A general approach for incremental approximation and hierarchical clustering

Guolong Lin* Chandrashekhar Nagarajan† Rajmohan Rajaraman‡
David P. Williamson§

April 26, 2010

Abstract

We present a general framework and algorithmic approach for incremental approximation algorithms. The framework handles cardinality constrained minimization problems, such as the k-median and k-MST problems. Given some notion of ordering on solutions of different cardinalities k, we give solutions for all values of k such that the solutions respect the ordering and such that for any k, our solution is close in value to the value of an optimal solution of cardinality k. For instance, for the k-median problem, the notion of ordering is set inclusion and our incremental algorithm produces solutions such that any k and k', $k < k'$, our solution of size k is a subset of our solution of size k'. We show that our framework applies to this incremental version of the k-median problem (introduced by Mettu and Plaxton [36]), and incremental versions of the k-MST problem, k-vertex cover problem, k-set cover problem, as well as the uncapacitated facility location problem (which is not cardinality-constrained). For these problems we either get new incremental algorithms, or improvements over what was previously known. We also show that the framework applies to hierarchical clustering problems. In particular, we give an improved algorithm for a hierarchical version of the k-median problem introduced by Plaxton [37].

1 Introduction

1.1 Incremental problems A company is building facilities in order to supply its customers. Because of limited capital, it can only build a few at this time, but intends to expand in the future in order to improve its customer service. Its plan for expansion is a sequence of facilities that it will build in order as it has funds. Can it plan its future expansion in such a way that if it opens the first k facilities in its sequence, this solution is close in value to that of an optimal solution that opens any choice of k facilities? The company's problem is the *incremental* k-median problem, and was originally proposed by Mettu and Plaxton [36][1]. The standard k-median problem has been the object of intense study in the algorithms community in the past few years. Given the locations of a set of facilities and a set of clients in a metric space, a demand for each client, and a parameter k, the *k-median problem* asks to find a set of k facilities to *open* such that the sum of the demand-weighted distances of the clients to the nearest open facility is minimized. In the incremental k-median problem, we are given the input of the k-median problem without the parameter k and must produce an ordering of the facilities. For each k, consider the ratio of the cost of opening the first k facilities in the ordering to the cost of an optimal k-median solution. The goal of the problem is to find an ordering that minimizes the maximum of this ratio over all values of k. An algorithm for the problem is said to be *α-competitive* if the maximum of the ratio over all k is no more than α. This value α is called the *competitive ratio* of the algorithm. We will also consider randomized algorithms for the incremental k-median problem. For a randomized algorithm, we consider the ratio of the expected cost

*Akamai Technologies, 8 Cambridge Center, Cambridge, MA 02142. Email: glin@akamai.com. Part of this work was done when the author was at Northeastern University.

†Yahoo!, 701 First Ave., Sunnyvale, CA 94089. Email: cn54@yahoo-inc.com.

‡College of Computer and Information Science, Northeastern University, Boston, MA 02115. Email: rraj@ccs.neu.edu. Supported in part by NSF grant CCF-0635119.

§School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY 14853. Email: dpw@cs.cornell.edu. Supported in part by NSF grant CCF-0514628 and an IBM Faculty Partnership Award.

[1]Mettu and Plaxton call it the *online* median problem, but we would like to draw a distinction between incremental and online problems.

of opening the first k facilities in the ordering to the cost of an optimal k-median solution. The algorithm is α-competitive if this ratio is at most α for all k.

In a similar manner, one can also define natural incremental versions of any cardinality constrained minimization problems, including the k-minimum spanning tree problem (k-MST), k-vertex cover, and k-set cover problems. In the k-MST problem, we are given an edge-weighted undirected graph and we wish to find a minimum-weight subtree that covers at least k vertices. In the incremental k-MST problem, we wish to find a sequence of edges such that if we choose the smallest prefix of the edges that connects at least k vertices, the weight of this solution is close in value to that of an optimal k-MST solution. In the standard weighted vertex cover problem, we are given an undirected graph with weights on the vertices and we wish to find a minimum-weight subset of vertices S such that every edge has at least one endpoint in S. In the k-vertex cover problem, we wish to find a minimum-weight set of vertices that covers at least k edges. In the incremental k-vertex cover problem, we wish to find a sequence of vertices, such that if we choose the smallest prefix of vertices in the sequence that covers at least k edges, this solution is close in value to that of the optimal k-vertex cover solution. An incremental version of the facility location problem, which is not cardinality-constrained, has also been defined [37].

Perhaps less obviously, many hierarchical clustering problems can also be cast as incremental problems. In hierarchical clustering, we give clusterings with k clusters for all values of k by starting with a cluster for each point, and repeatedly merging selected pairs of clusters until all points are in a single cluster. Given some objective function on a k-clustering, again we would like to ensure that for any k, the cost of our k-clustering obtained in this way is not too far away from the cost of an optimal k-clustering. The connection with incremental problems is this: for the incremental k-median problem, we insist that for any k, k', with $k < k'$, our solution with k facilities is *ordered* with respect to our solution on k' facilities; namely, the smaller solution is a subset of the larger. In hierarchical clustering, for any k, k', with $k < k'$, our k-clustering must also be ordered with respect to our k'-clustering; namely, the k'-clustering must be a refinement of the k-clustering. We can then consider various clustering criteria: minimize the maximum radius from a cluster center (k-center), minimize the sum of demand-weighted distances of points to their cluster center (k-median), or minimize the sum of demand-weighted distances-squared of points to their cluster center (k-means). From these we obtain hierarchical variants, which we say are α-competitive if for any k, the k-clustering produced by our algorithm is at most α times the cost of an optimal k-clustering under the given objective.

1.2 Our contribution Our central contribution is to give a general approach for solving incremental optimization problems. We then apply this to the incremental versions of the k-median, k-means, k-MST, facility location, k-vertex cover, and k-set cover problems. Furthermore, we apply it to hierarchical clustering problems with the k-median and k-means objective functions. We state our approach in terms of *posets* on solutions to the problems, in which two solutions are comparable in the poset if they obey the ordering imposed by the incremental solution (e.g. if one of the k-median solutions is a subset of the other, or one of the k-vertex cover solutions is a subset of the other, or one of the k-clusterings is a refinement of the other). Each solution in the poset has a cost, as defined by the underlying optimization problem. In addition, we associate a benefit with each solution that models the constraint of the optimization problem (corresponding, for example, to the number of unopened facilities, or the number of edges covered). The goal of the incremental problem is to find a chain of solutions such that for any b, the least element in the chain (according to the partial order) that has benefit at least b has cost close to that of an optimal solution of benefit at least b.

To obtain a competitive solution for a given incremental problem in polynomial-time, our algorithm relies on an α-approximation algorithm for the underlying offline optimization problem. It also relies on an *augmentation* subroutine that, given two solutions of benefits b, b', $b < b'$, which are incomparable in the poset, finds another solution of benefit at least b' that is comparable in the poset to the solution of benefit b. If one can show that this solution has cost no more than a linear combination of the costs of the original two solutions, then one can obtain an $O(\alpha)$-competitive algorithm, where the constant in the big-O depends on the constants in the linear combination. The basic idea of the incremental algorithm is to build a chain of solutions of geometrically increasing cost by repeatedly applying the augmentation subroutine to the current solution in the chain and a solution generated by the α-approximation algorithm that has cost no more than the next bound in the geometrically increasing order. Similar ideas are implicit in the minimum latency approximation algorithm of Blum, Chalsani, Coppersmith, Pulleyblank, Raghavan, and Sudan [6], the incremental facility location algorithm of Plaxton [37] and the hierarchical k-center algorithm of Dasgupta and Long [15]. Choosing a random shift of the geometrically

		Competitive ratio					
		Via optimal		Via approx		Via LMP approx	
Problem	Prev known	Det	Rand	Det	Rand	Det	Rand
Incremental k-median	29.86 [36]	8^*	$2e^*$	$24 + \epsilon^*$	$6e + \epsilon^*$	16	$4e$
Incremental k-MST	8 [6], $2e$ [21]	4	e	8	$2e$		
Incremental k-vertex cover		4	e	8	$2e$		
Incremental k-set cover		4	e	$O(\log n)$	$O(\log n)$		
Incremental facility location	$6 + \epsilon$ [37]	4	e	6	$1.5e$		
Hierarchical k-median	238.88 [37]	20.71	10.03	$62.13 + \epsilon$	$30.09 + \epsilon$	41.42	20.06

Table 1: Our summary of results. The first column gives the best previously known competitive ratio for a polynomial-time algorithm. The second and third state the competitive ratio for incremental solutions obtained using optimal algorithms for the benefit-constrained problems and are thus non-polynomial-time algorithms; they should be viewed as existential results. The fourth and fifth state the competitive ratio for our polynomial-time algorithms via an α-approximation algorithm. In the entries for the incremental k-set cover problem, n refers to the number of elements in the instance. The sixth and seventh give the competitive ratio for our polynomial-time algorithms via a Lagrangean multiplier preserving ρ-approximation algorithm. The results with * were independently obtained by Chrobak, Kenyon, Noga, and Young [12].

increasing sequence as in Goemans and Kleinberg [21] and Dasgupta and Long [15] gives improved randomized algorithms.

In some cases, we are able to improve the competitive ratio still further. In particular, if there exists a Lagrangean multiplier preserving ρ-approximation algorithm for the problem in which Lagrangean relaxation has been applied to the benefit constraint, we are able to give the same result as above in which this algorithm replaces the α-approximation algorithm. This yields improved competitive ratios in the cases where we have such algorithms with performance guarantees ρ better than the best known performance guarantee α for the problem with the benefit constraint. In particular, there is a Lagrangean multiplier preserving 2-approximation algorithm for the facility location problem (due to Jain, Mahdian, Markakis, Saberi, and Vazirani [31]), which we can use in place of a $(3+\epsilon)$-approximation algorithm for the k-median problem (due to Arya, Garg, Khandekar, Meyerson, Munagala, and Pandit [3]), yielding improvements in the competitive ratios for the incremental k-median problem, hierarchical k-median problem, and hierarchical k-means problem.

We summarize our main results in Table 1.

1.3 Related work All of the optimization problems studied in this paper are NP-hard and have been extensively studied with respect to their approximability. For instance, hardness results for k-median, k-MST, and facility location can be found in [34], [38], and [23], respectively; hardness of k-vertex cover and k-set cover follow directly from those of vertex cover [27] and set cover [17], respectively. Several approximation algorithms are known for k-median, k-MST, facility location, vertex cover, and set cover problems (see, for example the books [29, 41]).

The k-median problem and the related uncapacitated facility location problems have been the objects of intense study in the algorithms community in the past few years. The currently best known approximation algorithms for these problems have performance guarantees of $3 + \epsilon$ (due to Arya et al. [3]) and 1.5 (due to Byrka [9]) respectively. Also of interest to us is the best known Lagrangean multiplier preserving approximation algorithm for the facility location problem with performance guarantee of 2, which is due to Jain et al. [31]. The k-MST problem was originally motivated by the network design application of finding the cheapest network (e.g., a cable network or an oil pipeline) that spans a given number of nodes [38]. The problem also arises in certain solutions to the well-studied minimum latency problem [2]. The best currently known approximation algorithm for the k-MST problem has a performance guarantee of 2 (Garg [20]).

The k-vertex cover and k-set cover problems are motivated by applications where solutions satisfying *all* covering constraints may be too expensive, and partial covers are acceptable [5, 4, 19, 1]. For instance, suppose a company does not have enough resources to reach all potential clients, and hence aims to reach a targetted fraction of the clients. A natural problem is to determine the smallest set of facility locations achieving this aim, given the set of clients that each location can cover. This is an instance of the k-set cover problem. Similarly, clustering applications where a small number of outlier data points can be ignored can also be modeled as partial covering problems. Our incremental k-vertex cover algorithm relies on a 2-approximation algorithm for k-vertex cover, while our incremental k-set cover algorithm relies on an $O(\log n)$-approximation algorithm for k-set cover [8, 19, 28, 35, 39].

There has been a lot of previous work on incremental approximation algorithms, but it was usually done on a problem-by-problem basis. Mettu and Plaxton [36] introduce the incremental k-median problem, and give a 29.86-competitive algorithm for it. Their algorithm runs in near linear time and their argument also applies when the distances satisfy a weaker version of triangle inequality, yielding an $O(1)$-competitive solution for the incremental k-means problem. Plaxton [37] introduces the incremental facility location problem and gives a $(4+\epsilon)\alpha$-competitive algorithm for it, given any α-approximation algorithm for the uncapacitated facility location problem. This yields a $(6+\epsilon)$-competitive algorithm for incremental facility location using the 1.5-approximation algorithm of [9] for uncapacitated facility location. González [22] gives a 2-approximation algorithm for the k-center problem, which is also a 2-competitive algorithm for the incremental k-center problem. Implicit in the work of Charikar, Chekuri, Feder, and Motwani [10] on incremental clustering are a deterministic 8-competitive algorithm and a randomized $2e$-competitive algorithm for the hierarchical k-center problem. Dasgupta and Long [15] explicitly introduce the idea of finding competitive hierarchical clusterings, and derive the same bounds as above for the hierarchical k-center problem. Plaxton [37] gives an 8α-competitive algorithm for the hierarchical k-median problem, given an α-competitive algorithm for the incremental k-median problem. Using the algorithm of Mettu and Plaxton [36] gives a 238.88-competitive algorithm for the hierarchical k-median problem. The work of [37] also includes an $O(1)$-competitive algorithm for the hierarchical k-means problem. Implicit in work on the minimum latency problem is a number of different algorithms for the incremental k-MST problem; given an α-approximation algorithm for the k-MST problem, the work of Blum et al. [6] yields a 4α-competitive algorithm, while a randomized $e\alpha$-competitive algorithm is implicit in Goemans and Kleinberg [21]. Other work on incremental approximation algorithms includes incremental flow (Hartline and Sharp [26]) and incremental bin packing (Codenotti, De Marco, Leoncini, Mantangero, Santini [14]).

Independently Chrobak, Kenyon, Noga, and Young [12] also discovered the same $(24+\epsilon)$-competitive deterministic and $(6e+\epsilon)$-competitive randomized algorithms as ours for the incremental k-median problem. They also consider the incremental version of a median problem in which the goal is to minimize the number of medians required to satisfy a given cost constraint. These results are derived by a reduction from a new problem, which they call online bribery, for which tight upper and lower bounds are established. Chrobak et al. [12] also extend their work to fractional k-medians, approximately metric distance functions, which include the k-means objective, and bicriteria approximations.

As a paradigm for dealing with uncertainty, incremental approximations are most closely related to online algorithms, stochastic optimization, and universal approximations. The study of online algorithms considers problems in which the input is revealed over time, and the algorithm must make decisions without knowledge of future inputs [7, 18, 40]. The study of stochastic optimization (e.g., see [16, 24, 25, 30]) considers problems in which the cost of future decisions may be significantly different than those now: the future is unknown, but is chosen randomly from one of a number of different possible scenarios which are known (possibly given as a black box). In contrast, an incremental algorithm for a problem performs all of the computation offline and outputs a single chain of solutions such that for every possible benefit constraint there is a valid solution that is close to the optimal. Our measure of performance for incremental solutions is modeled on the measure of competitive ratio from online algorithms. The notion of universal approximations studied in [33] considers a much stronger notion of uncertainty in the sense that the number of possibilities for the unknown portion of the input is exponential in the size of the problem. As a result, the competitive ratios achievable within the incremental framework are much smaller than the approximations achievable within the universal framework.

2 A general framework for incremental optimization

In this section, we present a general framework for incremental optimization (Section 2.1), a generic approximation algorithm for incremental optimization problems that lie within this framework (Section 2.2) and an alternate view of the generic algorithm useful for getting a better competitive ratio (Section 2.3).

2.1 Problem definitions The problems we consider in this paper are all minimization problems and share the following characteristics. Each optimization problem Π can be specified by a quadruple $\langle U, \mathsf{ben}, \mathsf{cost}, p\rangle$, where U is a set of feasible solutions, $\mathsf{ben} : U \to \mathcal{R}$ and $\mathsf{cost} : U \to \mathcal{R}$ are *benefit* and *cost* functions, respectively, and the goal is to seek a solution S that minimizes $\mathsf{cost}(S)$ subject to the condition $\mathsf{ben}(S) \geq p$. We refer to Π as an *offline problem* to distinguish it from its incremental version, which we now define. We introduce a binary relation $\preceq$, which induces a partial order on U, i.e., $\langle U, \preceq\rangle$ is a *poset.* Throughout this paper, we focus on benefit and cost functions that are nonnegative and monotonically non-decreasing with respect to the partial order; that is, if $S \preceq S'$, then $\mathsf{ben}(S) \leq \mathsf{ben}(S')$ and $\mathsf{cost}(S) \leq \mathsf{cost}(S')$. We also assume that the empty set $\emptyset$ is a feasible solution with cost 0; that is, $\emptyset \in U$ and $\mathsf{cost}(\emptyset) = 0$. (The element $\emptyset$ is also a bottom element of $\preceq$ in all the problem formulations considered in this paper.)

