Alcom-FT Technical Report Series
ALCOMFT-TR-03-156

# Incremental Algorithms for Facility Location and $k$-Median*

Dimitris Fotakis

Computer Technology Institute
Riga Feraiou 61, 26221 Patras, Greece
Email: fotakis@cti.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 which achieves a constant performance ratio and the first incremental algorithm for $k$-Median which achieves a constant performance ratio using $O(k)$ medians, thus resolving an open question of [5]. Our algorithm is based on a merge rule which ensures that the algorithm's configuration monotonically converges to each optimal facility according to a certain notion of distance. Using this property, we reduce the general case to the special case that the optimal solution consists of a single facility.

## 1 Introduction

The problems of Facility Location and $k$-Median find many applications in the areas of network design and data clustering and have been the subject of intensive research over the last decade (e.g., see [19] for a survey and [8] for approximation algorithms and applications). In addition to the offline setting, there are many practical applications where the demand points arrive online and the solution must be constructed incrementally using no information about future demands. Meyerson [17] was the first to consider Facility Location in the online setting.

The model of incremental algorithms for clustering problems is also motivated by similar applications where the demand sequence is not known in advance and the algorithm must maintain a consistently good clustering using a restricted set of operations which result in a solution of hierarchical structure. The framework of incremental clustering was introduced by Charikar, Chekuri, Feder and Motwani in [3], where both a detailed description and a careful justification of the model are presented.

In *Incremental $k$-Median* (see also [5]), the demand points arrive one at a time. Each new 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 its 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

* This work was partially supported by the Future and Emerging Technologies 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.

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 the distance from the median of the cluster the demand is currently included in.

The definition of *Incremental Facility Location* is similar. 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. As before, the assignment cost of a demand is the distance from the facility the demand is currently assigned to.

We evaluate the performance of incremental algorithms using the *performance ratio* [3]. An incremental algorithm achieves a performance ratio of $c$ if for all demand sequences, the cost incurred by the algorithm is at most $c$ times the cost incurred by an optimal offline algorithm, which has full knowledge of the demand sequence, on the same instance. We consider the metric versions of Facility Location and $k$-Median and assume that the demand points are located in a metric space. We also use $n$ to 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 to merge some existing clusters with each other. These decisions are irrevocable because clusters cannot be broken up. In addition, the definition of the performance ratio is essentially identical to the definition of the competitive ratio (e.g., [2]). Hence, Incremental $k$-Median could also be regarded as the natural online version of $k$-Median. Even though it would better reflect our motivation and point of view, we have avoided casting Incremental $k$-Median as "Online $k$-Median". The most important reason is that in contrast to online problems (e.g., [2]), where there usually exists a notion of irrevocable cost associated with each irrevocable decision made by the algorithm, in the context of $k$-Median, we are not aware of any simple and natural notion of irrevocable cost which could be associated with the irrevocable decision that a demand is clustered together with some other demands.

Incremental algorithms also bear a resemblance to one-pass streaming algorithms for clustering problems (e.g., see [11] for a formulation of the streaming model and [10, 6] for applications to $k$-Median). However, 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 data. A streaming algorithm for $k$-Median is not restricted in terms of the solution's structure or the set of operations available (e.g., the algorithm of [6]). On the the other hand, incremental algorithms must maintain a good hierarchical clustering by making irrevocable decisions. As for time and space efficiency, although preference is given to quasi-linear-time and small-space algorithms, we only explicitly require that incremental algorithms should run in polynomial time (see also [3]). Nevertheless, all known incremental algorithms for clustering problems can be either directly regarded as or easily transformed to time and space efficient one-pass streaming algorithms (e.g., [3, 10, 5, 6]).

**Previous Work.** Charikar, Chekuri, Feder and Motwani [3] introduce the framework of incremental clustering and present incremental algorithms for $k$-Center (i.e., minimizing the maximum cluster radius) which achieve a constant performance ratio using $k$ clusters. Charikar and Panigrahy [5] present an incremental algorithm for Sum $k$-Radius (i.e., minimizing the sum of cluster radii) which achieves a constant performance ratio using $\mathrm{O}(k)$ clusters.

The definition of Incremental $k$-Median directly follows from the framework introduced in [3]. The incremental version of $k$-Median was first considered by Charikar and Panigrahy in [5], where it is shown that no deterministic algorithm maintaining at most $k$ clusters can achieve a performance ratio better than $\Omega(k)$. Hence, we relax the requirement on the number of clusters allowing the algorithm to maintain up to $c\,k$ clusters, where $c > 1$ is a constant. Determining whether there exists an incremental algorithm for $k$-Median which achieves a constant performance ratio using $\mathrm{O}(k)$ medians is suggested as an interesting open problem in [5].

The only known incremental algorithms for $k$-Median are the one-pass streaming algorithms of [10] and [6]. More specifically, the streaming algorithms of Guha, Mishra, Motwani, and O'Callaghan [10] can be regarded as incremental algorithms under the assumption that the number of demands $n$ is known in advance. For $k$ much smaller than $n^\epsilon$, their algorithms achieve a performance ratio of $2^{\mathrm{O}(1/\epsilon)}$ using $n^\epsilon$ medians and run in $\mathrm{O}(nk\,\mathrm{poly}(\log n))$ time and $n^\epsilon$ space. The best known streaming algorithm for $k$-Median is the one-pass algorithm of Charikar, O'Callaghan, and Panigrahy [6]. Under the assumption that $n$ is known in advance, this algorithm can be easily transformed to an incremental algorithm which achieves a constant performance ratio with high probability (whp.[1]) 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.

Meyerson [17] considers the problem of Online Facility Location, where the demand points arrive one at a time and must be irrevocably assigned to either an existing or a new facility upon arrival. In [17], a randomized $\mathrm{O}(\frac{\log n}{\log\log n})$-competitive algorithm and a lower bound of $\omega(1)$ are presented. In [7], 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. The lower bounds of [17, 7] hold only if the decision of opening a facility at a particular location is irrevocable. Hence, they do not apply to the incremental version, where the algorithm can take advantage of merging existing clusters with each other.

**Related Work on Facility Location and $k$-Median.** In the offline case, where the demand set is fully known in advance, there are constant factor approximation algorithms for Facility Location based on Linear Programming rounding (e.g., [20, 21]), local search (e.g., [14, 4,

[1] Throughout this paper, "whp." means "with probability at least $1 - \mathrm{O}(1/n)$".

