped to an isolated active coalition K and is currently assigned to w_K, then u's assignment cost will never exceed $(1+\frac{1}{\lambda})\max\{d(c_K, w_K), \lambda D(K)\}+d_u^*$. Therefore, the final assignment cost of u according to the current algorithm's configuration is an upper bound on its actual assignment cost in the future.

Inner and Outer Demands. A demand u mapped to a non-isolated active coalition K is *inner* if $d_u^* < D_N(K)$, and *outer* otherwise. A demand u mapped to an isolated active coalition K is *inner* if $d_u^* < \frac{1}{\lambda}\max\{d(c_K, w'_K), \lambda D(K)\}$, and *outer* otherwise. In this definition, w'_K denotes the nearest facility to c_K at u's assignment time.

We can prove that the final assignment cost of each outer demand is within a constant factor from its optimal assignment cost. From now on, we entirely focus on bounding the total assignment cost of inner demands throughout the execution of the algorithm.

Good and Bad Inner Demands. At any point in time, the set of *good demands* of an *isolated* active coalition K, denoted by G_K, consists of the inner demands of K which have *always* (i.e., from their assignment time until the present time) been assigned to w_K (i.e., the nearest facility to c_K). G_K is empty as long as K is either not active or non-isolated. We call *bad* every inner demand of K which is not good.

Each new inner demand mapped to an isolated active coalition K is initially assigned to the nearest facility to c_K because this facility is much closer to c_K than any other facility (see also Lemma 1) and inner demands are included in a small ball around c_K. Hence, each new inner demand mapped to K becomes good. An inner demand remains good as long as its assignment cost converges to its optimal assignment cost. In particular, a good inner demand becomes bad as soon as either K becomes broken or the location of the nearest facility to c_K changes and the facility at the former location w_K is not merged with the facility at the new location w'_K. A bad demand can never become good again.

With the exception of good demands, each demand is *irrevocably* charged with its final assignment cost at its assignment time. We keep track of the actual assignment cost of good demands until they become bad. This is possible because the good demands of an isolated active coalition K are always assigned to the nearest facility to c_K. Good demands are irrevocably charged with their final assignment cost at the moment they become bad.

Isolated Coalitions. We use the following lemma to bound the assignment cost of the inner demands mapped to an isolated active coalition K.

Lemma 4. *Let K be an isolated active coalition. The total actual and the total final assignment cost of the good inner demands of K is:*
$\sum_{u\in G_K} d(u, w_K) < 2\beta f + 3\sum_{u\in G_K} d_u^*$ *and* $\sum_{u\in G_K} \overline{d}_u < 4.5\beta f + 7\sum_{u\in G_K} d_u^*$

Proof Sketch. We sketch the proof of the first inequality. The second inequality is derived from the first one using the definition of the final assignment cost.

The proof is by induction over a sequence of merge operations where the former nearest facility to c_K is merged with the new nearest facility to c_K. Let

w be the nearest facility to c_K, i.e., $w_K = w$. We can show that for any isolated coalition K, the location of the nearest facility to c_K cannot change until a new facility mapped to K opens. Let w' be the next facility mapped to K and let G_K be the set of good demands just before w' opens. By Lemma 1, the location of the nearest facility to c_K changes from $w_K = w$ to $w'_K = w'$ as soon as w' opens. We inductively assume that the inequality holds just before w' opens, and show that it remains valid until either the location of the nearest facility to c_K changes again or K becomes broken.

If w is not merged with w', the set of good demands becomes empty and the inequality holds trivially. If w is merged with w', we can apply Lemma 1 and show that for every $u \in G_K$, $d(u, w') \leq \frac{1}{2}d(u, w) + \frac{3}{2}d^*_u$. Just after w has been merged with w', the set of good demands remains G_K, but each good demand is now assigned to w'. Hence, we can apply the inductive hypothesis and the inequality above and prove that $\sum_{u \in G_K} d(u, w') \leq \beta f + 3 \sum_{u \in G_K} d^*_u$.