We illustrate the above notation by considering the k-MST and k-median problems. For the k-MST problem, we let U be the set of all connected subgraphs of G that contain r, $\preceq$ be the $\subseteq$ relation over the edge subsets, the benefit of a solution be the number of vertices it spans, and the cost of a solution be the sum of weights of its edges. For the k-median problem, we let $U = 2^F$ be the set of all feasible solutions, each solution represented by the set of open facilities, $\preceq$ be the $\supseteq$ relation over the facility sets, $\mathsf{ben}(S)$ for a solution S be $|F| - |S|$, and $\mathsf{cost}(S)$ for a solution S be the cost of connecting the clients to their nearest facilities in S.

The incremental version of Π is specified by the quadruple $\langle U, \preceq, \mathsf{ben}, \mathsf{cost}\rangle$ and seeks a *chain* $\mathcal{C}$ of $\langle U, \preceq\rangle$; that is, $\mathcal{C}$ is a sequence of solutions $S_1, \ldots, S_m$ for some integer m, such that $S_i \in U$ for $1 \leq i \leq m$ and $S_i \preceq S_{i+1}$ for $1 \leq i < m$. Define the competitive ratio of $\mathcal{C}$ as

$$\sup_{0<p\leq B_{max}} \frac{\mathsf{cost}(\pi(\mathcal{C}, p))}{\mathsf{cost}(\mathsf{Opt}(p))},$$

where $B_{max} = \max_{S\in U} \mathsf{ben}(S)$ is the maximum benefit achieved by a feasible solution, $\pi(\mathcal{C}, p)$ denotes the smallest indexed element of $\mathcal{C}$ whose benefit is at least p, and $\mathsf{cost}(\mathsf{Opt}(p))$ is the cost of an optimal solution for the offline problem for benefit p, namely $\langle U, \mathsf{ben}, \mathsf{cost}, p\rangle$. In the incremental version of the k-MST problem, the desired output is a chain of subgraphs of the input graph G, where each chain element is a subgraph of its successor. In the incremental version of the k-median problem, the desired output is a chain of subsets of the facilities, where each chain element (subset of facilities) is a subset of the previous element[2].

DEFINITION 2.1. *An α-approximation algorithm for the problem Π finds a solution S in U for every given benefit p such that $\mathsf{cost}(S) \leq \alpha \cdot \mathsf{cost}(\mathsf{Opt}(p))$ and $\mathsf{ben}(S) \geq p$. Usually such problems have the set of integers from 1 to B_{max} as the range of benefit function.*

2.2 A generic incremental approximation algorithm The core of each of our approximation algorithms for incremental optimization problems is a subroutine for augmenting a given solution to achieve a certain benefit. In this section, we present a sufficient condition for the existence of such an augmentation. By repeatedly invoking this augmentation subroutine (which is specific to the particular problem), we show how to derive a sequence that has a good competitive ratio. We begin by defining the augmentation property.

DEFINITION 2.2. *(γ, δ)-Augmentation: We say that the (γ, δ)-augmentation property holds for reals $\gamma, \delta \geq 0$ if for every solution S of U and every real $p \leq B_{max}$, there exists an augmentation S' such that*

1. $S \preceq S'$.
2. $\mathsf{cost}(S') \leq \gamma\mathsf{cost}(S) + \delta\mathsf{cost}(\mathsf{Opt}(p))$.

[2] We note that an incremental solution to the k-MST problem is a sequence of graphs, each obtained by *adding* edges to the preceding subgraph, while an incremental solution to the k-median problem is a sequence of facility sets, each obtained by *deleting* facilities from the preceding solution. These diverging definitions are a direct consequence of the different roles the parameter k plays in these problems: k is a lower bound on the number of vertices spanned in the k-MST problem, whereas k is an upper bound on the number of facilities opened in the k-median problem. Thus, in our cost-benefit framework, the benefit of a subgraph in the k-MST problem is the number of vertices spanned, whereas the benefit of a facility location solution S in the k-median problem is $|F| - |S|$. It may have been more natural to define an incremental solution to the k-median problem to be a sequence of facility sets, each obtained by adding facilities to the preceding solution. And indeed, one can redefine our framework to achieve this; however, such a reformulation would result in less natural definitions for the incremental versions of the other problems we are studying here, including k-MST, k-vertex cover, and k-set cover.

3. $\mathsf{ben}(S') \geq p$.

Let $\mathsf{Augment}(S, p, \gamma, \delta)$ *denote a subroutine that computes such an augmentation. For efficiency reasons, we also introduce a companion subroutine* $\mathsf{CostBound}(S, p, \gamma, \delta)$ *which returns a bound on the cost of* $\mathsf{Augment}(S, p, \gamma, \delta)$*. In particular, for every feasible solution* S *and benefit* p*, we have*

$$\mathsf{cost}(\mathsf{Augment}(S, p, \gamma, \delta)) \leq \mathsf{CostBound}(S, p, \gamma, \delta) \leq \gamma \mathsf{cost}(S) + \delta \mathsf{cost}(\mathsf{Opt}(p)).$$

We now present two generic incremental optimization algorithms, given an augmentation subroutine. One is deterministic, while the other is randomized. Since these two algorithms share the same structure, differing only in the parameter setting (the Initialization step below), they are shown together. In the subsequent sections, we show that for each of the problems we consider in this paper, the augmentation subroutine can be implemented using an approximation algorithm to the offline optimization problem for suitable choices of γ and δ.

Algorithm 1 INCAPPROX(γ, δ)

---

1. Initialization:

   $S_0 = \mathsf{Augment}(\emptyset, q, \gamma, \delta)$, where q is the largest value for which $\mathsf{CostBound}(\emptyset, q, \gamma, \delta) = 0$. $C_0 = \max\{1, \mathsf{cost}(S_0)\}$.

   1D: (Deterministic) $i = 0$, $\beta = 2\gamma$, $\beta_0 = \beta$.

   1R: (Randomized) $i = 0$, β is the minimizer of $\frac{\beta - 1}{(1-\gamma/\beta)\ln\beta}$, $\beta_0 = \beta^X$, where X is uniform from $[0, 1)$.

2. Iteration i: $S_{i+1} = \mathsf{Augment}(S_i, p, \gamma, \delta)$, where p is the largest value for which $\mathsf{CostBound}(S_i, p, \gamma, \delta)$ is at most $\beta_0 \beta^i C_0$.
3. Termination: If $\mathsf{ben}(S_{i+1}) \neq B_{max}$, $i \leftarrow i + 1$, go to step 2; Otherwise, return sequence $S_1, \cdots, S_{i+1}$.

REMARK 2.1. *For some applications discussed in this paper, most notably the incremental and hierarchical median problems, the poset induced by the partial order is, in fact, a ranked poset; that is, every maximal chain in the poset is of the same length*[3]*. For these problems, we can replace the chain* $\mathcal{C}$ *that is output by the above incremental algorithm by any maximal chain that contains* $\mathcal{C}$*, without increasing the competitive ratio.*

THEOREM 2.1. *Assume that for any* $S \in U$, $\mathsf{cost}(S)$ *is either zero or bounded away from zero; that is,* $\inf\{\mathsf{cost}(S) : S \in U, \mathsf{cost}(S) > 0\} > 0$. *If* (γ, δ)*-Augmentation holds for reals* $\gamma \geq 1$, $\delta \geq 1$, *then (i)* INCAPPROX(γ, δ) *(Deterministic) computes an incremental solution with competitive ratio* $4\gamma\delta$*; (ii)* INCAPPROX(γ, δ) *(Randomized) computes an incremental solution with competitive ratio* $\min_\mu \frac{\delta(\mu-1)}{(1-\gamma/\mu)\ln\mu}$*, which equals* $e\delta$*, when* $\gamma = 1$.

Proof. By scaling costs, we can assume without loss of generality that the cost of every solution in U is either zero or at least one. Fix a real $p \leq B_{max}$. Let S^* denote an optimal solution for the offline instance with benefit p. We consider two cases.

If $\mathsf{cost}(S^*) = 0$ then $p \leq \mathsf{ben}(S_0)$ by the maximality of $\mathsf{ben}(S_0)$ and the fact that $\mathsf{CostBound}(\emptyset, p, \gamma, \delta) = 0$. In this case we have found a solution S_0 such that $\mathsf{cost}(S^*) = \mathsf{cost}(S_0) = 0$ and $\mathsf{ben}(S^*) \leq \mathsf{ben}(S_0)$, and the claim of the theorem holds.

The remainder of the proof concerns the case where $\mathsf{cost}(S^*) \neq 0$. We then have $\mathsf{cost}(S^*) \geq 1$. Since either $\mathsf{cost}(S_0) = 0$ or $S_0 = \emptyset$, we also have $\mathsf{cost}(S^*) \geq \mathsf{cost}(S_0)$. Therefore $\mathsf{cost}(S^*) \geq C_0$. Let k be the smallest integer such that $\delta\mathsf{cost}(S^*)/(1 - \gamma/\beta) \leq \beta_0\beta^k C_0$. We note that $k \geq 0$ since

$$\frac{\delta\mathsf{cost}(S^*)}{1 - \gamma/\beta} > \mathsf{cost}(S^*) \geq \beta_0 C_0/\beta.$$

We now argue that $\mathsf{cost}(\mathsf{Augment}(S_k, p, \gamma, \delta))$ is at most $\beta_0\beta^k C_0$. We consider two cases: $k = 0$ and $k > 0$. If $k = 0$, then we obtain

$$\begin{aligned}
\mathsf{cost}(\mathsf{Augment}(S_0, p, \gamma, \delta)) &\leq \gamma\mathsf{cost}(S_0) + \delta\mathsf{cost}(S^*) \\
&\leq \gamma C_0 + \beta_0(1 - \gamma/\beta)C_0 \\
&\leq C_0\beta_0.
\end{aligned}$$

[3]In poset $(U, \preceq)$, chain $\mathcal{C} = S_1, \ldots, S_m$ is maximal if there exists no $S \in U$ such that either $S_1 \preceq S$ or $S_m \in S$.

For $k > 0$, we use the following property that is enforced by each iteration of IncApprox: for $j \geq 1$, $\mathsf{cost}(S_j) \leq \beta_0\beta^{j-1}C_0$. Applying the preceding inequality with $j = k$, we obtain $\mathsf{cost}(S_k) \leq \beta_0\beta^{k-1}C_0$. We now derive

$$\begin{aligned}\mathsf{cost}(\mathsf{Augment}(S_k, p, \gamma, \delta)) &\leq \gamma\mathsf{cost}(S_k) + \delta\mathsf{cost}(S^*)\\ &\leq \gamma\beta_0\beta^{k-1}C_0 + \beta_0\beta^k C_0 \cdot (1-\gamma/\beta)\\ &\leq C_0\beta_0\beta^k.\end{aligned}$$

We now establish a bound on the competitive ratios of the two versions of the algorithm using the solution S_{k+1}, which has benefit at least p and cost at most $\beta_0\beta^k C_0$.

Deterministic: By the minimality of k, we lower bound $\mathsf{cost}(S^*)$ by $\beta_0\beta^{k-1}C_0(1-\gamma/\beta)/\delta$ and obtain the following upper bound on the competitive ratio of $\mathcal{C}$.

$$\begin{aligned}\mathsf{cost}(\pi(\mathcal{C}, p))/\mathsf{cost}(\mathsf{Opt}(p)) &\leq \mathsf{cost}(S_{k+1})/\mathsf{cost}(\mathsf{Opt}(p))\\ &\leq \frac{\beta_0\beta^k}{\beta_0\beta^{k-1}\cdot\frac{1-\gamma/\beta}{\delta}} = \beta^2\delta/(\beta-\gamma).\end{aligned}$$

The above bound is minimized when $\beta = 2\gamma$, thus yielding a $4\delta\gamma$ competitive ratio.

Randomized: In the randomized algorithm, β_0 is a random variable β^X, where X is uniform in $[0, 1)$. Let Y denote the random variable $\log_\beta(\beta_0\beta^k C_0 \frac{1-\gamma/\beta}{\delta\mathsf{cost}(S^*)})$. We now argue that Y is uniform in $[0, 1)$. Letting C equal $\log_\beta(\frac{\delta\mathsf{cost}(S^*)}{C_0(1-\gamma/\beta)})$, we obtain $Y = k - (C - X)$. On the other hand, it follows from the definition of k that $k = \lceil C - X\rceil$. Since X is chosen uniformly at random in $[0, 1)$, so is Y.

Thus, the expectation of the ratio $\beta_0\beta^k\frac{(1-\gamma/\beta)C_0}{\delta\mathsf{cost}(S^*)}$ is $\int_0^1 \beta^y dy = \frac{\beta-1}{\ln\beta}$. We conclude that the competitive ratio is at most

$$\begin{aligned}E\left[\frac{\beta_0\beta^k C_0}{\mathsf{cost}(S^*)}\right] &= E\left[\frac{\beta_0\beta^k}{\frac{\delta\mathsf{cost}(S^*)}{C_0(1-\gamma/\beta)}}\right]\cdot\frac{\delta}{1-\gamma/\beta}\\ &= \frac{\delta(\beta-1)}{(1-\gamma/\beta)\ln\beta}.\end{aligned}$$

We select β to minimize the above bound. In particular, with $\gamma = 1$, we set $\beta = e$, obtaining a ratio of $e\delta$. □

Remark 2.2. *The assumption about the cost of any feasible solution in the statement of Theorem 2.1 trivially holds when U is finite. Indeed, the assumption applies to all problems studied in this paper, with the exception of a special case of the facility location problem, which we handle separately in Section 3.3.*

We now derive a polynomial upper bound on the running time of IncApprox under the assumption that the range of the benefit function is the set of integers, which is true for all problems studied in this paper except the incremental facility location problem. We establish the polynomial running time of IncApprox for incremental facility location separately in Section 3.3.

The running time of IncApprox is dominated by the calls to the augmentation subroutine. The number of calls made to the augmentation subroutine in each iteration (including the initialization step) depends on the particular search technique we use to find a maximum benefit augmented solution within a certain cost bound. One simple method is to let the CostBound function simply return the cost of the augmented solution (i.e., $\mathsf{CostBound}(S, p, \gamma, \delta) = \mathsf{cost}(\mathsf{Augment}(S, p, \gamma, \delta))$), and to perform a linear search through all benefit values. Then the number of calls made during an iteration is at most B_{max}. Since the number of iterations is at most $\log_\beta(\mathsf{Maxcost}/\beta)$, we obtain an upper bound of $O(B_{max}\log_\beta(\mathsf{Maxcost}/\beta))$ calls to the augmentation subroutine. For all the problems with integer benefits that we consider in this paper, B_{max} is at most the size of the problem instance, and hence the running time of IncApprox is polynomial in the size of the instance.

We can give a more efficient implementation by replacing the above linear search by binary search if, for all S, $\mathsf{CostBound}(S, p, \gamma, \delta)$ is a mononotically nondecreasing function of p. (We can define such a CostBound subroutine for all the problems we study in this paper.) Given this monotonicity property, we can perform a binary search on benefit values to find the maximum benefit augmented solution that has cost at most $\beta_0\beta^i$, according to CostBound. The number of calls made in any iteration is then $O(\log B_{max})$, yielding an upper bound of $O(\log_\beta(\mathsf{Maxcost}/\beta)\log B_{max})$ total calls to the augmentation subroutine.

2.3 An alternate view of the generic incremental approximation algorithm This alternate view of the algorithm offers better understanding of the algorithm when the range of the benefit function is the set of positive integers. In the above algorithm, we use the augmentation subroutine as a black box. For the incremental problems discussed in this paper, when we implement this subroutine, we almost always take two elements S_1, S_2 with $\mathsf{ben}(S_1) < \mathsf{ben}(S_2)$, where S_2 is usually a good approximate solution to $\mathsf{Opt}(\mathsf{ben}(S_2))$, and create another element S such that $S_1 \preceq S$ and $\mathsf{ben}(S) \geq \mathsf{ben}(S_2)$. With this implementation detail in mind, a high-level and equivalent view of the above algorithm is the following: Identify a suitable set of benefit values such that the corresponding (good) approximate solutions' costs lie in different *buckets*, where the bucket size increases geometrically. One then constructs a chain out of these solutions iteratively using the augmentation subroutine. This alternate approach is explained in more detail in this section as it helps us give better approximation ratios for some problems.

DEFINITION 2.3. (γ, δ')*-Nesting: We say that the* (γ, δ')*-nesting property holds for reals* $\gamma, \delta' \geq 0$ *if for any two solutions* S_1 *and* S_2 *of* U *with* $\mathsf{ben}(S_1) < \mathsf{ben}(S_2)$*, there exists a solution* S *such that*

1. $S_1 \preceq S$.

2. $\mathsf{cost}(S) \leq \gamma \mathsf{cost}(S_1) + \delta' \mathsf{cost}(S_2)$.

3. $\mathsf{ben}(S) \geq \mathsf{ben}(S_2)$.

Let $\mathsf{Nesting}(S_1, S_2, \gamma, \delta')$ *denote a subroutine that computes such a solution* S.