1]), and the primal-dual method (e.g., [13, 12]). The best known polynomial-time algorithm achieves an approximation ratio of 1.52 [15], while no polynomial-time algorithm can achieve an approximation ratio less than 1.463 unless NP $=$ DTIME($n^{\mathrm{O}(\log\log n)}$) [9]. For $k$-Median, the best known polynomial-time algorithm achieves an approximation ratio of $3 + \mathrm{o}(1)$ [1], while no polynomial-time algorithm can achieve an approximation ratio less than $1 + 2/e$ unless NP $=$ DTIME($n^{\mathrm{O}(\log\log n)}$) [12]. As it is also observed in [17], our setting should not be confused with the setting of [16, 18], where the demand set is fully known in advance and the number of facilities/medians increases online.

**Results.** We present the first incremental algorithm for metric Facility Location which achieves a constant performance ratio. Using this algorithm 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]. Our algorithm is deterministic, runs in $\mathrm{O}(n^2 k)$ time and $\mathrm{O}(n)$ space, and is the first incremental algorithm for $k$-Median which does not assume any advance knowledge of $n$. Combining our techniques with the techniques of [6], 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 an one-pass streaming algorithm for $k$-Median. Similarly to the algorithms of [10, 6], the randomized version of our algorithm assumes that a constant factor approximation to $\log n$ is known in advance.

**Intuition and Techniques.** Our initial motivation for studying the incremental versions of Facility Location and $k$-Median came from the observation that the lower bounds of [17, 7] on the competitive ratio for Online Facility Location do not apply if the algorithm can also merge existing clusters with each other. Nevertheless, the lower bound of [7] implies that any algorithm maintaining $\mathrm{o}(k \log n)$ medians 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. In other words, any analysis not establishing that merge operations can also *decrease* the algorithm's assignment cost cannot avoid either a logarithmic blow-up in the number of clusters or a performance ratio which depends on $n$. For instance, any analysis treating merge as a black-box operation could not achieve a constant performance ratio unless it uses $\Omega(k \log n)$ medians/facilities (e.g., the algorithm of [6]). Hence, we need a novel approach to establish a constant performance ratio for Incremental Facility Location.

We use a modification of the deterministic algorithm of [7] for opening new facilities. The first new idea is to use a rule for closing existing facilities (namely, a merge rule) which ensures that the algorithm's configuration monotonically converges to each optimal facility according to a certain notion of distance. To achieve this goal, we introduce a merge rule which is based on distance rather than cost considerations. More specifically, each facility maintains its merge radius which is non-increasing with time. The merge radius of a new facility is initialized to a fraction of its distance from the nearest existing facility. Then, if a new facility opens within the merge radius of an existing facility, the existing facility is merged with the new one. It is not difficult to prove that in contrast to online algorithms for Facility Location [17, 7], this algorithm does not suffer from facility proliferation. On the other hand, it retains the basic property of online algorithms. Namely, the algorithm's configuration keeps converging to each optimal facility (according to a certain notion of distance) despite some existing facilities being closed.

The technically involved part of our analysis has to do with the algorithm's assignment cost. If the optimal solution consists of a single facility, we show that each new facility is much closer to the optimal facility than any existing facilities. Since we keep merging existing facilities with new ones, the algorithm's assignment cost gradually converges to the optimal assignment cost. Using an argument similar to those used in the analysis of online algorithms [17, 7], we also bound the increase in the algorithm's assignment cost before a new facility opens.

The second ingredient of our approach is a reduction from the general case to the special case above. Our technique, which is also implicit in [7], may be of independent interest and find applications to online problems different from the incremental and online versions of Facility Location.

Let $K$ be any subset of optimal facilities. As long as $K$'s diameter is much smaller than $K$'s distance to the current algorithm's configuration (according to the notion of distance mentioned above), we say that $K$ forms a *coalition*. Then, $K$ essentially behaves as a single optimal facility. As long as $K$ is closer to the algorithm's configuration than any other optimal facility, we say that $K$ is *isolated*. Then, the part of the algorithm's configuration related to $K$ is not affected by any new demands not assigned to $K$ in the optimal solution. Intuitively, as long as $K$ remains isolated, it behaves as if there were no optimal facilities outside $K$. The crucial observation is that as long as $K$ forms an *isolated coalition*, we can bound the assignment cost of the demands assigned to $K$ in the optimal solution similarly to the case that the optimal solution consists of a single facility.

Any hierarchical decomposition of the optimal facility locations can be used to maintain a minimal collection of *active* coalitions always forming a partitioning of the set of optimal facilities. Then, the assignment cost of each new demand can be charged to the unique active coalition which contains the optimal facility the demand is assigned to. We can prove that the assignment cost charged to isolated active coalitions is within a constant factor from the corresponding optimal cost. To bound the assignment cost charged to non-isolated active coalitions, we show that there exists a hierarchical decomposition such that a coalition can remain both non-isolated and active only if its distance to the algorithm's configuration lies in a short interval.

The incremental algorithm for $k$-Median operates in phases. Using an estimated upper bound on the optimal cost, each phase determines a facility cost and invokes the incremental algorithm for Facility Location. If the upper bound is not valid anymore, the algorithm increases it by a constant factor and proceeds with the next phase. This is essentially the doubling approach used in online load balancing (e.g., [2, Chapter 12]). The same approach is also used in the incremental algorithms for $k$-Center [3] and Sum $k$-Radius [5], and in the streaming algorithm for $k$-Median of [6]. To improve time and space efficiency, we use a modification of the randomized algorithm of [6] to move each new demand from its original location to one of $\mathrm{O}(k \log^2 n)$ gathering points. Then, instead of the original demand, the incremental algorithm for Facility Location receives a demand located at the corresponding gathering point.