As long as w' remains the nearest facility to c_K and K remains an isolated active coalition, each new inner demand mapped to K is initially assigned to w' and becomes a good demand. The rule for opening new facilities and Ineq. (1) imply that the initial assignment cost of these demands is at most βf. □

As long as K remains an isolated active coalition, it holds a credit of $\mathrm{O}(\beta f)$ which absorbs the additive term of $2\beta f$ corresponding to the part of the actual assignment cost of good inner demands which cannot be charged to their optimal assignment cost. The good inner demands of K are charged with their final assignment cost as soon as they become bad and G_K becomes empty. If G_K becomes empty because K becomes broken, the additive term of $4.5\beta f$ in the final assignment cost of the demands becoming bad is charged to K's credit. Otherwise, G_K becomes empty because the location of the nearest facility to c_K has changed and the facility w at the previous location w_K is not merged with the new facility w' at the new location w'_K. By Lemma 3, the facility w must be a supported facility. Hence, the additive term of $4.5\beta f$ can be charged to the optimal assignment cost of the demands in B_w, since $3x\,\mathrm{Asg}^*(B_w) \geq \beta f$. Lemma 1 implies that each supported facility is charged with the final assignment cost of some good demands which become bad at most once.

Non-isolated Coalitions. The inner demands mapped to non-isolated active coalitions are irrevocably charged with their final assignment cost at their assignment time. To bound their total final assignment cost, we develop a standard potential function argument based on the notion of *unsatisfied inner demands*. The set of unsatisfied inner demands of a non-isolated active coalition K, denoted by N_K, consists of the inner demands of K which are currently unsatisfied, i.e., in the set L. N_K is empty as long as K is either isolated or not active.

Lemma 5. *For every non-isolated active coalition K, $|N_K| \cdot g(c_K) \leq (\psi + 4)\gamma^2\beta f$.*

As long as K remains a non-isolated active coalition, c_K has a deficit of $5\,|N_K| \cdot g(c_K)$, which, by Lemma 5, never exceeds $5(\psi+4)\gamma^2\beta f$. If a new demand u is an inner demand of K and remains unsatisfied at its assignment time, we can prove that its final assignment cost is at most $5\,g'_u(c_K)$ (we recall that $g'_u(c_K)$ denotes the configuration distance of c_K at u's assignment time). Since u is added to N_K, the increase in the deficit of c_K compensates for the final assignment cost of u.

The deficit of c_K is 0 before K becomes active and after K has become either isolated or broken. Hence, the total final assignment cost of the inner demands mapped to K while K remains a non-isolated active coalition does not exceed the total decrease in the deficit of c_K. Each time the configuration distance $g(c_K)$ decreases but the set of unsatisfied inner demands N_K remains unaffected, the deficit of c_K decreases by a factor of $\ln(\frac{g(c_K)}{g'(c_K)})$. We can also prove that each time opening a new facility causes some demands to become satisfied and be removed from N_K, $g(c_K)$ decreases by a factor of 3. Thus, the deficit of c_K again decreases by a factor of $\ln(\frac{g(c_K)}{g'(c_K)})$. A non-isolated active coalition K becomes either isolated or broken no later than $g(c_K)$ has decreased by a factor of $(1+\frac{1}{\rho})\gamma^2$. Therefore, the total decrease in the deficit of c_K is at most $\mathrm{O}(5(\psi+4)\gamma^2\beta\ln((1+\frac{1}{\rho})\gamma^2)f)$. We increase this cost by $\mathrm{O}(\beta\ln((1+\frac{1}{\rho})\gamma^2)f)$ to take into account the final assignment cost of the inner demands of K which open a new facility and do not remain unsatisfied at their assignment time.

Concluding the Proof of Theorem 1. Let $u_1,\dots,u_n$ be the demand sequence considered by IFL. Putting everything together, we conclude that after the demand u_j has been considered, the facility cost of IFL does not exceed $a_1\mathrm{Fac}^*+b_1\mathrm{Asg}^*_j$ and the assignment cost of IFL does not exceed $a_2\mathrm{Fac}^*+b_2\mathrm{Asg}^*_j$, where $\mathrm{Asg}^*_j=\sum_{i=1}^{j} d^*_{u_i}$, and $a_1=1$, $a_2=2\beta\ln(3\gamma^2)(5(\psi+4)\gamma^2+3)$, $b_1=3x/\beta$, and $b_2=4((\rho+1)\gamma^2+2)+14x$. With a more careful analysis, we can improve a_2 to $4\beta\log(\gamma)(12(\psi+2)+3)$ and b_2 to $(\lambda+2)(8\psi+25)$.