REMARK 2.3. *Note that the* Augment *and the* Nesting *subroutine are equivalent. The only minor difference is that the* Augument *subroutine takes a solution and a particular benefit as arguments to find a solution with benefit no less than given benefit whereas the* Nesting *subroutine takes two solutions as arguments and finds a solution with benefit no less than the benefit of the second solution given.*

Algorithm 2 ALTINCAPPROX$(\gamma, \delta', \alpha)$

---

1. Initialization:

   1D: (Deterministic) $i = 1$, $S_0 = \emptyset$, $\beta = 2\gamma$, $\beta_0 = 1$.
   1R: (Randomized) $i = 1$, $S_0 = \emptyset$, β is the minimizer of $\frac{\beta - 1}{(1-\gamma/\beta)\ln\beta}$, $\beta_0 = \beta^X$, where X is uniform from $[0, 1)$.

2. Use an α-approximation algorithm to compute approximate solutions $V_1, V_2, \ldots V_{B_{max}}$ for benefit values $1, 2, \ldots, B_{max}$ respectively.
3. Bucketing: Order these solutions according to their cost into buckets of form $[0, 0]$, $(\beta_0, \beta_0\beta]$, $(\beta_0\beta, \beta_0\beta^2]$, $\ldots$, $(\beta_0\beta^{k-1}, \beta_0\beta^k], \ldots$
4. Pick the solution with highest benefit from each of these non-empty buckets. Let these solutions be $\overline{V}_1, \overline{V}_2, \ldots, \overline{V}_r = V_{B_{max}}$ respectively.
5. Iteration i: $S_i = \mathsf{Nesting}(S_{i-1}, \overline{V}_i, \gamma, \delta')$,
6. Termination: If $\mathsf{ben}(S_i) \neq B_{max}$, $i \leftarrow i + 1$, go to step 5; Otherwise, return sequence $S_1, \cdots, S_i$.

THEOREM 2.2. *If* (γ, δ')*-nesting holds for reals* $\gamma \geq 1$, $\delta' > 0$*, and an* α*-approximation algorithm exists for the problem, then (i)* ALTINCAPPROX$(\gamma, \delta', \alpha)$ *(Deterministic) computes an incremental solution with competitive ratio* $4\gamma\delta'\alpha$*;* ALTINCAPPROX$(\gamma, \delta', \alpha)$ *(Randomized) computes an incremental solution with competitive ratio* $\min_\beta \frac{\delta'\alpha(\beta-1)}{(1-\gamma/\beta)\ln\beta}$*, which equals* $e\delta'\alpha$*, when* $\gamma = 1$.

Proof. Fix a $p \leq B_{max}$. If $p \leq \mathsf{ben}(S_0)$, then the solution $S_0 = \emptyset$ provides benefit at least p with least cost, which implies $\mathsf{cost}(\pi(\mathcal{C}, p))$ equals $\mathsf{cost}(\mathsf{Opt}(p))$, establishing the desired claim for this case.

In the remainder, we assume that $p > \mathsf{ben}(S_0)$. Let $i \geq 1$ be such that $\mathsf{ben}(S_{i-1}) < p \leq \mathsf{ben}(S_i)$. Let the solution $\overline{V}_i$ be from the bucket $(M/\beta, M]$. So $\mathsf{cost}(\overline{V}_i) \leq M$, $\mathsf{cost}(\overline{V}_{i-1}) \leq M/\beta$ and so on.

Deterministic case:

$$\begin{aligned}
\mathsf{cost}(\pi(\mathcal{C},p)) &= \mathsf{cost}(S_i) \\
&\leq \delta'\mathsf{cost}(\overline{V}_i) + \gamma\mathsf{cost}(S_{i-1}) \\
&\leq \delta'\mathsf{cost}(\overline{V}_i) + \gamma\delta'\mathsf{cost}(\overline{V}_{i-1}) + \gamma^2\delta'\mathsf{cost}(S_{i-2}) \\
&\leq \delta'\mathsf{cost}(\overline{V}_i) + \gamma\delta'\mathsf{cost}(\overline{V}_{i-1}) + \gamma^2\delta'\mathsf{cost}(\overline{V}_{i-2}) + \dots \\
&\leq \delta' M\left(1 + \frac{\gamma}{\beta} + \frac{\gamma^2}{\beta^2} + \dots\right) \\
&\leq \frac{\delta' M}{1-\frac{\gamma}{\beta}} \qquad (2.1) \\
&\leq \frac{\delta'\beta}{1-\frac{\gamma}{\beta}} \cdot \mathsf{cost}(V_p) \\
&\leq \frac{\delta'\beta\alpha}{1-\frac{\gamma}{\beta}} \cdot \mathsf{cost}(\mathsf{Opt}(p))
\end{aligned}$$

The second inequality follows from (γ,δ')-Nesting inequality $\mathsf{cost}(S_{i-1}) \leq \delta'\mathsf{cost}(\overline{V}_{i-1}) + \gamma\mathsf{cost}(S_{i-2})$ and the penultimate equation follows from the fact that $\mathsf{cost}(V_p)$ lies between M/β and M and so $M \leq \beta\mathsf{cost}(V_p)$. This ratio is minimized when $\beta = 2\gamma$ which gives a $4\gamma\delta'\alpha$-competitive algorithm.

Randomized case: Since β_0 is a random variable β^X where X is uniform in $[0,1)$, it follows that $M/\mathsf{cost}(V_p)$ is a random variable β^Y, where Y is uniform $[0,1)$. From Equation 2.1,

$$\begin{aligned}
E\left(\mathsf{cost}(\pi(\mathcal{C},p))\right) &\leq E\left(\frac{\delta' M}{1-\frac{\gamma}{\beta}}\right) \\
&\leq \left(\frac{\delta'\mathsf{cost}(V_p)}{1-\frac{\gamma}{\beta}}\right) \cdot E\left(\frac{M}{\mathsf{cost}(V_p)}\right) \\
&\leq \left(\frac{\delta'\mathsf{cost}(V_p)}{1-\frac{\gamma}{\beta}}\right) E(\beta^Y) \\
&\leq \left(\frac{\delta'\mathsf{cost}(V_p)}{1-\frac{\gamma}{\beta}}\right) \cdot \frac{\beta-1}{\ln\beta} \\
&\leq \left(\frac{\delta'(\beta-1)\alpha}{\left(1-\frac{\gamma}{\beta}\right)\ln\beta}\right) \cdot \mathsf{cost}(\mathsf{Opt}(p))
\end{aligned}$$

When $\gamma = 1$, the ratio is minimized at $\beta = e$ which gives an $e\delta'\alpha$-competitive algorithm. □

The running time of AltIncApprox is dominated by the calls to the approximation algorithm which is done B_{max} times. Then we call the nesting subroutine r times which is bounded by B_{max}. So the algorithm requires B_{max} calls to the approximation algorithm and at most B_{max} calls to the nesting subroutine.

3 Applications

In this section, we apply our framework of Section 2 to incremental versions of several classical optimization problems.

3.1 The incremental k-MST problem Given a complete graph $G = (V,E)$, $|V| = n$, with metric cost function $w : E \to Q^+$ and a root $r \in V$, the (rooted) *k-MST problem* seeks a minimum-cost subgraph of G that spans at least k vertices, including r. In the incremental k-MST problem, we seek a sequence of $n-1$ edges of E, $e_1, e_2, \dots, e_{n-1}$ such that for any $k \in [2,n]$, the first $k-1$ edges of the sequence span k vertices including r. For each k, consider the ratio of the sum of the cost of the first $k-1$ edges to the cost of an optimal k-MST of

G that covers r. The goal of incremental k-MST is to seek a sequence of edges that minimizes the maximum of this ratio, over all k.

In our framework, U is the set of all connected subgraphs of G that contain r and $\preceq$ is the $\subseteq$ relation of the edge subsets. The benefit of a solution is the number of vertices it spans, and the cost is the sum of the edge weights.

Lemma 3.1. *There exists a $(1,1)$-augmentation for the k-MST problem, and a $(1,\alpha)$-augmentation that can be implemented in polynomial time, where $\alpha = 2$. Also a $(1,1)$-nesting exists and can be efficiently implemented for the k-MST problem.*

Proof. The augmentation operation $\mathsf{Augment}$ is straightforward. Given any component $S \subseteq E$ spanning r, and $k \leq n$, if it already contains at least k vertices, we are done. Otherwise, let S^* be an optimum solution to the rooted k-MST problem. We can take the edges of S and a subset of edges of S^* to connect the vertices in S^* to those of S. This is possible since both the S and S^* contains the root vertex. This yields a component S' spanning at least $k = |S^*|$ vertices with cost bounded by $\mathsf{cost}(S) + \mathsf{cost}(S^*)$. The companion $\mathsf{CostBound}$ function returns $\mathsf{cost}(S) + \mathsf{cost}(S^*)$.

Similarly, in order to implement the augmentation efficiently, we use the polynomial-time 2-approximation algorithm [20] to obtain a solution S_1 to the k-MST problem, and connect the newly discovered vertices to S. This yields a component S' spanning at least k vertices with cost at most $\mathsf{cost}(S) + \mathsf{cost}(S_1)$ which is at most $\mathsf{cost}(S) + 2\mathsf{cost}(S^*)$. To obtain a monotonic cost bound function, we have $\mathsf{CostBound}$ return $\mathsf{cost}(S) + \mathsf{cost}(S_1)$. Since the cost of the k-MST solution returned by the algorithm of [20] is monotonically increasing with k, $\mathsf{CostBound}$ is mononotic with respect to the benefit parameter.

The existence of $(1,1)$-Nesting and efficient implementation follows by taking S and S^* to be S_1 and S_2 which are the inputs to the nesting subroutine. □

Theorem 3.1. *There exists a solution to incremental k-MST problem with competitive ratio 4. A deterministic solution with competitive ratio 4α and a randomized solution with competitive ratio $e\alpha$ can be computed efficiently, where $\alpha = 2$.*

Proof. Immediate from Lemma 3.1 and Theorem 2.1. □

We note that this computation of an 8-competitive incremental MST is implicit in the work of Blum et al. [6].

3.2 Incremental and hierarchical median problems

3.2.1 The incremental k-median problem For convenience, we restate the definition of the k-median problem. Given the locations of a set F of $|F| = n_f$ facilities and a set C of $|C| = n_c$ clients in a metric space, *the k-median problem* is that of finding a set of k facilities to *open* such that the sum of the distances of the clients to the nearest open facility is minimized. The discussions in this section can be extended to the case when clients have demands and the problem is to find a set of k facilities so as to minimize the demand-weighted distances of clients to the nearest open facility. Let c_{ij} denote the distance between any two locations i and j. In the incremental k-median problem, we seek an ordering of the facilities. For each k, consider the ratio of the cost of opening the first k facilities in the ordering to the cost of an optimal k-median solution. The goal is to find an ordering that minimizes the maximum of this ratio over $k = 1, \ldots, n_f$. An algorithm for the problem is said to be *α-competitive* if the maximum of the ratio over all k is no more than α. This value α is called the *competitive ratio* of the algorithm.

We model the incremental median problem using our framework of Section 2 by the quadruple $\langle U, \preceq, \mathsf{ben}, \mathsf{cost}\rangle$. The set $U = 2^F$ is the set of all feasible solutions, each solution represented by the set of open facilities. The binary relation is given as $S_1 \preceq S_2$ iff $S_1 \supseteq S_2$, $\mathsf{ben}(S)$ equals $n_f - |S|$, and $\mathsf{cost}(S)$ is the cost of connecting the clients to their nearest facilities in S. The output of our incremental approximation algorithm is a chain of subsets of the facilities, where each chain element (subset of facilities) is a subset of the previous element. As shown in Theorem 3.2 below, the desired sequence of facilities for the incremental median problem is simply a concatenation of the differences between consecutive sets of this chain, presented in reverse order. The main claim of the following lemma is implicit in [32] and [13].

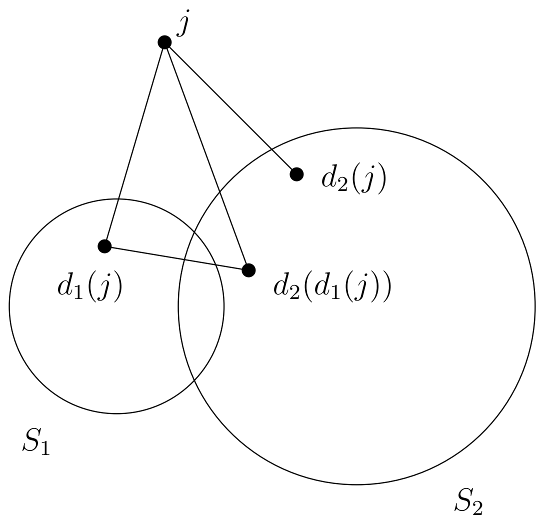

Figure 1: Proof of Lemma 3.2

Lemma 3.2. *There exists a $(1,2)$-augmentation for the incremental median problem. A $(1,2\alpha)$-augmentation can be efficiently implemented, where $\alpha = 3+\epsilon$. Also a $(1,2)$-nesting exists and can be efficiently implemented.*

Proof. Let S_2 ($\mathsf{ben}(S_2) < p$) be a set of facilities. We would like to augment it to get a benefit of at least p. Let S_1 be a set of facilities with benefit p. According to the definition, $|S_2| > |S_1|$. We aim to find a subset S such that $S_2 \preceq S$, i.e., $S \subset S_2$, and $|S| \leq |S_1|$.

For any location (client or facility) j, let $d_1(j)$ (resp., $d_2(j)$) be the closest facility to j in S_1 (resp., S_2). For any client j let us bound the distance $c_{j,d_2(d_1(j))}$ (see Figure 1):

$$\begin{aligned} c_{j,d_2(d_1(j))} &\leq c_{j,d_1(j)} + c_{d_1(j),d_2(d_1(j))} \\ &\leq c_{j,d_1(j)} + c_{d_1(j),d_2(j)} \\ &\leq c_{j,d_1(j)} + c_{j,d_1(j)} + c_{j,d_2(j)} \\ &= 2c_{j,d_1(j)} + c_{j,d_2(j)}, \end{aligned}$$

where the second inequality follows since $d_2(d_1(j))$ is the closest median in S_2 to $d_1(j)$. Define $S = \{d_2(i) : i \in S_1\}$; that is, S is the set of facilities in S_2 that are closest to the facilities in S_1. Let $d(j)$ be the closest facility in S for a location j. For any client j, $c_{j,d(j)} \leq c_{j,d(d_1(j))} = c_{j,d_2(d_1(j))} \leq 2c_{j,d_1(j)} + c_{j,d_2(j)}$. Summing this over all clients, we obtain $\mathsf{cost}(S) \leq \mathsf{cost}(S_2) + 2\mathsf{cost}(S_1)$. Note that $S \subset S_2$ and $|S| \leq |S_1|$. This proves the existence and efficient implementation of a $(1,2)$-nesting.

Using an optimum solution (resp., α-approximate solution [3]) to the k-median problem for S_1 proves the existence (resp., efficient implementation) of the $(1,2)$-augmentation (resp., $(1,2\alpha)$-augmentation). □

Theorem 3.2. *There exists a solution to the incremental median problem with competitive ratio 8. A deterministic solution with competitive ratio 8α and a randomized solution with competitive ratio $2e\alpha$ can be computed efficiently, where $\alpha = 3+\epsilon$.*

Proof. The existence and computability of chains of $\langle U, \preceq \rangle$ with the desired competitive ratios follow immediately from Lemma 3.2 and Theorem 2.1. To convert a given chain $\mathcal{C}$ of facility sets into a sequence of medians, we simply generate a maximal chain containing $\mathcal{C}$ and concatenate the differences between consecutive sets of this chain in reverse order. By the definition of competitive ratio (see Section 2.1), the competitive ratio of the chain is at least that of the median sequence, thus completing the proof of the theorem. □

3.2.2 The hierarchical k-median problem We define *an assignment* of a k-median solution as function from clients to facilities that assigns each client to an open facility in the solution. In the hierarchical k-median problem, we give an ordering of facilities along with assignments $a_1, \ldots, a_{n_f}$ such that the assignment a_k assigns clients only to the first k facilities in the ordering; this corresponds to a clustering with k clusters. To ensure that the clusterings are formed by merging pairs of clusters, we require that for any two assignments a_k and a_{k-1} that a_{k-1} can be obtained from a_k by reassigning all the clients assigned to the kth facility in the ordering to a single facility earlier in the ordering. Now consider the ratio of the cost of assignment a_k to the cost of an optimal k-median solution. The goal of the problem is to find an ordering of facilities and a valid sequence of assignments so as to minimize the maximum of this ratio over all $k = 1, \ldots, n_f$. An algorithm for the problem is said to be *α-competitive* if the maximum of the ratio over all k is no more than α.

We show how to cast the hierarchical median problem into our incremental optimization framework. A solution to the k-median problem is represented as a pair (S, a) containing a subset S of facilities and an assignment a of clients to facilities in S. In the incremental k-median problem, the assignment function assigns each client to its nearest available facility. This cannot be assumed for the hierarchical median problem. For any i in S, let $a^{-1}(i)$ be the set of clients assigned to i by a. Given a solution (S, a), we say that a is *locally-optimal* for S if for all i in S, assigning all clients in $a^{-1}(i)$ to any other single facility in S will not decrease the total cost. We adopt the convention that if the assignment is omitted, the default assignment is to assign each client to its nearest available facility.