**Notation.** Throughout this paper, we only consider unit demands by allowing multiple demands to be located at the same point. We always use $n$ to denote the total number of demands. 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 a subspace $M' \subseteq M$, $D(M') = \max_{u,v \in M'}\{d(u, v)\}$ denotes the diameter of $M'$. For a point $u \in M$ and a subspace $M' \subseteq M$, $d(M', u) = \min_{v \in M'}\{d(v, u)\}$ denotes the distance between $u$ and the nearest point in $M'$. It is $d(\emptyset, u) = \infty$. For subspaces $M', M'' \subseteq M$, $d(M', M'') = \min_{u \in M''}\{d(M', u)\}$ denotes the minimum distance between a point in $M'$ and a point in $M''$. For a subspace $M' \subseteq M$, $\mathrm{sep}(M') = d(M', M \setminus M')$ denotes the distance separating any point in $M'$ from any point not in $M'$. It is $\mathrm{sep}(\emptyset) = \mathrm{sep}(M) = \infty$. For a point $u \in M$ and a non-negative number $r$, $\mathrm{Ball}(u, r)$ denotes the ball of center $u$ and radius $r$, $\mathrm{Ball}(u, r) = \{v \in M : d(u, v) \leq r\}$.

We assume that the algorithm has access to the distance function through a constant time oracle which given any two points $u, v$, returns their distance $d(u, v)$. Moreover, claiming a space complexity of $\mathrm{O}(g(n, k))$, we mean that the algorithm runs in the space required to represent $\mathrm{O}(g(n, k))$ weighted points each of weight not exceeding $n$. These conventions are common (e.g., [3, 10, 6]) and do not affect the comparison of our results with previous work.

## 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*.

We use the notion of unsatisfied demands (the set $L$) to ensure that each demand contributes to the opening cost of a new facility at most once. A demand becomes unsatisfied and is added to $L$ upon arrival. Each unsatisfied demand holds a *potential* which is always equal to its distance from the nearest facility. If the neighborhood $B_u$ of a new demand $u$ has accumulated a potential of $\beta f$, a new facility located at the same point with $u$ opens. Then, the unsatisfied demands in $B_u$ lose their potential, become satisfied, and are removed from $L$.

Each facility $w \in F$ maintains the set $C(w)$ of the demands currently assigned to $w$ and the set $\mathrm{Init}(w) \subseteq C(w)$ of the demands *initially assigned* to $w$. The demands in $\mathrm{Init}(w)$ are assigned to $w$ when they arrive, while the demands in $C(w) \setminus \mathrm{Init}(w)$ have been initially assigned to a facility different from $w$. Furthermore, each facility $w \in F$ maintains its *merge radius* $m(w)$ and the corresponding *merge ball* $\mathrm{Ball}(w, m(w))$. The algorithm ensures that $w$ is the only facility in its merge ball. 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 $w'$ is included in $w$'s merge ball, $w$ is merged with $w'$. Namely, $w$ is closed and removed from $F$, and any demand currently assigned to $w$ is re-assigned to $w'$. The algorithm keeps decreasing $m(w)$ to maintain the invariant that no merge operation can dramatically increase the total assignment cost of the demands in $\mathrm{Init}(w)$. More specifically, it is always the case 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 initially 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, a crude analysis shows that 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 any $x \geq 18$, $\beta \geq \frac{4(x+1)}{x-8}$, and $\psi \in [\max\{\frac{6\beta}{2\beta-3}, 4\}, 5]$,* IFL *achieves a constant performance ratio.*

**Preliminaries.** For an arbitrary fixed sequence of demands, we compare the algorithm's cost with the cost of an arbitrary fixed add-optimal facility configuration[2]. We always denote this solution by $F^*$ and refer to it as *the optimal solution*. To avoid confusing the algorithm's facilities with the facilities in $F^*$, we use the term *optimal center*, or simply *center*, to refer to an optimal facility in $F^*$ and the term *facility* to refer to an algorithm's facility in $F$.

The optimal solution $F^*$ consists of $k$ centers $c_1, c_2, \ldots, c_k$. Each demand is assigned to the nearest center in $F^*$. For a demand $u$, $c_u$ denotes the optimal center $u$ is assigned to. We use the clustering induced by $F^*$ to map the demands and the algorithm's facilities to optimal centers. In particular, a demand $u$ is always mapped to $c_u$, i.e., the optimal center $u$ is assigned to. Similarly, a facility $w$ is mapped to the nearest optimal center denoted by $c_w$. Also, let $d_u^* = d(c_u, u) = d(F^*, u)$ denote the optimal assignment cost of $u$, let $\mathrm{Fac}^* = kf$ be the optimal facility cost, and let $\mathrm{Asg}^* = \sum_{i=1}^n d_u^*$ be the optimal assignment cost.

[2] A facility configuration $F$ is *add-optimal* if its total cost cannot decrease by adding a new facility to $F$. Formally, for any $w$, $f + \sum_u d(F \cup \{w\}, u) \geq \sum_u d(F, u)$.

```
Let x, β, and ψ be appropriately chosen constants
F ← ∅; L ← ∅; /* Initialization */
For each new demand u:
    L ← L ∪ {u}; r_u ← d(F,u)/x;
    B_u ← Ball(u, r_u) ∩ L; Pot(B_u) = Σ_{v∈B_u} d(F,v);
    if Pot(B_u) ≥ βf then
        Let w' be the location of u;
        open(w'); L ← L \ B_u;
        for each w ∈ F \ {w'} do
            if d(w,w') ≤ m(w) then merge(w → w');
    Let w be the facility in F closest to u;
    update_merge_radius(m(w));
    initial_assignment(u, w);

open(w')
    F ← F ∪ {w'}; Init(w') ← ∅;
    C(w') ← ∅; m^(1)(w') ← 3 r_u;

merge(w → w')
    F ← F \ {w}; C(w') ← C(w') ∪ C(w);

update_merge_radius(m(w))
    m^(2)(w) = max{r : |Ball(w, r/ψ) ∩ (Init(w) ∪ {u})| · r ≤ βf};
    m(w) = min{m^(1)(w), m^(2)(w)};

initial_assignment(u, w)
    Init(w) ← Init(w) ∪ {u}; C(w) ← C(w) ∪ {u};
```

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

In addition to $x$, $\beta$, and $\psi$, let $\lambda = 3x+2$, $\rho = (\psi+2)(\lambda+2)$, and $\gamma = 12\rho$ be also constants. Let also $u_1, \ldots, u_n$ be the demand sequence considered by IFL. We show that after the demand $u_j$ has been considered, $1 \leq j \leq n$, 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 $a_1 = 1$, $a_2 = 2\beta\ln(3\gamma^2)(5(\psi+4)\gamma^2+3)$, $b_1 = \frac{3x}{\beta}$, and $b_2 = 4((\rho+1)\gamma^2+2)+14x$. With a more careful analysis, we can improve $a_2$ and $b_2$ to $a'_2 = 4\beta\log(\gamma)(12(\psi+2)+3)$ and $b'_2 = (\lambda+2)(8\psi+25)$. Moreover, we can remove the assumption that $F^*$ is add-optimal by replacing the bound on the algorithm's assignment cost with $\max\{a_2, b_2\}(\mathrm{Fac}^* + \mathrm{Asg}^*_j)$.