3 An Incremental Algorithm for k-Median

The incremental algorithm for k-Median is based on the following standard lemma. We recall that a_1, a_2, b_1, and b_2 denote the constants in the performance ratio of IFL.

Lemma 6. *Let* Asg^* *be the optimal assignment cost for an instance of k-Median. For any* $\Lambda > 0$, IFL *with facility cost* $f=\frac{b_2}{a_2}\frac{\Lambda}{k}$ *maintains a solution of assignment cost no greater than* $(a_2+b_2)\mathrm{Asg}^*+b_2\Lambda$ *with no more than* $(a_1+a_2\frac{b_1}{b_2}\frac{\mathrm{Asg}^*}{\Lambda})k$ *medians.*

The deterministic version of the incremental algorithm operates in phases using IFL as a building block. Phase i is characterized by an upper bound Λ_i on the optimal assignment cost for the demands considered in the current phase. The algorithm invokes IFL with facility cost $f_i=\frac{b_2}{a_2}\frac{\Lambda_i}{k}$. By Lemma 6, as soon as either the number of medians exceeds νk or the assignment cost exceeds $\mu\Lambda_i$, where ν,μ are appropriately chosen constants, we can be sure that the optimal cost has also exceeded Λ_i. Then, the algorithm merges the medians produced by the current phase with the medians produced by the previous phases, increases the upper bound by a constant factor, and proceeds with the next phase.

Theorem 2. *There exists a deterministic incremental algorithm for k-Median which achieves a constant performance ratio using* $\mathrm{O}(k)$ *medians.*

The randomized version also proceeds in phases. Phase i invokes the algorithm Para_Cluster [4] with facility cost $\hat{f}_i = \frac{\Lambda_i}{k(\log n+1)}$ to generate a modified instance which can be represented in a space efficient manner. The modified instance contains the same number of demands, which now occupy only $\mathrm{O}(k \log^2 n)$ different gathering points. Then, the algorithm invokes IFL with facility cost $f_i = \frac{b_2}{a_2} \frac{\Lambda_i}{k}$ to cluster the modified instance.

For an incremental implementation, each new demand is first moved to a gathering point. Then, a new demand located at the corresponding gathering point is given to IFL, which assigns it to a median. Both actions are completed before the next demand is considered. The current phase ends if either the number of gathering points, the gathering cost, the number of medians, or the assignment cost on the modified instance become too large. We should emphasize that IFL treats the demands moved to the same gathering point as *different demands* and may put them in different clusters.

In contrast to the deterministic version which does not require any advance knowledge of n, the randomized version needs to know $\log n$ in advance.

Theorem 3. *There exists a randomized incremental algorithm for k-Median which runs in* $\mathrm{O}(nk^2 \log^2 n)$ *time and* $\mathrm{O}(k^2 \log^2 n)$ *space and achieves a constant performance ratio whp. using* $\mathrm{O}(k)$ *medians.*

References

1. A. Anagnostopoulos, R. Bent, E. Upfal, and P. Van Hentenryck. A Simple and Deterministic Competitive Algorithm for Online Facility Location. TR CS-03-16, Brown University, 2003.
2. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.
3. M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental Clustering and Dynamic Information Retrieval. In *Proc. of STOC '97*, pp. 626–635, 1997.
4. M. Charikar, L. O'Callaghan, and R. Panigrahy. Better Streaming Algorithms for Clustering Problems. In *Proc. of STOC '03*, pp. 30–39, 2003.
5. M. Charikar and R. Panigrahy. Clustering to Minimize the Sum of Cluster Diameters. In *Proc. of STOC '01*, pp. 1–10, 2001.
6. D. Fotakis. On the Competitive Ratio for Online Facility Location. In *Proc. of ICALP '03*, LNCS 2719, pp. 637–652, 2003.
7. D. Fotakis. Incremental Algorithms for Facility Location and k-Median. Available from `http://users.auth.gr/~fotakis/data/incremen.pdf`, 2004.
8. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering Data Streams. In *Proc. of FOCS '00*, pp. 359–366, 2000.
9. M.R. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on Data Streams. TR 1998-011, DEC Systems Research Center, 1998.
10. A. Meyerson. Online Facility Location. In *Proc. of FOCS '01*, pp. 426–431, 2001.