In the quadruple $\langle U, \preceq, \mathsf{ben}, \mathsf{cost}\rangle$, we let U be the set of all pairs (S, a) such that a is locally-optimal in (S, a). It is easy to see that U includes all optimal k-median solutions, for all values of k. Let $\mathsf{cost}(S, a)$ represents the cost of assigning the clients to the facilities in S according to the assignment a whereas $\mathsf{cost}(S)$ represents the cost of assigning clients to the nearest facility in S.

Definition 3.1. *Given solution pairs (S_1, a_1) and (S_2, a_2) in U, we say (S_1, a_1) is* nested *in (S_2, a_2) if*

1. *$S_1 \subset S_2$;*
2. *$\forall j \in C$, if $a_2(j) \in S_1$ then $a_1(j) = a_2(j)$;*
3. *$\forall j, k \in C$, if $a_2(j) = a_2(k)$ then $a_1(j) = a_1(k)$.*

We denote nested solutions by $(S_1, a_1) \subset (S_2, a_2)$.

We define $\preceq$ as $(S_2, a_2) \preceq (S_1, a_1)$ iff $(S_1, a_1) \subset (S_2, a_2)$, the benefit of a solution (S, a) as $n_f - |S|$, and the cost of (S, a) to be the service cost for the clients according to the assignment a. By definition, the benefit function is monotonically non-decreasing with the partial order. The same holds for the cost function since the assignment in any solution is locally optimal. We now develop an incremental approximation for $\langle U, \preceq, \mathsf{ben}, \mathsf{cost}\rangle$ and show that a chain output by this algorithm can be transformed to a hierarchical ordering of solution pairs, with the desired competitive ratio.

We first prove the following lemma which will be useful in deriving the augmentation lemma.

Lemma 3.3. *Given a set V_1 of facilities and a solution $(V_2, a_2) \in U$ such that $V_1 \subseteq V_2$, we can obtain a solution $(V_1, a_1) \in U$ such that (V_1, a_1) is nested in (V_2, a_2) and $\mathsf{cost}(V_1, a_1) \leq 2\mathsf{cost}(V_2, a_2) + \mathsf{cost}(V_1)$.*

Proof. Let $d_1(j)$ denote the nearest median in V_1 to the client j. We define two functions P and Q. The function P maps each facility in $V_2 \setminus V_1$ to the nearest facility in V_1. This is the "parent" function of the hierarchical algorithms of Dasgupta and Long [15] and Plaxton [37]. The function Q maps each facility i in $V_2 \setminus V_1$ to a facility in V_1, which services the clients in $a_2^{-1}(i)$ at the least total cost, among all facilities in V_1.

We now create assignment a_1: for any client j, $a_1(j) = a_2(j)$ if $a_2(j) \in V_1$ and $a_1(j) = Q(a_2(j))$ if $a_2(j) \in V_2 \setminus V_1$. It is easy to verify that (V_1, a_1) is locally-optimal and the pairs (V_1, a_1) is nested in (V_2, a_2). Consider the assignment a, which is defined the same way as a_1 by replacing P for Q. By the definition of Q, it follows that $\mathsf{cost}(V_1, a_1) \leq \mathsf{cost}(V_1, a)$. If we show that $c_{j,a(j)} \leq 2c_{j,a_2(j)} + c_{j,d_1(j)}$ for all clients j, then summing this over all clients in C will give the required result.

If $a_2(j) \in V_1$ then $c_{j,a(j)} = c_{j,a_2(j)} \leq 2c_{j,a_2(j)} + c_{j,d_1(j)}$. If $a_2(j) \in V_2 \setminus V_1$ then

$$\begin{aligned} c_{j,a(j)} = c_{j,P(a_2(j))} &\leq c_{j,a_2(j)} + c_{a_2(j),P(a_2(j))} \\ &\leq c_{j,a_2(j)} + c_{a_2(j),d_1(j)} \\ &\leq 2c_{j,a_2(j)} + c_{j,d_1(j)}. \end{aligned}$$

The second inequality follows from the definition of $P(\cdot)$; the third is due to triangle inequality. $\square$

LEMMA 3.4. *There exists a* $(3,2)$*-augmentation for the hierarchical median problem. A* $(3,2\alpha)$*-augmentation can be efficiently implemented, where* $\alpha = 3+\epsilon$.

Proof. Let the subroutine $\mathsf{Augment}(S,p,\gamma,\delta)$ be called with $S = (V_2,a_2) \in U$ and $\mathsf{ben}(S) < p$. Let V be a solution to the k-median problem with benefit p and let a be the closest facility assignment of clients to V. Using Lemma 3.2 we can find another set $V_1 \subset V_2$ with $|V_1| = |V|$ such that $\mathsf{cost}(V_1) \le \mathsf{cost}(V_2) + 2\mathsf{cost}(V)$. Using V_1 and (V_2,a_2) in Lemma 3.3 we get $\mathsf{cost}(V_1,a_1) \le \mathsf{cost}(V_1) + 2\mathsf{cost}(V_2,a_2)$. Since $\mathsf{cost}(V_2) \le \mathsf{cost}(V_2,a_2)$,

$$\begin{aligned}\mathsf{cost}(V_1,a_1) &\le \mathsf{cost}(V_1) + 2\mathsf{cost}(V_2,a_2)\\ &\le 2\mathsf{cost}(V) + \mathsf{cost}(V_2) + 2\mathsf{cost}(V_2,a_2)\\ &\le 3\mathsf{cost}(V_2,a_2) + 2\mathsf{cost}(V).\end{aligned}$$

Using an optimum solution (resp., an α-approximate solution [3]) to the k-median problem for V proves the first (resp., second) assertion. $\square$

THEOREM 3.3. *There exists a solution to the hierarchical median problem with competitive ratio* 24. *A deterministic solution with competitive ratio* 24α *and a randomized solution with competitive ratio* 10.76α *can be computed efficiently, where* $\alpha = 3+\epsilon$.

Proof. The existence and computability of chains of $\langle U, \preceq\rangle$ with the desired competitive ratios follow from Lemma 3.4 and Theorem 2.1. To convert a given chain $\mathcal{C}$ into a hierarchical sequence of solution pairs, we return the reverse of a maximal chain that contains $\mathcal{C}$. One can obtain a maximal chain of a given chain as follows: between any consecutive pair of solution pairs (S_0,a_0) and (S,a) such that $|S_0| = |S| + k$, for some $k > 1$, insert solution pairs (S_1,a_1), (S_2,a_2), $\ldots$, (S_{k-1},a_{k-1}), where S_i equals $S_{i-1} \setminus \{f\}$ for an arbitrary f in $S_{i-1} \setminus S$ and a_i is identical to a_{i-1} except that all clients assigned to f in a_{i-1} are assigned to a facility in S_i that offers the least service cost. By the definition of competitive ratio (see Section 2.1), the competitive ratio of the chain is at least that of the hierarchical sequence, thus completing the proof of the theorem. $\square$

Here we use the ALTINCAPPROX(γ,δ,α) algorithm to give better approximation ratios for the hierarchical k-median problem. We can then take advantage of the fact that two solutions that we wish to nest have costs that are geometrically related. We use α-approximate solutions k-median solutions $V_1, V_2, \ldots, V_{B_{max}}$ and then bucket them according to Step 3 of the algorithm to arrive at solutions $\overline{V}_1, \overline{V}_2, \ldots, \overline{V}_r = V_{B_{max}}$ one from each non-empty bucket. Now we get nested solutions along with their assignments from these solutions using Lemma 3.2 and Lemma 3.3 as follows. Let $S_1 = \overline{V}_1$ and a_1 be the assignment of clients to the nearest facility in S_1. S_{i+1} is iteratively obtained using Lemma 3.2 on S_i and $\overline{V}_{i+1}$ with

$$\mathsf{cost}(S_{i+1}) \le 2\mathsf{cost}(\overline{V}_{i+1}) + \mathsf{cost}(S_i).$$

Then assignment a_{i+1} on S_{i+1} can be obtained from S_i, a_i and S_{i+1} using Lemma 3.3 with

$$\mathsf{cost}(S_{i+1},a_{i+1}) \le 2\mathsf{cost}(S_i,a_i) + \mathsf{cost}(S_{i+1}).$$

THEOREM 3.4. *There exists a solution to the hierarchical median problem with competitive ratio* 20.71. *A deterministic solution with competitive ratio* 20.71α *and a randomized solution with competitive ratio* 10.03α *can be computed efficiently, where* $\alpha = 3+\epsilon$.

Proof. The nested solutions obtained from the ALTINCAPPROX(γ,δ,α) give the corresponding approximation ratios for the hierarchical k-median problem.

Fix a $p \le B_{max}$. Let i be such that $\mathsf{ben}(S_{i-1},a_{i-1}) < p \le \mathsf{ben}(S_i,a_i)$. Let the solution $\overline{V}_i$ be from the bucket $(M/\beta, M]$. So $\mathsf{cost}(\overline{V}_i) \le M$, $\mathsf{cost}(\overline{V}_{i-1}) \le M/\beta$ and so on.

$$
\begin{aligned}
\mathsf{cost}(S_i, a_i) &\leq \mathsf{cost}(S_i) + 2\mathsf{cost}(S_{i-1}, a_{i-1}) \\
&\leq \mathsf{cost}(S_i) + 2\mathsf{cost}(S_{i-1}) + 2^2\mathsf{cost}(S_{i-2}, a_{i-2}) \\
&\leq \sum_{j=0}^{i} 2^j \mathsf{cost}(S_{i-j}) \\
&\leq 2\mathsf{cost}(\overline{V}_i) + \mathsf{cost}(S_{i-1}) + \sum_{j=1}^{i} 2^j \mathsf{cost}(S_{i-j}) \\
&= 2\mathsf{cost}(\overline{V}_i) + (1+2)\mathsf{cost}(S_{i-1}) + \sum_{j=2}^{i} 2^j \mathsf{cost}(S_{i-j}) \\
&\leq 2\mathsf{cost}(\overline{V}_i) + (1+2)(2\mathsf{cost}(\overline{V}_{i-1}) + \mathsf{cost}(S_{i-2})) + \sum_{j=2}^{i} 2^j \mathsf{cost}(S_{i-j}) \\
&\leq \sum_{j=0}^{i} \left(\sum_{m=0}^{j} 2^m \right) 2\mathsf{cost}(\overline{V}_{i-j}) \\
&\leq 2\sum_{j=0}^{i} (2^{j+1} - 1) \frac{M}{\beta^j} \\
&\leq 2M \left(\sum_{j=0}^{\infty} \frac{2^{j+1}}{\beta^j} - \sum_{j=0}^{\infty} \frac{1}{\beta^j} \right) \\
&= 2M \left(\frac{2\beta}{\beta-2} - \frac{\beta}{\beta-1} \right) \\
&= \frac{2\beta^2 M}{(\beta-1)(\beta-2)}.
\end{aligned}
$$

$$
\mathsf{cost}(S_i, a_i) \leq \frac{2\beta^2 M}{(\beta-1)(\beta-2)} \leq \frac{2\beta^3 \mathsf{cost}(V_p)}{(\beta-1)(\beta-2)} \leq \frac{2\beta^3 \alpha}{(\beta-1)(\beta-2)} \mathsf{cost}(\mathsf{Opt}(p)).
$$

Optimizing for β gives $\beta = 3 + \sqrt{3}$ which gives a 20.709α-competitive approximation for hierarchical k-median problem. For the randomized algorithm, we obtain

$$
E[\mathsf{cost}(S_i, a_i)] \leq \frac{2\beta^2 E[M]}{(\beta-1)(\beta-2)} = \frac{2\beta^2 \mathsf{cost}(V_p)}{(\beta-2)\ln\beta} \leq \frac{2\beta^2 \alpha}{(\beta-2)\ln\beta} \mathsf{cost}(\mathsf{Opt}(p))
$$

Optimizing this equation for β gives $\beta = 6.355$ which gives a 10.03α-competitive randomized approximation algorithm for the hierarchical k-median problem. □

3.2.3 The incremental k-means problem This problem is identical to the k-median counterpart, except that the k-means cost function is the the sum of squares of the distances of the clients to their nearest open facility. The set of solutions, their benefit, the binary operator, and the structure of posets are exactly the same as that of the k-median problem.

LEMMA 3.5. *There exists a $(2, 8)$-augmentation for the incremental k-means problem. Given an α-approximation algorithm to the k-means problem, a $(2, 8\alpha)$-augmentation can be computed. Also, a $(2, 8)$-nesting exists and can be efficiently implemented.*

Proof. From the proof of Lemma 3.2 we know that for every location j, $c_{j,d(j)} \leq 2c_{j,d_1(j)} + c_{j,d_2(j)}$. Here $d(j), d_1(j)$

and $d_2(j)$ are as defined in the proof. Squaring this equation we get

$$\begin{aligned} c^2_{j,d(j)} &\leq 4c^2_{j,d_1(j)} + c^2_{j,d_2(j)} + 4c_{j,d_1(j)}c_{j,d_2(j)} \\ &\leq 4c^2_{j,d_1(j)} + c^2_{j,d_2(j)} + \tau^2 c^2_{j,d_1(j)} + \frac{4}{\tau^2}c^2_{j,d_2(j)} \\ &\leq (4+\tau^2)c^2_{j,d_1(j)} + \left(1+\frac{4}{\tau^2}\right)c^2_{j,d_2(j)} \end{aligned}$$

for all $\tau > 0$. The result follows by summing the above equation over all clients and taking $\tau = 2$. □

THEOREM 3.5. *There exists a solution to the incremental k-means problem with competitive ratio 64. Given a polynomial-time α-approximation for the k-means problem, a deterministic solution and a randomized solution with competitive ratios 64α and 31.82α, respectively, can also be computed efficiently.*

Proof. This is immediate from Lemma 3.5 and Theorem 2.1. For the randomized case we minimize the competitive ratio for the values of τ and β to get a 32.82α competitive algorithm with $\tau = 1.61$ and $\beta = 6.47$. □

3.2.4 The hierarchical k-means problem This problem again is very similar to the hierarchical k-median problem with a slight change in the cost function. In the hierarchical k-means problem, we give an ordering of facilities along with assignments $a_1, \ldots, a_{n_f}$ such that the assignment a_k assigns clients only to the first k facilities in the ordering. Here as in the hierarchical median problem the assignments are nested. The cost of an assignment a is the sum of squares of the distances of the clients to the facility assigned to that client by the assignment a. Now consider the ratio of the cost of assignment a_k to the cost of an optimal k-means solution. The goal of the problem is to find an ordering of facilities and a valid sequence of assignments so as to minimize the maximum of this ratio over all $k = 1, \ldots, n_f$.

The set of solutions, their benefit, the binary operator, and the structure of posets are exactly the same as that of the hierarchical k-median problem.

LEMMA 3.6. *Given two k-means solutions V_1 and V_2, $V_1 \subset V_2$ and an assignment a_2 on V_2 we can obtain an assignment a_1 on V_1 such that (V_1, a_1) and (V_2, a_2) are nested and $\mathsf{cost}(V_1, a_1) \leq (4+\rho^2)\mathsf{cost}(V_2, a_2) + (1 + \frac{4}{\rho^2})\mathsf{cost}(V_1)$ for all $\rho > 0$.*

Proof. We define the functions d_1 and P and the assignment a_1 exactly as we defined in the proof of Lemma 3.3. From the proof we know that $c_{j,a_1(j)} \leq 2c_{j,a_2(j)} + c_{j,d_1(j)}$ for all clients j. By squaring this equation, we get

$$\begin{aligned} c^2_{j,a_1(j)} &\leq 4c^2_{j,a_2(j)} + c^2_{j,d_1(j)} + 4c_{j,a_2(j)}c_{j,d_1(j)} \\ &\leq 4c^2_{j,a_2(j)} + c^2_{j,d_1(j)} + \rho^2 c^2_{j,a_2(j)} + \frac{4}{\rho^2}c^2_{j,d_1(j)} \\ &\leq (4+\rho^2)c^2_{j,a_2(j)} + \left(1+\frac{4}{\rho^2}\right)c^2_{j,d_1(j)} \end{aligned}$$

for all $\rho > 0$. We arrive at the required result by summing the above equation over all clients. □

LEMMA 3.7. *There exists an $(18, 8)$-augmentation for the hierarchical k-means problem. Given an α-approximation algorithm for the k-means problem, an $(18, 8\alpha)$-augmentation can be computed. Also, an $(18, 8)$-nesting exists and can be efficiently implemented.*

Proof. Given (V_2, a_2), with $\mathsf{ben}(V_2) < p$, let V be a solution to the k-means problem with benefit p. Along the same lines of proof of the Lemma 3.5 we can find another set $V_1 \subset V_2$ with $|V_1| = |V|$ such that $\mathsf{cost}(V_1) \leq (1 + 4/\tau^2)\mathsf{cost}(V_2) + (4+\tau^2)\mathsf{cost}(V)$. Using this V_1 and (V_2, a_2) in Lemma 3.6 we get $\mathsf{cost}(V_1, a_1) \leq$

$(4+\rho^2)\mathsf{cost}(V_2,a_2)+(1+4/\rho^2)\mathsf{cost}(V_1)$. Since $\mathsf{cost}(V_2)\le \mathsf{cost}(V_2,a_2)$ we get

$$\begin{aligned}\mathsf{cost}(V_1,a_1) &\le (4+\rho^2)\mathsf{cost}(V_2,a_2)+\left(1+\frac{4}{\rho^2}\right)\mathsf{cost}(V_1)\\ &\le (4+\rho^2)\mathsf{cost}(V_2,a_2)+\left(1+\frac{4}{\rho^2}\right)\left(1+\frac{4}{\tau^2}\right)\mathsf{cost}(V_2)+\left(1+\frac{4}{\rho^2}\right)(4+\tau^2)\mathsf{cost}(V)\\ &= \left[4+\rho^2+\left(1+\frac{4}{\rho^2}\right)\left(1+\frac{4}{\tau^2}\right)\right]\mathsf{cost}(V_2,a_2)+\left(1+\frac{4}{\rho^2}\right)(4+\tau^2)\mathsf{cost}(V)\end{aligned}$$

for all $\rho,\tau>0$. Optimizing for the values of ρ and τ so as to minimize the deterministic competitive ratio for the generalized algorithm gives $\rho=2\sqrt{2}$ and $\tau=2/\sqrt{3}$. This reduces the inequality to $\mathsf{cost}(V_1,a_1)\le 18\mathsf{cost}(V_2,a_2)+8\mathsf{cost}(V)$. Using an optimum solution to k-means problem for V proves the first assertion. Using an α-approximate solution to the k-means problem for V proves the second assertion. □

Theorem 3.6. *There exists a solution to the hierarchical k-means problem with competitive ratio* 576. *Given an α-approximation algorithm for the k-means problem, a deterministic and a randomized solution with competitive ratios* 576α *and* 151.1α *can be computed.*

Proof. In the deterministic case the competitive ratio is obtained by using the augmentation algorithm in Lemma 3.7. For randomized case we optimize the competitive ratio of the randomized algorithm for the values of ρ,τ and β to arrive at a competitive ratio of 151.01α with $\rho\simeq 3.15,\tau\simeq 1.01$ and $\beta\simeq 48.1$. The existence and computability of chains of $\langle U,\preceq\rangle$ with the desired competitive ratios follow immediately from Lemma 3.7 and Theorem 2.1. We use the same approach followed in the proof of Theorem 3.3 to convert a given chain of $\mathcal{C}$ of solution pairs to a hierarchical sequence of solution pairs, thus completing the proof of the theorem. □

3.3 The incremental facility location problem This problem was first defined by Plaxton [37], who also gives a $(4+\epsilon)\alpha$ competitive algorithm, where α is the best available approximation factor for the facility location problem. We show that our framework also handles this problem with competitive ratio 4α.