Every time we want to explicitly refer to the algorithm's configuration (or some function of it) at the moment a demand is considered/facility opens, we use the demand's/facility's identifier as a subscript. Moreover, we always assume that the algorithm first updates its configuration and then performs the demand's initial assignment. Hence, we distinguish between the algorithm's configuration at the demand's arrival and assignment times using plain symbols to refer to the former and primed symbols to refer to the latter time. For example, for a demand $u$, $F_u/F'_u$ is the facility configuration at $u$'s arrival/assignment time. Similarly, for a facility $w$, $F_w/F'_w$ is the facility configuration just before/after $w$ opens. Saying that an existing facility $w$ is merged with a new facility $w'$, we mean that 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). We proceed to establish the basic properties of IFL.

**Lemma 1.** *Let $\beta \geq \frac{4(x+1)}{x-8}$. Then, for any facility $w$ mapped to $c_w$, $d(c_w, w) \leq \frac{d(F_w, c_w)}{3}$.*

*Proof Sketch.* To reach a contradiction, let us assume that $w$ is a facility such that $d(c_w, w) > \frac{d(F_w, c_w)}{3}$. Since $B_w \subseteq \mathrm{Ball}(w, \frac{d(F_w, w)}{x})$, we can show that for any $u \in B_w$, $d(u, w) < \frac{4}{x-4}\, d^*_u$ 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 any $\beta \geq \frac{4(x+1)}{x-8}$, $f + \sum_{u \in B_w} d(u, w) < \sum_{u \in B_w} d^*_u$, which contradicts to the add-optimality of $F^*$. □

**Proposition 1.** *For any facility $w$, there will always exist a facility in $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$ and any demand currently assigned to $w$ will always remain assigned to a facility in $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$.*

*Proof.* The proposition is true as long as $w$ remains open. If $w$ is merged with a new facility $w'$, we inductively assume that the proposition is true for $w'$. Then, the proposition follows from the observation that $\mathrm{Ball}(w', \frac{x}{x-3}\, m(w'))$ is included in $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$. □

**Facility Cost.** We distinguish between *supported* facilities, whose opening cost can be charged to the optimal assignment cost, 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. Since each demand contributes to the opening cost of a facility at most once, the total cost of supported facilities is at most $\frac{3x}{\beta}\, \mathrm{Asg}^*$. Next, we prove that there always exists at most one unsupported facility mapped to each optimal center. Therefore, the algorithm's facility cost does not exceed $\mathrm{Fac}^* + \frac{3x}{\beta}\, \mathrm{Asg}^*$.

**Lemma 2.** *Let $w$ be an unsupported facility mapped to an optimal center $c_w$, and let $w'$ be any new facility also mapped to $c_w$. If $w'$ opens while $w$ is still open, then $w$ is merged with $w'$.*

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

To prove that $m^{(1)}(w) \geq \frac{3}{2}\, d(c_w, w)$, we show that for any unsupported facility $w$, $d(c_w, w) < \frac{4\, d(F_w, c_w)}{3(x-4)}$. We first observe that there must be a demand $u \in B_w$ such that $d^*_u < \frac{1}{3x}\, d(F_w, u)$, because $w$ is an unsupported facility. Then, the claim follows from the fact that the radius of $B_w$ is $\frac{d(F_w, w)}{x}$. Since $m^{(1)}(w) = 3\, \frac{d(F_w, w)}{x}$, we obtain that $m^{(1)}(w) \geq \frac{3}{2}\, d(c_w, w)$.

The prove that $m^{(2)}(w) \geq \frac{3}{2}\, d(c_w, w)$, we first observe that $m^{(2)}(w)$ cannot become smaller than $\frac{3}{2}\, d(c_w, w)$ unless the number of demands in $\mathrm{Ball}(w, \frac{3\, d(c_w, w)}{2\psi})$ becomes greater than

$\frac{2\beta f}{3d(c_w,w)}$. This contradicts to the add-optimality of $F^*$ because for any $\psi \geq \frac{6\beta}{2\beta-3}$, these demands are closer to each other than to any optimal center in $F^*$. □

**Assignment Cost.** We first introduce the basic notions used in the analysis of the algorithm's assignment cost.

*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 $c$ (Proposition 1).

*Coalitions.* A set of optimal centers $K \subseteq F^*$ with representative $c_K \in K$ forms a *coalition* as long as $g(c_K) \geq \rho D(K)$. A coalition $K$ becomes *broken* as soon as $g(c_K) < \rho D(K)$. A coalition $K$ is *isolated* if $g(c_K) \leq \frac{1}{3}\,\mathrm{sep}(K)$ and *non-isolated* otherwise. A *hierarchical decomposition* $\mathcal{K}$ of $F^*$ is any complete laminar set system[3] on $F^*$. Any hierarchical decomposition of $F^*$ contains at most $2\,|F^*| - 1$ distinct 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 covers $F^*$. 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., $g(c_K) \geq \rho D(K)$), while any superset of $K$ in $\mathcal{K}$ has become broken (i.e., for any $K' \in \mathcal{K}, K \subset K'$, it is $g(c_{K'}) < \rho D(K')$). 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 (isolated) can become active (resp. non-isolated) again.

Let $D_N(K) = \max\{D(K), \frac{1}{3\rho}\,\mathrm{sep}(K)\}$. By definition, $K$ becomes either isolated or not active/broken as soon as $g(c_K) < \rho D_N(K)$. We can show that there is a hierarchical decomposition of $F^*$ such that no coalition $K$ becomes active before $g(c_K) < (\rho + 1)\gamma^2 D_N(K)$ (see also [7]). In the following, we always assume that the set of active coalitions is given by a fixed hierarchical decomposition $\mathcal{K}$ of $F^*$ such that for any non-isolated active coalition $K$, $\rho D_N(K) \leq g(c_K) < (\rho+1)\gamma^2 D_N(K)$. Using this property, we show that the algorithm's performance ratio is essentially determined by its performance ratio for isolated active coalitions, which can be analyzed similarly to the special case that $F^*$ consists of a single optimal center.