The setting is similar to that of the k-median problem. Here, though there is no restriction on the number of facilities to open, instead there is a facility cost $v(i)$ for opening a facility i in addition to the connection costs between each client and the nearest open facility. Without loss of generality, we assume that the minimum non-zero cost of opening a facility is 1; i.e., $\min\{v(i): i\in F, v(i)\neq 0\}$ is 1 (otherwise, the facility costs can be scaled appropriately). To define the incremental facility location problem, we introduce a positive scaling factor λ, so that the total cost associated with opening a subset $Y\subseteq F$ is

$$\mathsf{cost}_\lambda(C,Y)=\lambda\cdot\sum_{j\in C}c(j,Y)+\sum_{i\in Y}v(i),$$

where $c(j,Y)=\min_{i\in Y}c(j,i)$. The incremental problem is to compute an ordered sequence of the facilities F, $(f_1,f_2,\cdots,f_n)$ and a *threshold sequence*[4] $t_1\le t_2\le\cdots\le t_n$ drawn from $\mathbb{R}\cup\{\infty\}$, such that for any scaling factor $\lambda>0$ and k, the smallest index such that $t_k\ge\lambda$, $\mathsf{cost}_\lambda(C,\{f_i|i\le k\})$ is a good approximation to $\mathsf{Opt}_\lambda=\min_{Y\subseteq F}\mathsf{cost}_\lambda(C,Y)$.

To fit this problem into the framework described in Section 2, we conceptually reformulate the problem. For simplicity, we assume in the sequel that the opening cost for every location is positive. At the end of this section, we show how this assumption can be removed. Each solution element S of U is a subset of $F\times\mathbb{R}$ that satisfies the condition that for two distinct x and y in S, $\pi_1(x)\neq\pi_1(y)$, where π_i is the projection to the i^{th} coordinate, $i=1,2$. For nonempty $S\in U$, define $\mathsf{ben}(S)=\max\{\pi_2(s): s\in S\}$. The binary operator is defined as $S_1\preceq S_2$ iff $S_1\subseteq S_2$. If $S\neq\emptyset$, the cost function is defined as

$$\mathsf{cost}(S)=\mathsf{ben}(S)\cdot\sum_{j\in C}c(j,\pi_1(S))+\sum_{i\in\pi_1(S)}v(i).$$

[4] The definition of threshold sequence in [37] is slightly different from ours, but serves the same purpose.

We define $\mathsf{cost}(\emptyset)$ and $\mathsf{ben}(\emptyset)$ to be both 0. The incremental ordering $(S_1, \ldots, S_k)$ yields a sequence of facility-threshold pairs $\{(f_1, t_1), \ldots, (f_n, t_n)\}$, where S_i is a prefix of the sequence. Note that this form naturally gives the threshold sequence $(t_1, \ldots, t_n)$.

We note that in the incremental solution for the facility location problem, every facility set in the sequence is a *superset* of the preceding facility set in the sequence, whereas an incremental solution to the k-median problem is a sequence of facility sets, each obtained by *deleting* a facility from the preceding solution. This divergence is due to the differing roles of the parameters k and λ that form the basis of the incremental versions of the respective problems: the cost of an optimal solution to the k-median problem is nonincreasing with increasing k, whereas the cost of an optimal solution to the λ-facility location problem is nondecreasing with increasing λ.

We now establish the augmentation property for incremental facility location. We also present a companion $\mathsf{CostBound}$ subroutine that is monotonic in the benefit parameter; this helps prove the polynomial-time complexity of IncApprox for incremental facility location in Theorem 3.7.

Lemma 3.8. *There exists a $(1, 1)$-augmentation for the λ-facility location problem. A $(1, \alpha)$-augmentation can be computed efficiently, where $\alpha = 1.5$. Furthermore, a monotonic* $\mathsf{CostBound}$ *function can be defined.*

Proof. Given an $S \subseteq F \times \mathbb{R}$, and $\lambda > 0$, if $\mathsf{ben}(S) \geq \lambda$, we are done. Hence, we assume $\mathsf{ben}(S) < \lambda$. Let $F^* \subseteq F$ be an optimal solution for λ-facility location problem. (Equivalently, $S^* = F^* \times \lambda$.) We augment S to S' as follows.

1. Initialize: $S' \leftarrow S$.
2. If $|F^* \backslash \pi_1(S')| > 1$, goto Step 3. Otherwise, goto Step 4.
3. Pick one $f \in F^* \backslash \pi_1(S')$, append $(f, \mathsf{ben}(S))$ to S'. goto Step 2.
4. For the $f \in F^* \backslash \pi_1(S')$, append (f, λ) to S'. Exit.

By construction, $\mathsf{ben}(S') = \lambda$. Since $\pi_1(S^*) \subseteq \pi_1(S')$, $c(j, \pi_1(S')) \leq c(j, \pi_1(S^*))$. The cost associated with S' is bounded as follows.

$$\begin{aligned}
\mathsf{ben}(S') \cdot \sum_{j \in C} c(j, \pi_1(S')) + \sum_{i \in \pi_1(S')} v(i) &\leq \mathsf{ben}(S^*) \cdot \sum_{j \in C} c(j, \pi_1(S^*)) + \sum_{i \in \pi_1(S^*)} v(i) + \sum_{i \in \pi_1(S)} v(i) \\
&\leq \mathsf{cost}(S^*) + \mathsf{cost}(S).
\end{aligned}$$

To complete the proof of the first assertion of the lemma, we have $\mathsf{Augment}(S, \lambda, 1, 1)$ return S' and $\mathsf{CostBound}(S, \lambda, 1, 1)$ return $\mathsf{cost}(S) + \mathsf{cost}(S^*)$. By construction and the bound on the cost, the desired properties of the augmentation subroutine, as given in Definition 2.2, hold. Moreover, since the cost of an optimal solution for λ-facility location is nondecreasing with increasing λ, the $\mathsf{CostBound}$ subroutine is monotonic in the benefit parameter.

To prove the second assertion of the lemma, we replace F^* with the poly-time computable 1.5-approximate solution due to [9]. The subroutine $\mathsf{Augment}(S, \lambda, 1, 1.5)$ returns S' and $\mathsf{CostBound}(S, \lambda, 1, 1.5)$ returns $1.5\mathsf{LP\text{-}Opt}_\lambda + \mathsf{cost}(S)$, where $\mathsf{LP\text{-}Opt}_\lambda$ is the cost of an optimal fractional solution to the λ-facility location problem. By [9], the solution F^* computed has cost at most $1.5\mathsf{LP\text{-}Opt}_\lambda$. By definition and the upper bound on cost, the desired properties of the augmentation subroutine hold. Furthermore, $\mathsf{CostBound}$ is monotonic since the cost of an optimal fractional solution to the λ-facility location problem is nondecreasing with increasing λ. □

Theorem 3.7. *There exists an incremental solution for the incremental facility location problem with competitive ratio 4. Moreover, an incremental solution of ratio 4α and a randomized solution of expected ratio $e\alpha$ can be computed efficiently, where $\alpha = 1.5$.*

Proof. Since every opening cost is nonzero and hence at least one, the cost of any nonempty solution is at least the minimum opening cost, which is one. By definition, the cost of $\emptyset$ is zero. Therefore, the condition of Theorem 2.1 that the cost of a solution be either zero or at least one holds, and the desired claims about the incremental approximation ratio follow from Lemma 3.8 and Theorem 2.1.

It remains to argue that the incremental solutions of ratios 4α and $e\alpha$ can be computed efficiently. In particular, we need to argue that the search procedure during each iteration (including the initialization step) of IncApprox can be implemented in polynomial time. That is, we need to show how to determine the largest value λ for which $\mathsf{CostBound}(S, \lambda, \gamma, \delta)$ is at most a certain value C (zero in the initialization step and $\beta_0\beta^{i+1}$ in iteration i). The monotonicity of $\mathsf{CostBound}(S, \lambda, \gamma, \delta)$ with respect to λ suggests a binary search over λ. Since the range for λ is the set of positive reals, however, it is not obvious whether such a binary search will terminate in polynomial time. We argue that the desired λ can be obtained using binary search within an interval $[\lambda_0, \lambda_{max}]$ such that (a) the ratio λ_{max}/λ_0 is at most exponential in the size of the instance, and (b) the binary search can terminate when we have identified an interval whose size is inverse exponential in the size of the instance.

Consider any set Y of facilities. As λ increases, the cost of Y, as a solution for the λ-facility location problem, increases linearly in λ. So each possible subset Y is a line in a two-dimensional graph that plots the cost of Y, as a solution to λ-facility location, as a function of λ. The optimal solutions to the λ-facility location problem are then given by the lower envelope of these lines. Each vertex on this lower envelope represents a λ for which there exist $F_1, F_2 \subseteq F$ such that (a) $\mathsf{cost}_\lambda(C, F_1) = \mathsf{cost}_\lambda(C, F_2)$, and (b) the connection cost and opening cost of F_1 are different than those of F_2. The latter condition is true since F_1 and F_2 correspond to two different lines in the plot. We thus have

$$\lambda \cdot \sum_{j \in C} c(j, F_1) + \sum_{i \in F_1} v(i) = \lambda \cdot \sum_{j \in C} c(j, F_2) + \sum_{i \in F_2} v(i),$$

where $\sum_{j \in C} c(j, F_1) \neq \sum_{j \in C} c(j, F_2)$ and $\sum_{i \in F_1} v(i) \neq \sum_{i \in F_2} v(i)$. Suppose c_{max} and v_{max} are the largest distance and opening cost, respectively. Since all the distances and facility opening costs are integers, we obtain that λ is a rational number whose denominator is at most nc_{max} and the numerator at most nv_{max}. Thus, λ_{max} is at most nv_{max}, and λ_0 and the smallest difference between the λ values corresponding to two adjacent vertices of the lower envelope are at least $1/(nc_{max})$. Consequently a binary search over λ within the range $[1/(nc_{max}), nv_{max}]$ will terminate in time $\log(n^2 v_{max}(c_{max})^2)$, which is polynomial in the size of the instance. □

Finally, we consider the case where the opening cost of a facility could be zero. In this case, the condition of Theorem 2.1 that the cost of a solution be either zero or strictly bounded away from zero may not hold. To see this, let Y_0 be the subset of facilities in F where the opening cost is zero. Then, the cost of the solution $Y_0 \times \{\lambda\}$ tends to zero as λ tends to zero.

We now claim that the solution $S_\lambda = Y_0 \times \{\lambda\}$ achieves optimum cost for $\lambda \leq \lambda_0 = 1/(nc_{max})$, where c_{max} is the maximum distance between any two points. To see this, we first note that for $\lambda \leq 1/(nc_{max})$, the cost of any facility location solution $X \subset Y_0$ is at least that of Y_0 since the opening costs are zero for both solutions while the connecting cost of Y_0 is at most that of X. On the other hand, any solution that opens a facility outside Y_0 has a cost of at least 1, while the cost of Y_0 is at most $\lambda n c_{max} \leq 1$.

We next redefine the set of feasible solutions as follows. Each solution element S of U is a subset of $F \times \mathbb{R}$ such that for any two distinct x and y in S, $\pi_1(x) \neq \pi_1(y)$, and for all $f \in Y_0$, (f, λ_0) is in S. We now apply the framework of Section 2 to the incremental facility location and run IncApprox with initial solution $S_0 = Y_0 \times \{\lambda_0\}$ instead of $\emptyset$. In this case, the cost of any solution is either zero or strictly bounded away from zero. So we apply Lemma 3.8 and Theorem 2.1 to compute an ordered sequence of the facilities $(f_1, f_2, \ldots, f_n)$ and a threshold sequence such that for any scaling factor $\lambda > \lambda_0$, $\mathsf{cost}_\lambda(C, \{f_i | i \leq k\})$ is a good approximation to $\mathsf{Opt}_\lambda = \min_{Y \subseteq F} \mathsf{cost}_\lambda(C, Y)$, where k is the smallest index such that $t_k \geq \lambda$. Note that the first $|Y_0|$ facilities in this sequence are the elements of Y_0 while the first $|Y_0|$ thresholds are all λ_0. To obtain an incremental solution for all $\lambda > 0$, we return the same ordered sequence of facilities and the threshold sequence to be the same as before except that the first $|Y_0|$ values are all zeros instead of λ_0. For any $\lambda \in (0, \lambda_0]$, the incremental facility location solution returned is Y_0, which is optimal. For $\lambda > \lambda_0$, the desired competitive ratio(s) hold.

3.4 Incremental covering problems

3.4.1 The incremental k-set cover problem Given a universe X of n elements and a collection of subsets of X, $\mathcal{C} = \{C_1, \cdots, C_m\}$ and a cost function $c : \mathcal{C} \to Q^+$, find an ordered sequence of $\mathcal{C}$, such that for *any* $k \in [1, n]$, the minimal prefix of the sequence that covers k elements is a good approximation to the k-set cover problem. Recall that the k-set cover problem asks for a min-cost subcollection of $\mathcal{C}$ that covers at least k elements.

In the language of Section 2, the universe U is $2^{\mathcal{C}}$. The benefit of $S \subseteq \mathcal{C}$ is simply the total number of elements covered by S. Then $S_1 \preceq S_2$ iff $S_1 \subseteq S_2$, and $\mathsf{cost}(S)$ is the sum of the weights of the subsets in S.

Lemma 3.9. *There exists a $(1,1)$-augmentation for k-set cover problem. Moreover, a $(1,\alpha)$-augmentation can be computed efficiently, where $\alpha = \ln n + 1$.* □

Proof. The proof is straightforward. Given a partial solution S and k, if $\mathsf{ben}(S) \geq k$, we are done. Otherwise, let $S(k)$ be any set collection with $\mathsf{ben}(S(k)) \geq k$. Clearly, by setting $S' = S \cup S(k)$, we have $\mathsf{ben}(S') \geq k$ and its cost is bounded by

$$\mathsf{cost}(S) + \mathsf{cost}(S(k)).$$

Using the optimum solution to k-set cover for $S(k)$ proves the first assertion. And using a poly-time computable α-approximate solution (e.g., [39] gives $\alpha = \ln n + 1$) for $S(k)$ proves the second assertion.

Theorem 3.8. *There exists a solution for the incremental k-set cover problem with competitive ratio 4. Moreover, a solution with ratio 4α can be computed efficiently, where $\alpha = \ln n + 1$.*

Proof. Follows from Lemma 3.9 and Theorem 2.1. □

3.4.2 The incremental k-vertex cover problem Just as vertex cover is a special case of set cover, k-vertex cover problem is a special case of k-set cover, where each edge (resp., vertex) in the vertex cover problem is an element (resp., set) in the set cover poblem. We hence have a corresponding incremental vertex cover problem. A 2-approximation algorithm for k-vertex cover is known [8, 19, 35].