A new demand $u$ makes a coalition $K$ broken/isolated if $K$ has been active/non-isolated before $u$ and becomes broken/isolated after $u$. Each new demand $u$ is mapped to the unique active coalition $K_u$ containing $c_u$ when $u$ arrives. If $K_u$ is isolated (non-isolated) when $u$ arrives, we say that $u$ is a demand of the isolated (resp. non-isolated) active coalition $K_u$. 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 use $w_K$ to 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 mapping

[3] 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 all the singleton sets $\{c\}$, $c \in F^*$.

the isolated active coalition $K$ to the facility in $F$ which is currently the nearest facility to $c_K$. Lemma 1 and Proposition 1 imply that as long as $K$ is an isolated active coalition, $w_K$ is much closer to $c_K$ than any other facility and rapidly converges to $c_K$.

*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. Let $\mathrm{in}_N(K)$ denote the set of inner demands and $\mathrm{out}_N(K)$ denote the set of outer demands mapped to $K$ as long as $K$ is a non-isolated active coalition. 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. Thereby, the characterization of a demand $u$ as inner or outer is determined according to the updated algorithm's configuration at $u$'s assignment time, in contrast to the active coalition $u$ is mapped to, which is determined according to the algorithm's configuration at $u$'s arrival time. Let $\mathrm{in}_I(K)$ denote the set of inner demands and $\mathrm{out}_I(K)$ denote the set of outer demands mapped to $K$ as long as $K$ is an isolated active coalition. It is not difficult to prove that the assignment cost of outer demands is always within a constant factor from their optimal assignment cost. Hence, our main concern is to bound the assignment cost of inner demands.

*Good and Bad Inner Demands.* The set of *good demands* of an *isolated* active coalition $K$, denoted by $\mathrm{G}_K$, consists of the inner demands of $K$ which have *always* (i.e., from their assignment time until the present time) been assigned to the facility at the current location of $w_K$ (the nearest facility to $c_K$). We define $\mathrm{G}_K$ to be empty as long as $K$ is either not active or non-isolated. We call *bad* any 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. Hence, each new inner demand mapped to $K$ becomes good and is added to $\mathrm{G}_K$. An inner demand remains good until either $K$ becomes broken or the location of $w_K$ changes and the facility at the former location $w_K$ is not merged with the facility at the new location $w'_K$. Then, the demand becomes bad and can never become good again. Since $w_K$ converges to $c_K$, the actual assignment cost of good inner demands should converge to their optimal assignment assignment cost. In particular, we show that the total assignment cost of the demands in $\mathrm{G}_K$ never exceeds $2\beta f$ plus 3 times their optimal assignment cost.

*Unsatisfied Inner Demands.* The set of *unsatisfied inner demands* of a *non-isolated* active coalition $K$, denoted by $\mathrm{N}_K$, consists of the inner demands of $K$ which are currently unsatisfied (i.e., included in $L$). We define $\mathrm{N}_K$ to be empty as long as $K$ is either not active or isolated. Formally, $\mathrm{N}_K$ is equal to $\mathrm{in}_N(K) \cap L$ as long as $K$ is a non-isolated active coalition, and empty otherwise. We can prove that for any non-isolated active coalition $K$, the product $|\mathrm{N}_K| \cdot g(c_K)$ is at most $((\psi + 4)\gamma^2 + 1)\beta f$ .

*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 a demand $u$ is currently assigned to a facility $w$, then $u$ will remain assigned to a facility in $\mathrm{Ball}(w, \frac{x}{x-3}\, m(w))$ (Proposition 1). We can also prove that if $u$ is mapped to an isolated active coalition $K$ and is currently assigned to $w_K$, then $u$'s assignment cost can never exceed $(1 + \frac{1}{\lambda}) \max\{d(c_K, w_K), \lambda D(K)\} + d_u^*$. Therefore, the final assignment cost of $u$ is an upper bound on its future actual assignment cost.

**Analysis (sketch).** With the exception of good demands, each demand is *irrevocably* charged with its final assignment cost at its assignment time. Then, we do not have to worry about the demand's actual assignment cost anymore. On the other hand, we keep track of the actual assignment cost of good demands until they become bad. This is not difficult 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.* Let $K$ be an isolated active coalition with representative $c_K$. The assignment cost of the demands mapped to $K$ can be analyzed independently of the demands mapped to other active coalitions and similarly to the special case that there is a single optimal center.

More specifically, we show that for any isolated active coalition $K$, (i) any facility spending some time as the nearest facility to $c_K$ can be merged only with a new facility mapped to $K$, (ii) any new facility mapped to $K$ either makes $K$ broken or is at least 2.5 times closer to $c_K$ than the current location of $w_K$, (iii) an unsupported facility mapped to $K$ is merged with the next facility mapped to $K$, (iv) a new demand/facility not mapped to $K$ cannot change either the location of $w_K$ or the value of $g(c_K)$, (v) any inner demand of $K$ which does not make $K$ broken is initially assigned to the nearest facility to $c_K$, and (vi) for any outer demand $u$ which is mapped to $K$ and does not make $K$ broken, $\overline{d}_u \leq 4(\lambda + 2) d_u^*$. The analysis of the assignment cost of inner demands mapped to isolated active coalitions is based on the following lemma.

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

*Proof Sketch.* We sketch the proof of the first inequality. The second inequality can be 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., $w_K = w$. By (i) and (iv), the location of $w_K$ cannot change until a new facility mapped to $K$ opens. Let $w'$ be the next facility mapped to $K$ and let $\mathrm{G}_K$ be the set of good demands just before $w'$ opens. Wlog. we can assume that $w'$ does not make $K$ broken, since $\mathrm{G}_K$ becomes empty otherwise. Then, by (ii), $d(c_K, w') \le \frac{2}{5}\, d(c_K, w)$, and the location of the nearest facility to $c_K$ must change 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 we show that it remains valid until either the location of $w_K$ changes again or $K$ becomes broken.

If $w$ is not merged with $w'$, the set of good demands becomes empty. Then, the inequality trivially holds just after $w'$ opens. If $w$ is merged with $w'$, (ii) implies that for any $u \in \mathrm{G}_K$, $d(u, w') \le \frac{1}{2} d(u, w) + \frac{3}{2} d^*_u$. Just after $w$ has been merged with $w'$, the set of good demands remains $\mathrm{G}_K$, but each $u \in \mathrm{G}_K$ is now assigned to $w'$. Hence, $\sum_{u \in \mathrm{G}_K} d(u, w') \le \beta f + 3 \sum_{u \in \mathrm{G}_K} d^*_u$. As long as $w'$ remains the nearest facility to $c_K$ and $K$ remains an isolated active coalition, each new inner demand of $K$ is initially assigned to $w'$ (cf. (v) above) and becomes a good demand. Let $\mathrm{G}_K(w')$ be the set of good inner demands of $K$ whose initial assignment takes place while $w'$ is the nearest facility to $c_K$. We prove that the total initial assignment cost of the demands in $\mathrm{G}_K(w')$ is at most $\beta f$.

More specifically, if $d(c_K, w') \ge \lambda D(K)$, we observe that the demands in $\mathrm{G}_K(w')$ remain unsatisfied and hold a potential equal to their initial assignment cost (i.e., equal to their distance from $w'$) until either a new facility mapped to $K$ opens or $K$ becomes broken, because any facility which makes some of them satisfied or decrease their potential must be mapped to $K$ (see also (ii) above). In addition, since $d(c_K, w') \ge \lambda D(K)$, the neighborhood $B_v = \mathrm{Ball}(v, \frac{d(F_v, v)}{x}) \cap L$ of any new inner demand $v$ which is added to $\mathrm{G}_K(w')$ includes any demand in $\mathrm{G}_K(w')$, because $\frac{d(F_v, v)}{x} = \frac{d(w', v)}{x} > \frac{3}{\lambda} d(c_K, w')$ (recall that $\lambda = 3x + 2$), while for any $u \in \mathrm{G}_K(w')$, $d(u, v) < \frac{3}{\lambda} d(c_K, w')$, because $\mathrm{G}_K(w')$ consists of inner demands of $K$ being initially assigned after $w'$ opens. Therefore, the potential accumulated by $\mathrm{G}_K(w')$ is at most $\beta f$. On the other hand, if $d(c_K, w') < \lambda D(K)$, we observe that as long as $K$ remains active, it must be $m(w') \ge \psi(\lambda + 2) D(K)$ (recall that $\rho = (\psi + 2)(\lambda + 2)$). Hence, $\mathrm{G}_K(w') \subseteq \mathrm{Init}(w') \cap \mathrm{Ball}(w', \frac{m(w')}{\psi})$, and we can use Ineq. (1). □

Using a potential function argument based on Lemma 3 and the claims (i) - (vi) above, we can bound the assignment cost of the demands mapped to $K$. When $K$ becomes an isolated active coalition, it receives a credit of $7\beta f$, which is not used until $K$ becomes broken. Let $u$ be a new demand mapped to the isolated active coalition $K$. If $u$ makes $K$ broken, $K$'s credit is charged with $u$'s final assignment cost, which cannot exceed $2.5\beta f$. If $u$ is an outer demand and does not make $K$ broken, its final assignment cost cannot exceed $4(\lambda + 2) d^*_u$ by (vi). If $u$ is an inner demand and does not make $K$ broken, it is initially assigned to the nearest facility to $c_K$ (cf. (v) above) and becomes a good demand. As long as $u$ remains a good demand, its actual assignment cost is equal to $d(u, w_K)$. By Lemma 3, the actual assignment cost of the demands in $\mathrm{G}_K$ never exceeds $2\beta f$ plus 3 times their optimal assignment cost. As long as $K$ remains active, its credit can absorb the additional cost of $2\beta f$.

The good inner demands of $K$ are charged with their final assignment cost as soon as they become bad and $\mathrm{G}_K$ becomes empty. By Lemma 3, the total final assignment cost of the demands in $\mathrm{G}_K$ does not exceed $4.5\beta f$ plus 7 times their optimal assignment cost. If $\mathrm{G}_K$

becomes empty because $K$ becomes broken, the additional cost of $4.5\beta f$ is charged to $K$'s credit. Otherwise, $\mathrm{G}_K$ 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 (i), both $w$ and $w'$ are mapped to $K$. Then, by (iii), the facility $w$ must be a supported facility. Hence, the additional cost of $4.5\beta f$ can be charged to the optimal assignment cost of the demands contributing to the opening cost of $w$ (i.e., the demands in $B_w$), since $3x\,\mathrm{Asg}^*(B_w) \geq \beta f$. We also prove that each supported facility is charged with the final assignment cost of some good demands which become bad at most once.

Since we consider at most $2|F^*|-1$ different coalitions and each of them can become isolated at most once, the total assignment cost of the demands in $C_I = \bigcup_{K\in\mathcal{K}} \mathrm{in}_I(K) \cup \mathrm{out}_I(K)$ (i.e., the demands mapped to isolated active coalitions) is at most $14\beta\,\mathrm{Fac}^* + 4(\lambda+2)\mathrm{Asg}^*(C_I) + 14x\mathrm{Asg}^*$.

*Non-isolated Coalitions.* The analysis of the assignment cost of inner demands mapped to non-isolated active coalitions is based on the following lemma.

**Lemma 4.** *For any non-isolated active coalition $K$, the product $|\mathrm{N}_K|\cdot g(c_K)$ can never exceed $(\psi+4)\gamma^2\beta f$.*

*Proof Sketch.* Since $g(c_K)$ is non-increasing, the product $|\mathrm{N}_K|\cdot g(c_K)$ can only increase if a new demand is added to the set of unsatisfied inner demands $\mathrm{N}_K$ (recall that $\mathrm{N}_K = \mathrm{in}_N(K)\cap L$). Let $v$ be the last demand added to $\mathrm{N}_K$. If $d(F_v, c_K) \geq \lambda D_N(K)$, then $B_v = \mathrm{Ball}(v, \frac{d(F_v,v)}{x}) \cap L$ includes any demand in $\mathrm{N}_K$, because $\frac{d(F_v,v)}{x} > 3D_N(K)$ ($\lambda = 3x+2$), while for any $u \in \mathrm{N}_K$, $d(u,v) < 3D_N(K)$, because for any inner demand $u$ of $K$, $d^*_u < D_N(K)$. Since $v$ remains unsatisfied, it must be $\mathrm{Pot}(B_v) < \beta f$, which implies that $|\mathrm{N}_K|(\lambda-2)D_N(K) < \beta f$. Then, the lemma follows from $g(c_K) < (\rho+1)\gamma^2 D_N(K)$, because $K$ is active when $v$ arrives.