Theorem 3.9. *There exists an incremental solution for the incremental k-vertex cover problem with competitive ratio 4. Moreover, a solution with ratio 4α can be computed efficiently, where $\alpha = 2$.*

4 A general approach for problems with bounded envelope

In this section, we present a variant of our framework of Section 2 that yields improved incremental approximations for certain problems that admit approximate solution sets referred to as *bounded envelopes*. We first define an approximate bounded envelope and then present our framework for incremental approximation that uses bounded envelopes. In Section 5, we apply this new framework to obtain approximations for the incremental and hierarchical median problems with competitive ratios better than those obtained in Section 3.2.

We start by giving an intuitive definition of the bounded envelope before turning to a more formal definition. In a bounded envelope, we are given lower bounds on the values of our offline problem for all possible benefit values. As a function of benefit, the lower bounds are piecewise-linear and non-decreasing. We are also given α-approximate solutions at all breakpoints of the lower bounds, where the cost of the solution is at most α times the value of the lower bound. See Figure 2 for an illustration. The main idea is that we can then apply an algorithm like IncApprox to these α-approximate solutions, and get a chain of solutions at all the breakpoints. Then to provide solutions at benefit values not at the breakpoints, we must invoke an *interpolation algorithm* that can create solutions for intermediate benefits of cost no more than the interpolated costs of the solutions at the breakpoints. See Figure 3 for an illustration. Using the fact that the lower bounds between the breakpoint values are linear allows us to conclude that the cost of the solutions created for intermediate benefit values is within the corresponding factor of an optimal solution to the offline problem. Effectively an α-bounded envelope plays the same role as the α-approximation algorithm for the offline problem. We can use this to take advantage of cases in which we have a polynomial-time algorithm to produce an α-bounded envelope, but no polynomial-time α-approximation algorithm is known.

Now for a formal definition. Consider a problem Π specified by a quadruple $\langle U, \preceq, \mathsf{ben}, \mathsf{cost}\rangle$ as discussed in Section 2. We additionally assume that the range of benefit function is positive integers with maximum value B_{max}.

Definition 4.1. *An α-approximate bounded envelope of a problem Π is a triple $(\mathcal{E}, \mathsf{bnd}, \mathsf{sol})$, where $\mathcal{E} = \{n_1, n_2, \ldots, n_l\}$ with $1 = n_1 < n_2 < \cdots < n_l = B_{max}$, for some integer l, bnd is a function from $\{1, 2, \ldots, B_{max}\}$ to $\Re$, and sol is a function from $\mathcal{E}$ to U, satisfying the following conditions:*

1. $\mathsf{bnd}(k) \leq \mathsf{cost}(\mathsf{Opt}(k))$ *for* $1 \leq k \leq B_{max}$*;*
2. $\mathsf{cost}(\mathsf{sol}(n_i)) \leq \alpha \cdot \mathsf{bnd}(n_i)$ *for* $1 \leq i \leq l$*;*
3. $\mathsf{bnd}(k) = \mathsf{bnd}(n_{i-1}) + \frac{k-n_{i-1}}{n_i-n_{i-1}}(\mathsf{bnd}(n_i) - \mathsf{bnd}(n_{i-1}))$ *for* $n_{i-1} \leq k \leq n_i$ *and* $i = 2, \ldots, l$.

We note that any α-approximation for Π immediately yields an α-approximate bounded envelope. For some problems, however, one can obtain polynomial-time computable bounded envelopes with approximation factors

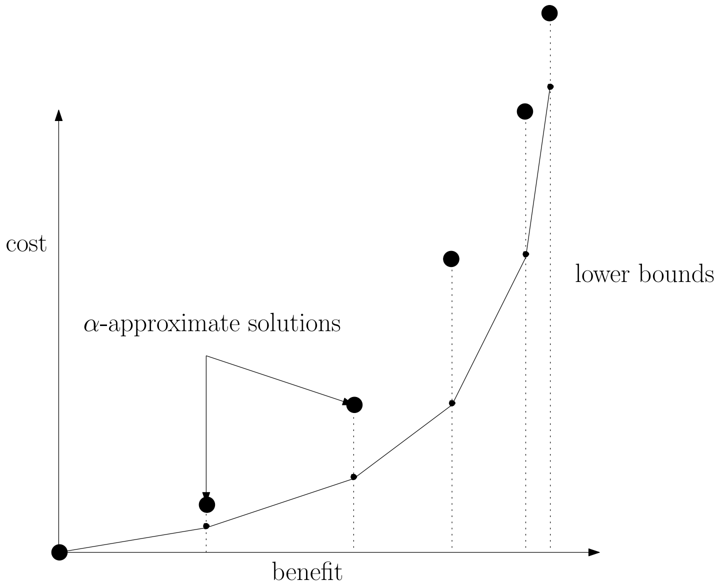

Figure 2: Example of a bounded envelope. We have piecewise linear, non-decreasing, lower bounds defined for all benefit values. α-approximate solutions are given at the breakpoints of the piecewise linear lower bounds.

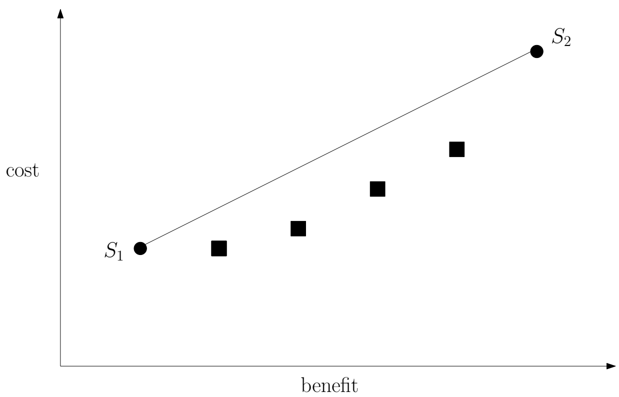

Figure 3: Example of an interpolation algorithm. We are given two solutions S_1 and S_2, where $S_1 \preceq S_2$, but $\mathsf{ben}(S_2) > \mathsf{ben}(S_1)$. The interpolation algorithm produces solutions with benefits intermediate to $\mathsf{ben}(S_1)$ and $\mathsf{ben}(S_2)$, but whose cost is no more than the line joining the costs of S_1 and S_2.

better than that of the best known polynomial-time approximations. This idea of an α-approximate bounded envelope was first used for the k-MST and the minimum latency problems by Archer, Levin, and Williamson [2], who obtain a faster and better approximation algorithm for the minimum latency problem. In Section 5, we present a 2-approximate bounded envelope for the k-median problem. Note that no 2-approximation algorithm is known for the k-median problem.

We now present a new framework for incremental approximation that yields better (poly-time computable) competitive ratios than the framework of Section 2 for certain problems that have (poly-time computable) bounded envelopes with approximation factors smaller than the best known (poly-time computable) approximation ratio. Our framework has three algorithmic ingredients: an approximate bounded envelope, an interpolation algorithm, and an augmentation algorithm. We now define the interpolation and augmentation algorithms. Recall from our intuitive explanation above that an interpolation algorithm allows us to take two nested solutions of different benefit values and obtain a solution of intermediate benefit that has cost at most the interpolation of the two solution costs (see again Figure 3).

Definition 4.2. *An interpolation algorithm $\mathcal{I}$ for problem Π is defined as an algorithm which when given two solutions S_1 and S_2 with $S_1 \preceq S_2$ and $\mathsf{ben}(S_1) < \mathsf{ben}(S_2)$, outputs a solution S such that $S_1 \preceq S \preceq S_2$, $\mathsf{ben}(S) = \mathsf{ben}(S_1) + 1$, and $\mathsf{cost}(S) \leq \mathsf{cost}(S_1) + \frac{1}{\mathsf{ben}(S_2) - \mathsf{ben}(S_1)}(\mathsf{cost}(S_2) - \mathsf{cost}(S_1))$.*

We define augmentation in a manner similar to that in Definition 2.2, with two differences: in the new definition, we only require the augmentation property for the benefit values in $\mathcal{E}$, and the cost upper bound on the augmented solutions uses the lower bound function bnd given by the bounded envelope, as opposed to the cost of an optimal solution. More precisely, we define the following augmentation property.

Definition 4.3. *(γ, δ)-BE-Augmentation: We say that the (γ, δ)-BE-augmentation property holds for reals $\gamma, \delta \geq 0$ with respect to a bounded envelope $(\mathcal{E}, \mathsf{bnd}, \mathsf{sol})$ if for every solution S of U and every real $p \in \mathcal{E}$, there exists an augmentation S' such that*

1. $S \preceq S'$.
2. $\mathsf{cost}(S') \leq \gamma \mathsf{cost}(S) + \delta \mathsf{bnd}(p)$.
3. $\mathsf{ben}(S') \geq p$.

As in Definition 2.2, we let $\mathsf{Augment}(S, p, \gamma, \delta)$ denote a subroutine that computes the above augmentation, with respect to an implicitly given bounded envelope $(\mathcal{E}, \mathsf{bnd}, \mathsf{sol})$. For efficiency reasons, we also introduce a companion subroutine $\mathsf{CostBound}(S, p, \gamma, \delta)$ which returns a bound on the cost of $\mathsf{Augment}(S, p, \gamma, \delta)$. In particular, for every feasible solution S and benefit $p \in \mathcal{E}$, we have

$$\mathsf{cost}(\mathsf{Augment}(S, p, \gamma, \delta)) \leq \mathsf{CostBound}(S, p, \gamma, \delta) \leq \gamma \mathsf{cost}(S) + \delta \mathsf{bnd}(p) \tag{4.2}$$

Now we present an algorithm for the incremental version of a problem Π using an α-approximate bounded envelope, an augmentation procedure and an interpolation algorithm $\mathcal{I}$ for the problem Π.

Theorem 4.1. *Assume that for any $S \in U$, $\mathsf{cost}(S)$ is either zero or bounded away from zero. If Π satisfies the (γ, δ)-BE-augmentation property with respect to a bounded envelope for reals $\gamma \geq 1$, $\delta > 0$, then (i) BoundedIncApprox(γ, δ) (Deterministic) computes an incremental solution for Π with competitive ratio $4\gamma\delta$; (ii) BoundedIncApprox(γ, δ) (Randomized) computes an incremental solution for Π with competitive ratio $\min_\beta \frac{\delta(\mu - 1)}{(1 - \gamma/\mu) \ln \mu}$, which equals $e\delta$, when $\gamma = 1$.*

Proof. The proof is similar to that of Theorem 2.1. If $\mathsf{cost}(\emptyset) \neq 0$, then we scale all the costs such that $\mathsf{cost}(\emptyset) = \beta$. Thus, we may assume, without loss of generality, that for any solution $S \in U$, $\mathsf{cost}(S)$ is either zero or $\geq \beta$.

We first establish the following invariant by induction on i.

$$\mathsf{cost}(S_i) \leq \beta_0 \beta^i \tag{4.3}$$

For the induction base $i = 0$, we consider two cases. If $\mathsf{cost}(\emptyset) = 0$, then $\mathsf{cost}(S_0) = 0$, else $\mathsf{cost}(S_0) = \mathsf{cost}(\emptyset) = \beta$; in either case the desired inequality holds. The induction step directly follows from the choice of S_{i+1} in Step (2) of the algorithm.

Algorithm 3 BoundedIncApprox(γ, δ)

1. Initialization:

   If $\mathsf{cost}(\emptyset) = 0$ then $S_0 = \mathsf{Augment}(\emptyset, q, \gamma, \delta)$, where q is the largest value for which $\mathsf{CostBound}(\emptyset, q, \gamma, \delta) = 0$. $S_0 = \emptyset$ otherwise.

   1D: (Deterministic) $i = 0$, $\beta = 2\gamma$, $\beta_0 = \beta$.

   1R: (Randomized) $i = 0$, β is the minimizer of $\frac{\beta-1}{(1-\gamma/\beta)\ln\beta}$, $\beta_0 = \beta^X$, where X is uniform from $[0, 1)$.

2. Iteration i: $S_{i+1} = \mathsf{Augment}(S_i, p, \gamma, \delta)$, where p is the largest value in the set $\mathcal{E} = \{n_1, n_2, \ldots, n_l\}$ from the bounded envelope for which $\mathsf{CostBound}(S_i, p, \gamma, \delta)$ is at most $\beta_0\beta^{i+1}$.
3. Termination: If $\mathsf{ben}(S_{i+1}) \neq B_{max}$ then $i \leftarrow i + 1$, go to step 2. Otherwise, return sequence $S_1, \cdots, S_{i+1}$ along with the solutions for all the intermediate benefit values obtained by interpolating between S_i's repeatedly using the interpolation algorithm.

Let $(\mathcal{E}, \mathsf{bnd}, \mathsf{sol})$ denote the bounded envelope with respect to which the problem Π satisfies the (γ, δ)-BE-augmentation property. Consider any $n_j \in \mathcal{E}$. Let k be the smallest integer such that

$$\delta\mathsf{bnd}(n_j) \leq \beta_0\beta^k \cdot (1 - \gamma/\beta). \tag{4.4}$$

We now argue that $\mathsf{cost}(\mathsf{Augment}(S_{k-1}, n_j, \gamma, \delta))$ is at most $\beta_0\beta^k$. It follows from our choice of k that

$$\mathsf{bnd}(n_j) > \beta_0\beta^{k-1} \cdot \frac{1 - \gamma/\beta}{\delta}. \tag{4.5}$$

We thus have

$$\begin{aligned}
\mathsf{cost}(\mathsf{Augment}(S_{k-1}, n_j, \gamma, \delta)) &\leq \gamma\mathsf{cost}(S_{k-1}) + \delta\mathsf{bnd}(n_j) && \text{(By Equation 4.2)} \\
&\leq \gamma\beta_0\beta^{k-1} + \beta_0\beta^k(1 - \gamma/\beta) && \text{(By Equations 4.3 and 4.4)} \quad (4.6) \\
&\leq \beta_0\beta^k. && (4.7)
\end{aligned}$$

Since by Step (2), $S_k = \mathsf{Augment}(S_{k-1}, p, \gamma, \delta)$ for the largest value of p in $\mathcal{E}$ such that $\mathsf{cost}(S_k) \leq \beta_0\beta^k$, Equation 4.7 implies $n_j \leq \mathsf{ben}(S_k)$. Let r be such that $\mathsf{ben}(S_{r-1}) < n_j \leq \mathsf{ben}(S_r)$. Then $n_j \leq \mathsf{ben}(S_r) \leq \mathsf{ben}(S_k)$.

Deterministic case: The approximation factor for the solution obtained with benefit n_j is no more than

$$\begin{aligned}
\mathsf{cost}(S_r)/\mathsf{bnd}(n_j) &\leq \mathsf{cost}(S_k)/\mathsf{bnd}(n_j) \\
&\leq \frac{\beta_0\beta^k}{\beta_0\beta^{k-1} \cdot \frac{1-\gamma/\beta}{\delta}} = \beta^2\delta/(\beta - \gamma).
\end{aligned}$$

This approximation factor holds for all the solutions returned by the algorithm with benefit $n_i \in \mathcal{E}$. Consider any solution S with benefit k returned by the algorithm with $n_{j-1} < k \leq n_j$. Let the solutions returned by the algorithm with benefits n_{j-1} and n_j be S_1 and S_2, respectively.

$$\begin{aligned}
\mathsf{cost}(S) &\leq \mathsf{cost}(S_1) + \frac{k - n_{j-1}}{n_j - n_{j-1}}(\mathsf{cost}(S_2) - \mathsf{cost}(S_1)) \\
&\leq \frac{\beta^2\delta}{\beta - \gamma}\left(\mathsf{bnd}(n_{j-1}) + \frac{k - n_{j-1}}{n_j - n_{j-1}}(\mathsf{bnd}(n_j) - \mathsf{bnd}(n_{j-1}))\right) \\
&= \frac{\beta^2\delta}{\beta - \gamma}\mathsf{bnd}(k) \\
&\leq \frac{\beta^2\delta}{\beta - \gamma}\mathsf{cost}(\mathsf{Opt}(k)).
\end{aligned}$$

The first inequality follows from the property of interpolation algorithm and the last equality follows from the property of the lower bounds of the bounded envelope. The above bound is minimized when $\beta = 2\gamma$, thus yielding a $4\delta\gamma$-approximation factor.

Randomized case: We omit the proof since it is very similar to that of the randomized case of Theorem 2.1.

□

As in the case of IncApprox, the running time of BoundedIncApprox is dominated by the calls to the augmentation subroutine. We can define CostBound to simply return the cost of the augmented solution, and perform a linear search to find the largest p in the set $\{n_1, n_2, \ldots, n_l\}$ with the desired property in each iteration. Then the number of calls per iteration is $O(B_{max})$, and since the number of iterations is $O(\log_\beta \mathsf{Maxcost})$, the number of calls to the augmentation subroutine is bounded by $O(B_{max} \cdot \log_\beta \mathsf{Maxcost})$.

We can obtain an improved running time if we define a CostBound subroutine that is monotonic in the benefit parameter and then perform a binary search to find the largest p in the set $\{n_1, n_2, \ldots, n_l\}$ with the desired property in each iteration. In this case, the number of calls per iteration is $O(\log B_{max})$, leading to a bound of $O(\log_\beta \mathsf{Maxcost} \cdot \log B_{max})$ on the number of calls to the augmentation subroutine.