On the other hand, if $d(F_v, c_K) < \lambda D_N(K)$, let $w$ be the nearest facility to $c_K$ when $v$ arrives, $d(c_K, w) < \lambda D_N(K)$. We observe that as long as $K$ remains a non-isolated active coalition, it must be $m(w) \geq \psi(\lambda+2)D_N(K)$, and $\mathrm{N}_K \subseteq \mathrm{Ball}(w, \frac{m(w)}{\psi})$. In addition, when $w$ opens, it must be $d(F_w, w) \geq x(\lambda+2)D_N(K)$, because $w$ would have made $K$ either isolated or broken otherwise. Hence, the set of unsatisfied inner demands of $K$ becomes empty when $w$ opens. In addition, for any new facility $w'$ which opens while $w$ is still open, either $d(c_K, w') \geq 2(\lambda+2)D_N(K)$ or $w'$ makes $K$ either isolated or broken. Therefore, any demand currently in $\mathrm{N}_K$ is initially assigned to $w$. Consequently, $\mathrm{N}_K \subseteq \mathrm{Init}(w) \cap \mathrm{Ball}(w, \frac{m(w)}{\psi})$ and the lemma follows from Ineq. (1). □

In addition to Lemma 4, our potential function argument for non-isolated active coalitions exploits the following properties: (i) for any non-isolated active coalition $K$, $g(c_K) \in [\,\rho D_N(K), (\rho+1)\gamma^2 D_N(K)\,)$, and (ii) for each new facility $w'$, if $B_{w'} \cap \mathrm{in}_N(K) \neq \emptyset$, then $g'(c_K) \leq \frac{1}{3}g(c_K)$. As long as $K$ is a non-isolated active coalition, it holds a credit of $(5(\psi+4)\gamma^2+2.5)(\ln(\frac{g(c_K)}{\rho D_N(K)})+1)\beta f$. Each new demand $u$ mapped to the non-isolated active coalition $K$ is charged with its final assignment cost at its assignment time. If $u$ either makes $K$ isolated or broken or opens a new facility $w'$ such that $B_{w'} \cap \mathrm{in}_N(K) \neq \emptyset$, the final assignment cost of $u$ is bounded by $2.5\beta f$ and is charged to $K$'s credit. Otherwise, if $u$ is an outer demand, then $\overline{d}_u \leq 4[(\rho+1)\gamma^2+2]\, d^*_u$, while if $u$ is an inner demand, then $\overline{d}_u \leq 5\, g'_u(c_K)$. In the latter case, $u$ is added to the set of unsatisfied inner demands $\mathrm{N}_K$ and the function $-\Upsilon_K^{(N)} = -5\,|\mathrm{N}_K| \cdot g(c_K)$ compensates for its final assignment cost. By Lemma 4, $\Upsilon_K^{(N)}$ is always bounded by $5(\psi+4)\gamma^2\beta f$. Therefore, every time either $g(c_K)$ decreases or some inner demands are removed from $\mathrm{N}_K$, the decrease in $K$'s credit compensates for the increase in $-\Upsilon_K^{(N)}$.

Consequently, the total assignment cost of the demands in $C_N = \bigcup_{K \in \mathcal{K}} \mathrm{in}_N(K) \cup \mathrm{out}_N(K)$ (i.e., the demands mapped to non-isolated active coalitions) is at most $2\beta\,[5(\psi+4)\gamma^2 + 2.5][\ln((1+\frac{1}{\rho})\gamma^2)+1]\,\mathrm{Fac}^* + 4\,[(\rho+1)\gamma^2+2]\,\mathrm{Asg}^*(C_N)$. By partitioning the interval $[\,\rho D_N(K), (\rho+1)\gamma^2 D_N(K)\,)$ into disjoint sub-intervals $[\,2^i \rho D_N(K), 2^{i+1}\rho D_N(K)\,)$ and considering different phases according to the sub-interval $g(c_K)$ belongs to, we can improve the previous bound to $4\beta\log(\gamma)(12(\psi+2)+2.5)\mathrm{Fac}^* + 8(\rho+1)\mathrm{Asg}^*(C_N)$.

## 3 An Incremental Algorithm for *k*-Median

To obtain an incremental algorithm for $k$-Median, we are 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 5.** *Let* $\mathrm{Asg}^*$ *be the cost of a feasible solution for an instance of $k$-Median, let* $\Lambda$ *be an estimation of* $\mathrm{Asg}^*$*, and let* $\delta = \frac{a_2}{b_2}$*. Then,* IFL *with facility cost* $f = \frac{\Lambda}{\delta k}$ *maintains a solution of cost no greater than* $(a_2+b_2)\,\mathrm{Asg}^* + b_2\,\Lambda$ *which consists of no more than* $(a_1 + a_2\frac{b_1}{b_2}\frac{\mathrm{Asg}^*}{\Lambda})\,k$ *medians.*

The algorithm $\mathrm{IM}(k)$ operates in phases using IFL as a building block. Phase $i$ is characterized by an upper bound $\Lambda_i$ on the optimal assignment cost of the demands considered in the current phase. $\mathrm{IM}(k)$ invokes IFL with facility cost $f_i = \frac{\Lambda_i}{\delta k}$, where $\delta = \frac{a_2}{b_2}$. As long as $\Lambda_i$ is a valid upper bound, IFL maintains a solution consisting of no more than $\nu k$ medians and costing at most $\mu\Lambda_i$, where $\nu, \mu$ are appropriately chosen constants. As soon as either the number of medians exceeds $\nu k$ or the cost exceeds $\mu\Lambda_i$, we can be sure that the optimal cost has also exceeded $\Lambda_i$ (Lemma 5). Then, $\mathrm{IM}(k)$ 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. The algorithm $\mathrm{IM}(k)$ is deterministic and runs in $\mathrm{O}(n^2 k)$ time and $\mathrm{O}(n)$ space. The proof of the following theorem follows from Lemma 5.

**Theorem 2.** *The algorithm* $\mathrm{IM}(k)$ *achieves a constant performance ratio using* $\mathrm{O}(k)$ *medians.*

The randomized algorithm RIM($k$) uses IFL and Gather as building blocks. The algorithm Gather, which is actually a modification of the algorithm PARA_CLUSTER [6], is made up of O($\log n$) independent invocations of Meyerson's randomized algorithm [17] with facility cost $\hat{f}_i = \frac{\Lambda_i}{k(\log n+1)}$. The algorithm RIM($k$) uses Gather to generate a modified instance which can be represented in a space efficient manner. The modified instance contains the same number of different unit demands, which now occupy only $\mathrm{O}(k \log^2 n)$ different locations. Then, RIM($k$) uses IFL to cluster the modified instance. For an incremental implementation, each new demand is first moved to a gathering point by Gather. 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 by Gather as *different unit demands* and may put them in different clusters[4]. In other words, the output of Gather is regarded as a sample taken from the points of the metric space and not as a first-level clustering. The cluster hierarchy maintained by the algorithm entirely consists of the medians produced by IFL.

Since the demands considered by IFL occupy only $\mathrm{O}(k \log^2 n)$ different locations, RIM($k$) can be implemented in $\mathrm{O}(nk^2 \log^2 n)$ time and $\mathrm{O}(k^2 \log^2 n)$ space. Similarly to the analysis of [6], we can prove that the gathering step increases the performance ratio by no more than a constant factor whp. In contrast to IM($k$) which does not require any advance knowledge of $n$, RIM($k$) needs to know $\log n$ in advance.

**Theorem 3.** *The algorithm* RIM($k$) *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.*

## 4 Open Problems

An intriguing open problem is to determine whether there exists a time and space efficient incremental algorithm for $k$-Median which does not assume any advance knowledge of $n$ and achieves a constant performance ratio using $\mathrm{O}(k)$ medians. Another interesting research direction is to improve the constants involved in the performance ratio of IFL. For isolated coalitions, the performance ratio can be significantly improved by a careful analysis. However, it is the analysis of non-isolated coalitions which increases the performance ratio by a large additive term. In addition to a really careful analysis, some new ideas concerning the analysis of non-isolated coalitions are required for establishing a performance ratio of practical interest.

[4] This actually increases the algorithm's time and space complexity by an additional factor of $k$.

## References


1. V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local Search Heuristics for $k$-Median and Facility Location Problems. In *Proc. of the 33rd ACM Symp. on Theory of Computing (STOC '01)*, pages 21–29, 2001.
2. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.
3. M. Charicar, C. Chekuri, T. Feder, and R. Motwani. Incremental Clustering and Dynamic Information Retrieval. In *Proc. of the 29th ACM Symp. on Theory of Computing (STOC '97)*, pages 626–635, 1997.
4. M. Charicar and S. Guha. Improved Combinatorial Algorithms for the Facility Location and $k$-Median Problems. In *Proc. of the 40th IEEE Symp. on Foundations of Computer Science (FOCS '99)*, pages 378–388, 1999.
5. M. Charicar and R. Panigrahy. Clustering to Minimize the Sum of Cluster Diameters. In *Proc. of the 33rd ACM Symp. on Theory of Computing (STOC '01)*, pages 1–10, 2001.
6. M. Charikar, L. O'Callaghan, and R. Panigrahy. Better Streaming Algorithms for Clustering Problems. In *Proc. of the 35th ACM Symp. on Theory of Computing (STOC '03)*, pages 30–39, 2003.
7. D. Fotakis. On the Competitive Ratio for Online Facility Location. In *Proc. of ICALP 03*, volume 2719 of *LNCS*, pages 637–652, 2003.
8. S. Guha. *Approximation Algorithms for Facility Location Problems.* PhD thesis, Stanford University, 2000.
9. S. Guha and S. Khuller. Greedy Strikes Back: Improved Facility Location Algorithms. In *Proc. of the 9th ACM-SIAM Symposium on Discrete Algorithms (SODA '98)*, pages 649–657, 1998.
10. S. Guha, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering Data Streams. In *Proc. of the 41st IEEE Symp. on Foundations of Computer Science (FOCS '00)*, pages 359–366, 2000.
11. M.R. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on Data Streams. Technical Report 1998-011, DEC Systems Research Center, 1998.
12. K. Jain, M. Mahdian, and A. Saberi. A New Greedy Approach for Facility Location Problems. In *Proc. of the 34th ACM Symp. on Theory of Computing (STOC '02)*, pages 731–740, 2002.
13. K. Jain and V. Vazirani. Approximation Algorithms for Metric Facility Location and $k$-Median Problems Using the Primal-Dual Schema and Lagrangian Relaxation. *J. of the Association for Computing Machinery*, 48(2):274–296, 2001.
14. M.R. Korupolu, C.G. Plaxton, and R. Rajaraman. Analysis of a Local Search Heuristic for Facility Location Problems. In *Proc. of the 9th ACM-SIAM Symposium on Discrete Algorithms (SODA '98)*, pages 1–10, 1998.
15. M. Mahdian, Y. Ye, and J. Zhang. Improved Approximation Algorithms for Metric Facility Location Problems. In *Proc. of APPROX 02*, volume 2462 of *LNCS*, pages 229–242, 2002.
16. R.R. Mettu and C.G. Plaxton. The Online Median Problem. In *Proc. of the 41st IEEE Symp. on Foundations of Computer Science (FOCS '00)*, pages 339–348, 2000.
17. A. Meyerson. Online Facility Location. In *Proc. of the 42nd IEEE Symp. on Foundations of Computer Science (FOCS '01)*, pages 426–431, 2001.
18. C.G. Plaxton. Approximation Algorithms for Hierarchical Location Problems. In *Proc. of the 35th ACM Symp. on Theory of Computing (STOC '03)*, pages 40–49, 2003.
19. D. Shmoys. Approximation Algorithms for Facility Location Problems. In *Proc. of APPROX 00*, volume 1913 of *LNCS*, pages 27–33, 2000.
20. D. Shmoys, E. Tardos, and K. Aardal. Approximation Algorithms for Facility Location Problems. In *Proc. of the 29th ACM Symp. on Theory of Computing (STOC '97)*, pages 265–274, 1997.
21. M. Sviridenko. An Improved Approximation Algorithm for the Metric Uncapacitated Facility Location Problem. In *Proc. of IPCO 02*, volume 2337 of *LNCS*, pages 240–257, 2002.