5 Applications to incremental and hierarchical median problems

In this section, we apply our framework of Section 4 to the incremental and hierarchical median problems. We need to provide three ingredients: a bounded envelope, an interpolation algorithm, and an augmentation algorithm. We first present interpolation algorithms for the incremental and hierarchical median problems in Section 5.1. We next show in Section 5.2 that an α-approximate bounded envelope for the k-median problem yields a $(1, 2\alpha)$-augmentation (with respect to the bounded envelope) for the incremental and hierarchical median problems. We then present a polynomial-time algorithm for finding a 2-approximate bounded envelope for the k-median problem in Section 5.3. Finally, we put these ingredients together to obtain improved competitive ratios for the incremental and hierarchical median problems.

5.1 Interpolation algorithms

Lemma 5.1. *There exists a polynomial-time interpolation algorithm for the incremental median problem.*

Proof. Given two solutions S_1 and S_2 with $S_1 \prec S_2$ and $\mathsf{ben}(S_1) < \mathsf{ben}(S_2)$, we give a procedure to find a k-median solution S such that $\mathsf{ben}(S) = \mathsf{ben}(S_1) + 1$ and $\mathsf{cost}(S) \le \mathsf{cost}(S_1) + (\mathsf{cost}(S_2) - \mathsf{cost}(S_1))/(\mathsf{ben}(S_2) - \mathsf{ben}(S_1))$.

Since $S_1 \prec S_2$, $S_2 \subset S_1$. Let the assignment corresponding to assigning the clients to the closest facility in S_1 and S_2 be a_1 and a_2. For every facility $h \in S_1 \setminus S_2$ define $\Delta(h) = \sum_{j:a_1(j)=h}(c_{j,a_2(j)} - c_{j,a_1(j)})$. Then $\Delta(h)$ is the change in cost by shifting all the clients assigned to h by a_1 to the facilities they are assigned to by a_2. Observe that $\mathsf{cost}(S_1, a_1) - \mathsf{cost}(S_2, a_2) = \sum_{h \in S_1 \setminus S_2} \Delta(h)$ since assignment a_1 and a_2 assign clients to the nearest facility in S_1 and S_2 respectively.

Consider the facility $i \in S_1 \setminus S_2$ with minimum Δ value. Define an assignment a such that $a(j) = a_2(j)$ if $a_1(j) = i$ and $a(j) = a_1(j)$ otherwise. Let $S = S_1 \setminus \{i\}$.

$$\begin{aligned}
\mathsf{cost}(S) - \mathsf{cost}(S_1) &\le \mathsf{cost}(S, a) - \mathsf{cost}(S_1, a_1) \\
&= \Delta(i) \\
&\le \frac{1}{\mathsf{ben}(S_2) - \mathsf{ben}(S_1)} \sum_{h \in S_1 \setminus S_2} \Delta(h) \\
&= \frac{1}{\mathsf{ben}(S_2) - \mathsf{ben}(S_1)} (\mathsf{cost}(S_2) - \mathsf{cost}(S_1))
\end{aligned}$$

So we have found S such that $\mathsf{ben}(S) = \mathsf{ben}(S_1) + 1$, $S_1 \prec S \prec S_2$ and $\mathsf{cost}(S) \le \mathsf{cost}(S_1) + (\mathsf{cost}(S_2) - \mathsf{cost}(S_1))/(\mathsf{ben}(S_2) - \mathsf{ben}(S_1))$.

□

Lemma 5.2. *There exists a polynomial-time interpolation algorithm for the hierarchical median problem.*

Proof. The interpolation procedure for the hierarchical median problem is exactly the same as in the proof of Lemma 5.1. Here again $\mathsf{cost}(S_1, a_1) - \mathsf{cost}(S_2, a_2) = \sum_{h \in S_1 \setminus S_2} \Delta(h)$ since $(S_1, a_1) \prec (S_2, a_2)$ and so the assignment of clients differ only for the clients assigned to the facilities in $S_1 \setminus S_2$ in a_1. This gives (S, a) such that $(S_1, a_1) \prec (S, a) \prec (S_2, a_2)$, $\mathsf{ben}(S, a) = \mathsf{ben}(S_1, a_1) + 1$ and $\mathsf{cost}(S, a) \leq \mathsf{cost}(S_1, a_1) + (\mathsf{cost}(S_2, a_2) - \mathsf{cost}(S_1, a_1))/(\mathsf{ben}(S_2) - \mathsf{ben}(S_1))$. □

5.2 Augmentation algorithms

Lemma 5.3. *The procedure* $\mathsf{Augment}(S, p, 1, 2\alpha)$ *for* $p \in \mathcal{E}$ *can be efficiently implemented for the incremental median problem when an* α*-approximate bounded envelope with breakpoint set* $\mathcal{E}$ *is given.*

Proof. The proof of Lemma 3.2, with the solution S_1 replaced by the α-approximate solution with benefit p obtained from the bounded envelope, gives us the required result. Note that here $\mathsf{cost}(\mathsf{Augment}(S, p, 1, 2\alpha)) \leq \mathsf{cost}(S) + 2\alpha \mathsf{bnd}(p)$ where $\mathsf{bnd}(p)$ is the lower bound for benefit p provided by the bounded envelope as required by the algorithm BoundedIncApprox(γ, δ). To define a monontonic $\mathsf{CostBound}$ subroutine, we have $\mathsf{CostBound}(S, p, 1, 2\alpha)$ return $\mathsf{cost}(S) + 2\alpha \mathsf{bnd}(p)$. □

Lemma 5.4. *The procedure* $\mathsf{Augment}(S, p, 3, 2\alpha)$ *for* $p \in \mathcal{E}$ *can be efficiently implemented for the hierarchical median problem when an* α*-approximate bounded envelope with breakpoint set* $\mathcal{E}$ *is given.*

Proof. The proof of Lemma 3.4, with the solution S_1 replaced by the α-approximate solution with benefit p obtained from the bounded envelope, gives us the required result. Note that here $\mathsf{cost}(\mathsf{Augment}(S, p, 3, 2\alpha)) \leq 3\mathsf{cost}(S) + 2\alpha \mathsf{bnd}(p)$ where $\mathsf{bnd}(p)$ is the lower bound for benefit p provided by the bounded envelope as required by the algorithm in Section 4. To define a monontonic $\mathsf{CostBound}$ subroutine, we have $\mathsf{CostBound}(S, p, 3, 2\alpha)$ return $3\mathsf{cost}(S) + 2\alpha \mathsf{bnd}(p)$.

□

5.3 A 2-approximate bounded envelope We prove the existence of a 2-approximate bounded envelope in two stages. First we define a Lagrangean Multiplier Preserving (LMP) Facility Location (FL) algorithm and show the existence of a polynomial-time $(2 - 1/n)$-approximate LMP FL algorithm (Lemma 5.5 of Section 5.3.1). Then we show that for any $\epsilon > 0$, any $(\alpha - \epsilon)$-approximate LMP FL algorithm can be used to get an α-approximate bounded envelope for the incremental median problem in time polynomial in the size of the instance and $1/\epsilon$ (Lemma 5.9 of Section 5.3.2). The same bounded envelope also applies for the hierarchical median problem.

5.3.1 The metric facility location problem and LMP facility location algorithm Recall the standard metric facility location problem, whose incremental version is studied in Section 3.3. As in the case of the k-median problem, we have a set F of facilities and a set C of clients in a metric space. Let f_i be the opening cost of facility i and c_{ij} be the cost of connecting client j to facility i; c_{ij} will be the distance between client j and facility i. The goal of the facility location problem is to find a subset of facilities to open so that the cost of opening these facilities and connecting each client to an open facility is minimized. Let $n_c = |C|$, $n_f = |F|$, $n_c + n_f = n$ and $n_c \cdot n_f = m$.

Consider the LP relaxation of the standard integer programming formulation of the facility location problem. The 0-1 variable y_i denotes whether the facility i is open and the 0-1 variable x_{ij} denotes whether the client j is served by facility i.

$$
\begin{array}{llll}
 & \text{Min} & \sum_{i \in F} f_i y_i + \sum_{i \in F, j \in C} c_{ij} x_{ij} & \\
 & \text{subject to:} & & \\
 & & \sum_{i \in F} x_{ij} = 1 & \forall j \in C \\
(FL - P) & & x_{ij} \leq y_i & \forall i \in F, \forall j \in C \\
 & & x_{ij} \geq 0 & \forall i \in F, \forall j \in C \\
 & & y_i \geq 0 & \forall i \in F.
\end{array}
$$

The dual of this LP relaxation is

$$
\begin{array}{llll}
 & \text{Max} & \sum_{j\in C} v_j & \\
 & \text{subject to:} & & \\
 & & \sum_{j\in C} w_{ij} \le f_i & \forall i \in F \\
(FL-D) & & v_j - w_{ij} \le c_{ij} & \forall i \in F, \forall j \in C \\
 & & w_{ij} \ge 0 & \forall i \in F, \forall j \in C.
\end{array}
$$

Definition 5.1. *An algorithm $\mathcal{A}$ is said to be β-*Lagrangean multiplier preserving *(β-LMP) if it outputs an integral solution $(\hat{x},\hat{y})$ feasible for $(FL-P)$ and a dual solution $(\hat{v},\hat{w})$ feasible for $(FL-D)$ such that*

$$\sum_{i\in F, j\in C} c_{ij}\hat{x}_{ij} + \beta \sum_{i\in F} f_i \hat{y}_i \le \beta \sum_{j\in C} \hat{v}_j. \tag{5.8}$$

Lemma 5.5. *There exists a $(2-1/n)$-approximate Lagrangean Multiplier Preserving Facility Location algorithm.*

Proof. We want to find an integer solution $(\hat{x},\hat{y})$ to the facility location problem and a feasible solution $(\hat{v},\hat{w})$ to the facility location dual such that

$$\sum_{i\in F, j\in C} c_{ij}\hat{x}_{ij} + \beta \sum_{i\in F} f_i \hat{y}_i \le \beta \sum_{j\in C} \hat{v}_j$$

for $\beta = 2 - 1/n$.

We run the second greedy algorithm of [31] (called Algorithm 2 in that paper) on the problem instance with the same service costs, but facility costs given by βf_i for facility i. The greedy algorithm computes a value α_j for each client j and an integer solution $(\bar{x},\bar{y})$, which satisfy the following two conditions, as shown in [31]. First, by [31, Theorem 8.3], for any given set S of clients, $\sum_{j\in S}\alpha_j - \hat{f}_i \le \beta \sum_{j\in S} c_{ij}$. Second, we have the equality

$$\sum_{j\in C}\alpha_j = \sum_{i\in F, j\in C} c_{ij}\bar{x}_{ij} + \sum_{i\in F}\hat{f}_i\bar{y}_i = \sum_{i\in F, j\in C} c_{ij}\bar{x}_{ij} + \beta\sum_{i\in F} f_i\bar{y}_i.$$

To get our desired result, we set the primal integral solution $(\hat{x},\hat{y})$ to be the same as $(\bar{x},\bar{y})$. For the dual, we first set $\hat{v}_j = \alpha_j/\beta$. This immediately gives us

$$\sum_{i\in F, j\in C} c_{ij}\hat{x}_{ij} + \beta \sum_{i\in F} f_i \hat{y}_i \le \beta \sum_{j} \hat{v}_j.$$

All that remains is to specify $\hat{w}$ such that $(\hat{v},\hat{w})$ is a feasible dual solution. Pick an arbitrary $i \in F$, and let $S \subseteq F$ be the facilities such that $\hat{v}_j - c_{ij} \ge 0$. Then by [31, Theorem 8.3], we have that

$$\sum_{j\in S}\alpha_j - \beta f_i \le \beta \sum_{j\in S} c_{ij},$$

or

$$\beta\sum_{j\in S}\hat{v}_j - \beta f_i \le \beta \sum_{j\in S} c_{ij},$$

or

$$\sum_{j\in S}(\hat{v}_j - c_{ij}) \le f_i,$$

or

$$\sum_{j\in C}\max(\hat{v}_j - c_{ij}, 0) \le f_i$$

by the choice of S.

By taking $\hat{w}_{ij} = \max(\hat{v}_j - c_{ij}, 0)$, all three constraints of the dual are satisfied. Thus, $(\hat{v},\hat{w})$ is a feasible solution to the facility location dual $(FL-D)$, completing the proof of the lemma. □

5.3.2 Getting a bounded envelope for k-median problem We now present an algorithm which gives an $(\alpha+\epsilon)$-bounded envelope for the k-median problem given an α-LMP algorithm for metric uncapacitated facility location problem for any $\epsilon > 0$. In essence given an α-LMP algorithm for facility location problem, we arrive at k-median solutions $\mathsf{sol}(n_1), \ldots, \mathsf{sol}(n_l)$ for $1 = n_1 < \cdots < n_l = n$ with $|\mathsf{sol}(k)| = k$ and bounds $\mathsf{bnd}(1), \ldots, \mathsf{bnd}(n)$ satisfying the following properties. Let the cost of optimal k-median solution be denoted by OPT_k. Let us fix an $\epsilon > 0$.

1. $\mathsf{bnd}(k) \leq OPT_k$ for $1 \leq k \leq n$
2. $\mathsf{cost}(\mathsf{sol}(n_i)) \leq (\alpha+\epsilon) \cdot \mathsf{bnd}(n_i)$ for $1 \leq i \leq l$
3. $\mathsf{bnd}(k) = \mathsf{bnd}(n_{i-1}) + \frac{\mathsf{bnd}(n_i) - \mathsf{bnd}(n_{i-1})}{n_i - n_{i-1}}(k - n_{i-1})$ for $n_{i-1} \leq k \leq n_i$ and $i = 2, \ldots, l$

Let us consider the following integer program for the k-median problem:

$$
\begin{aligned}
\text{Min} \quad & \sum_{i \in F, j \in C} c_{ij} x_{ij} & \\
\text{subject to:} \quad & & \\
& \sum_{i \in F} x_{ij} = 1 & \forall j \in C \\
& x_{ij} \leq y_i & \forall i \in F, \forall j \in C \\
& \sum_{i \in F} y_i \leq k & \\
& x_{ij} \in \{0,1\} & \forall i \in F, \forall j \in C \\
& y_i \in \{0,1\} & \forall i \in F.
\end{aligned}
$$

The LP relaxation of the program is

$$
(k-P) \qquad
\begin{aligned}
\text{Min} \quad & \sum_{i \in F, j \in C} c_{ij} x_{ij} & \\
\text{subject to:} \quad & & \\
& \sum_{i \in F} x_{ij} = 1 & \forall j \in C \\
& x_{ij} \leq y_i & \forall i \in F, \forall j \in C \\
& \sum_{i \in F} y_i \leq k & \\
& x_{ij} \geq 0 & \forall i \in F, \forall j \in C \\
& y_i \geq 0 & \forall i \in F.
\end{aligned}
$$

The dual of the LP relaxation is

$$
(k-D) \qquad
\begin{aligned}
\text{Max} \quad & \sum_{j \in C} v_j - zk & \\
\text{subject to:} \quad & & \\
& \sum_{j \in C} w_{ij} \leq z & \forall i \in F \\
& v_j - w_{ij} \leq c_{ij} & \forall i \in F, \forall j \in C \\
& w_{ij} \geq 0 & \forall i \in F, \forall j \in C \\
& z \geq 0. &
\end{aligned}
$$

The essential idea is as follows. To obtain the k-median solutions we use an α-LMP algorithm for the facility location problem. Let us call this algorithm $\mathcal{A}$. We parameterize this algorithm with a common facility opening cost z for all facilities. This algorithm outputs an integral primal solution $(\hat{x}, \hat{y})$ and dual solution $(\hat{v}, \hat{w})$ from which we show that we can arrive at a solution S and a lower bound $\hat{b}$ such that $\mathsf{cost}(S) \leq \alpha\hat{b}$ and $\hat{b} \leq OPT_{|S|}$. We perform a binary search on z with the aim of obtaining a solution with exactly k nodes for each k. The binary search proceeds until we either find a solution or obtain two sufficiently close values of z, z_1 and z_2, and corresponding solutions S_1 and S_2 with their corresponding lower bounds $\hat{b}_1$ and $\hat{b}_2$ such that $|S_1| < k$ and $|S_2| > k$. In the latter case, we show that if z_1 and z_2 are close enough (parametrized by a given $\epsilon > 0$), then an interpolation of $\hat{b}_1$ and $\hat{b}_2$ with a slight scaling (determined by ϵ) gives a lower bound on the optimal solution with k nodes. Finally, we pick a subset of solutions from the collection in order to obtain the desired properties.

Since $\mathcal{A}$ is an α-LMP algorithm it outputs an integral solution $(\hat{x}, \hat{y})$ feasible for $(FL - P)$ and a solution $(\hat{v}, \hat{w})$ feasible for $(FL - D)$ such that

$$\sum_{i \in F, j \in C} c_{ij}\hat{x}_{ij} + \alpha \sum_{i \in F} f_i \hat{y}_i \leq \alpha \sum_{j \in C} \hat{v}_j. \tag{5.9}$$

Lemma 5.6. *Let $\mathcal{A}$ return $(\hat{x}, \hat{y})$ and $(\hat{v}, \hat{w})$ when $f_i = z$ for all i. Suppose $\sum_{i \in F} \hat{y}_i = k$. Let $S_k = \{i : y_i = 1, i \in F\}$ and $\hat{b}_k = (\sum_{j \in C} v_j) - zk$. Then $\mathsf{cost}(S_k) \leq \alpha \cdot \hat{b}_k$ and $\hat{b}_k \leq OPT_k$.*

Proof. Since $f_i = z$ for all i, $(\hat{x}, \hat{y})$ is an integral solution feasible for $(FL-P)$ and $(\hat{v}, \hat{w})$ is a dual solution feasible for $(FL - D)$, we have an integral solution $(\hat{x}, \hat{y})$ feasible for $(k - P)$ and a dual solution $(\hat{v}, \hat{w}, z)$ feasible for $(k - D)$. Substituting $f_i = z$ for all i and $\sum_{i \in F} \hat{y}_i = k$ in (5.9) we get

$$\sum_{i \in F, j \in C} c_{ij}\hat{x}_{ij} + \alpha z k \leq \alpha \sum_{j \in C} \hat{v}_j \tag{5.10}$$

Using this we get the following bound on the cost of the solution S_k:

$$\mathsf{cost}(S_k) = \sum_{i \in F, j \in C} c_{ij}\hat{x}_{ij} \leq \alpha \left(\left(\sum_{j \in C} \hat{v}_j \right) - zk \right) = \alpha \hat{b}_k.$$

Since $(\sum_{j \in C} \hat{v}_j - zk)$ is the dual objective of $(k - D)$, by weak duality $\hat{b}_k = (\sum_{j \in C} \hat{v}_j - zk) \leq OPT_k$. □

Let $c_{\max}$ be the maximum distance between a client and a facility. Here we assume that when $z = 0$ algorithm $\mathcal{A}$ outputs all facilities (that is, $y_i = 1\ \forall i \in F$), and when z is very large the algorithm $\mathcal{A}$ outputs only one facility. In particular, when $z = 2n^2 c_{max}$, the $(2 - 1/n_f)$-approximate LMP facility location algorithm of Jain et al. [31] (analyzed in Lemma 5.5) opens just one facility. For each k, we perform a binary search on the values of z in an attempt to find a value of z for which the algorithm returns a solution with exactly k facilities open. The binary search proceeds until we either find such a solution or obtain two values of z, z_1 and z_2 with $z_1 - z_2 \leq \epsilon c_{\min}/(\alpha n_f)$, and corresponding solutions S_1 and S_2 with their corresponding lower bounds $\hat{b}_1$ and $\hat{b}_2$ such that $|S_1| < k$ and $|S_2| > k$ (here $c_{\min}$ is the minimum nonzero distance between a facility and a client, and n_f is $|F|$). We next scale down $\hat{b}_k$ by a factor of $\alpha/(\alpha + \epsilon)$ to obtain lower bounds for the bounded envelope; that is, we set $\mathsf{bnd}(k) = \alpha\hat{b}_k/(\alpha + \epsilon)$.

Lemma 5.7. *Consider any solution S_k returned by the algorithm with its corresponding lower bound $\mathsf{bnd}(k) = \alpha\hat{b}_k/(\alpha + \epsilon)$. Then $\mathsf{cost}(S_k) \leq (\alpha + \epsilon) \cdot \mathsf{bnd}(k)$.*

Proof.

$$\mathsf{cost}(S_k) \leq \alpha \cdot \hat{b}_k = \alpha \cdot \left(\frac{\alpha + \epsilon}{\alpha} \right) \mathsf{bnd}(k) = (\alpha + \epsilon) \cdot \mathsf{bnd}(k),$$

where the first inequality is by Lemma 5.6. □

Lemma 5.8. *Assume $OPT_k \geq c_{\min}$. Let S_1 and S_2 be the solutions returned by the algorithm $\mathcal{A}$ with k_1 and k_2 facilities, facility opening costs z_1 and z_2, and lower bounds $\mathsf{bnd}(k_1)$ and $\mathsf{bnd}(k_2)$, respectively. Let $z_1 - z_2 \leq (\epsilon c_{\min})(\alpha n_f)$. Let $p, q \geq 0$ be such that $k = pk_1 + qk_2$ and $p + q = 1$. If we set $\mathsf{bnd}(k) = p\mathsf{bnd}(k_1) + q\mathsf{bnd}(k_2)$ then $\mathsf{bnd}(k) \leq OPT_k$.*

Proof. Let $(\hat{v}^1, \hat{w}^1)$ and $(\hat{v}^2, \hat{w}^2)$ be the dual solutions returned by the algorithm $\mathcal{A}$ corresponding to the facility opening costs z_1 and z_2. Let $v = p\hat{v}^1 + q\hat{v}^2$ and $w = p\hat{w}^1 + q\hat{w}^2$. Then

$$\begin{aligned}
\mathsf{bnd}(k) &= p\mathsf{bnd}(k_1) + q\mathsf{bnd}(k_2) \\
&= \frac{\alpha}{\alpha+\epsilon}\left(p\hat{b}_{k_1} + q\hat{b}_{k_2}\right) \\
&= \frac{\alpha}{\alpha+\epsilon}\left(p\left(\sum_{j\in C} v_j^1 - z_1k_1\right) + q\left(\sum_{j\in C} v_j^2 - z_2k_2\right)\right) \\
&= \frac{\alpha}{\alpha+\epsilon}\left(\sum_{j\in C} v_j - pz_1k_1 - qz_1k_2 + qz_1k_2 - qz_2k_2\right) \\
&= \frac{\alpha}{\alpha+\epsilon}\left(\sum_{j\in C} v_j - z_1\left(pk_1 + qk_2\right) + qk_2\left(z_1 - z_2\right)\right) \\
&\le \frac{\alpha}{\alpha+\epsilon}\left(\sum_{j\in C} v_j - z_1k\right) + \frac{\alpha}{\alpha+\epsilon}\cdot qk_2\cdot\frac{\epsilon}{\alpha n_f}c_{min} \\
&\le \frac{\alpha}{\alpha+\epsilon}\left(\sum_{j\in C} v_j - z_1k\right) + \frac{\epsilon}{\alpha+\epsilon}c_{min} \\
&\le OPT_k
\end{aligned}$$

The last inequality follows from the following facts: (a) since (v^1, w^1, z_1) and (v^2, w^2, z_1) are both feasible for the dual $(k-D)$, so is (v, w, z_1) by convexity of the feasible region, and (b) both $\left(\sum_{j\in C} v_j - z_1k\right)$ and $c_{\min}$ are lower bounds on OPT_k. $\square$

The lemma assumes that $OPT_k \ge c_{\min}$, which is false only if $OPT_k = 0$. To deal with this case, we run any approximation algorithm for the k-median problem for all values of k; note that if $OPT_k = 0$, an approximation algorithm will return an optimal solution of cost 0. For the values of k for which the algorithm returns a solution of cost 0, we set $b_k = 0$.

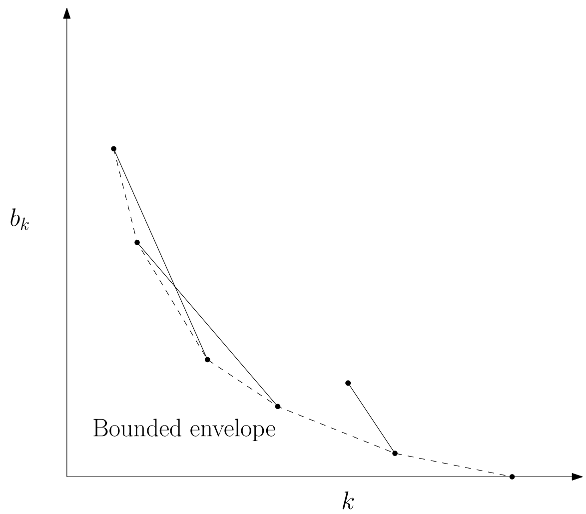

Figure 4: Bounded Envelope

At the end we have solutions S_k for some specific values of k and their corresponding lower bounds b_k such that $\mathsf{cost}(S_k) \leq (\alpha + \epsilon) \cdot b_k$. We plot these solutions S_k as points (k, b_k) in a two-dimensional graph with the interpolation of lower bounds, as in Lemma 5.8, represented as solid lines between (k_1, b_1) and (k_2, b_2) (See Figure 4). We also add the points corresponding to the optimal solutions for the values of k for which $OPT_k = 0$. We select a particular set of l solutions for $k = n_1, \ldots, n_l$ such that each point (n_i, b_{n_i}) is on the bounded envelope of the collection of points plotted on the graph. This fixes the values l and n_i for $i = 1, \ldots, l$. We can linearly interpolate the values b_k for $n_{i-1} < k < n_i$ so that property (3) of the properties of a bounded envelope is satisfied.

For values of k for which $OPT_k \geq c_{min}$, the lower envelope is below the solid lines and solid lines are valid lower bound by Lemma 5.8. For values of k for which $OPT_k = 0$, the bounded envelope coincides with value 0 and hence is a valid lower bound. This guarantees property (1) of the bounded envelope that $b_k \leq OPT_k$ for all k. So we have solutions S_{n_i} for $i = 1, \ldots, l$ and b_k for all k such that all three properties of $(\alpha + \epsilon)$-approximate bounded envelope are satisfied. Thus, we have established the following lemma.

Lemma 5.9. *There exists a polynomial-time computable* 2*-approximate bounded envelope for the incremental and hierarchical median problems.*

5.4 Putting it all together

Theorem 5.1. *For the incremental median problem, a* 16*-competitive deterministic solution and a* 4e*-competitive randomized solution can be computed efficiently. For the hierarchical median problem, a* 48*-competitive deterministic solution and a 21.52-competitive randomized solution can be computed efficiently.*

Proof. For the incremental median problem, both the deterministic and randomized cases are immediate from Lemmas 5.1, 5.3, and 5.9. For the hierarchical median problem, the deterministic and randomized bounds follow from Lemmas 5.2, 5.4, and 5.9. For the randomized case, we minimize the competitive ratio by selecting an appropriate value of β. □

We can obtain improved competitive ratios for the hierarchical median problem by using the same ideas as in Theorem 3.4 (the competitive ratios obtained are the same as in Theorem 3.4, with α set to 2). We omit the proof since the analysis details are identical to that of Theorem 3.4.

Theorem 5.2. *For the hierarchical median problem, a* 41.42*-competitive deterministic solution and a 20.06-competitive randomized solution can be computed efficiently.*

In summary, we developed a 2-bounded envelope for the k-median problem, and associated interpolation algorithms for both the incremental and hierarchical k-median problems, which allowed us to replace the factors of $(3 + \epsilon)$ in the competitive ratios of these problems coming from the approximation algorithm of Arya et al. [3] with a factor of 2.

6 Concluding Remarks

Our approach described in Section 2, and illustrated in Section 3, is general and can be easily used to handle other problems such as the k-center problem and the minimum dominating set problem. In Section 3.3, we have considered the incremental facility location problem introduced by [37]. Another natural incremental version of facility location can be defined using a partial facility location problem studied in [11], where all but s cities need to be served. Our approach again obtains an $O(1)$-competitive solution using a $O(1)$-approximation algorithm for the offline version.

One limitation of our work is that it may not lead to the best incremental solutions for a given problem. For instance, we can obtain an efficient 2-competitive algorithm for the unweighted vertex cover problem using the standard primal-dual approach (e.g., [41, Chap. 24]), while our generic approach only achieves a bound of 8. We also mention that for each of the problems discussed in the technical sections, there exists a constant c such that no c-competitive solution exists. For each of these problems, however, the best competitive ratio achievable is not known.

References

[1] O. Amini, F. V. Fomin, and S. Saurabh. Implicit branching and parameterized partial cover problems (extended abstract). In *FSTTCS*, pages 1–12, 2008.
[2] A. Archer, A. Levin, and D. P. Williamson. A faster, better approximation algorithm for the minimum latency problem. *SIAM Journal on Computing*, 37:1472–1498, 2008.
[3] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33:544–562, 2004.
[4] R. Bar-Yehuda. Using homogenous weights for approximating the partial cover problem. In *SODA*, pages 71–75, 1999.
[5] M. Bläser. Computing small partial coverings. *Inf. Process. Lett.*, 85(6):327–331, 2003.
[6] A. Blum, P. Chalasani, D. Coppersmith, B. Pulleyblank, P. Raghavan, and M. Sudan. The minimum latency problem. In *ACM STOC*, pages 163–171, 1994.
[7] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, UK, 1998.
[8] N. H. Bshouty and L. Burroughs. Massaging a linear programming solution to give a 2-approximation for a generalization of the vertex cover problem. In *STACS*, volume 1373 of *LNCS*, pages 298–308, 1998.
[9] J. Byrka. An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. Report PNA-E0611, CWI, Nov. 2006.
[10] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. *SIAM Journal on Computing*, pages 1417–1440, 2004.
[11] M. Charikar, S. Khuller, D. M. Mount, and G. Narasimhan. Algorithms for facility location problems with outliers. In *ACM-SIAM SODA*, pages 642–651, New York, Jan. 7–9 2001. ACM Press.
[12] M. Chrobak, C. Kenyon, J. Noga, and N. E. Young. Incremental medians via online bidding. *Algorithmica*, 50:455–478, 2008.
[13] M. Chrobak, C. Kenyon, and N. Young. The reverse greedy algorithm for the metric k-median problem. In *Information Processing Letters 97*, pages 68–72, 2006.
[14] B. Codenotti, G. De Marco, M. Leoncini, M. Montangero, and M. Santini. Approximation algorithms for a hierarchically structured bin packing problem. *Information Processing Letters*, 89:215–221, 2004.
[15] S. Dasgupta and P. Long. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70:555–569, 2005.
[16] B. Dean, M. Goemans, and J. Vondrak. Approximating the stochastic knapsack problem: the benefit of adaptivity. *Mathematics of Operations Research*, 33:945–964, 2008.
[17] U. Feige. A threshold of $\ln n$ for approximating set cover. *J. ACM*, 45(4):634–652, 1998.
[18] A. Fiat and G. J. Woeginger, editors. *Online Algorithms: The State of the Art*. Springer, 1998.
[19] R. Gandhi, S. Khuller, and A. Srinivasan. Approximation algorithms for partial covering problems. In *ICALP*, pages 225–236, 2001.
[20] N. Garg. Saving an epsilon: A 2-approximation for the k-MST problem in graphs. In *ACM STOC*, pages 396–402, 2005.
[21] M. X. Goemans and J. Kleinberg. An improved approximation ratio for the minimum latency problem. *Mathematical Programming*, 82:111–124, 1998.
[22] T. González. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.
[23] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *Journal of Algorithms*, 31:228–248, 1999.
[24] A. Gupta, M. Pal, R. Ravi, and A. Sinha. Boosted sampling: Approximation algorithms for stochastic optimization. In *ACM STOC*, pages 417–426, June 2004.
[25] A. Gupta, R. Ravi, and A. Sinha. LP rounding approximation algorithms for stochastic network design. *Mathematics of Operations Research*, 32:345–364, 2007.
[26] J. Hartline and A. Sharp. Hierarchical flow. In *INOC*, pages 681–687, 2005.
[27] J. Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.
[28] D. Hochbaum. The t-vertex cover problem: Extending the half integrality framework with budget constraints. In *APPROX*, pages 111–122, 1998.
[29] D. S. Hochbaum, editor. *Approximation Algorithms for NP-hard Problems*. PWS Publishing Company, Boston, MA, 1995.
[30] N. Immorlica, D. Karger, M. Minkoff, and V. Mirrokni. On the costs and benefits of procrastination: Approximation algorithms for stochastic combinatorial optimization problems. In *ACM-SIAM SODA*, pages 691–700, January 2004.
[31] K. Jain, M. Mahdian, E. Markakis, A. Saberi, and V. V. Vazirani. Greedy facility location algorithms analyzed using dual-fitting with factor-revealing LP. *Journal of the ACM*, 50:795–824, 2003.
[32] K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problem using the

primal-dual schema and Lagrangean relaxation. *Journal of the ACM*, 48:274–296, 2001.

[33] L. Jia, G. Lin, G. Noubir, R. Rajaraman, and R. Sundaram. Universal approximations for TSP, Steiner Tree, and Set Cover. In *ACM STOC*, pages 386–395, May 2005.

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

[35] J. Mestre. A primal-dual approximation algorithm for partial vertex cover: making educated guesses. In *RANDOM-APPROX*, pages 182–191, 2005.

[36] R. R. Mettu and C. G. Plaxton. The online median problem. *SIAM Journal on Computing*, 32:816–832, 2003.

[37] C. G. Plaxton. Approximation algorithms for hierarchical location problems. *Journal of Computer and System Sciences*, 72:425–443, 2006.

[38] R. Ravi, R. Sundaram, M. V. Marathe, D. J. Rosenkrantz, and S. S. Ravi. Spanning trees - short or small. *SIAM J. Discrete Math.*, 9(2):178–200, 1996.

[39] P. Slavík. Improved performance of the greedy algorithm for partial cover. *Information Processing Letters*, 64:251–254, 1997.

[40] D. D. Sleator and R. E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.

[41] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2001.