

Fully dynamic clustering and diversity maximization in doubling metrics

Paolo Pellizzoni^{1,2}, Andrea Pietracaprina², and Geppino Pucci²

¹ Max Planck Institute of Biochemistry, Germany

² Dept. of Information Engineering, University of Padova, Italy

Abstract. We present deterministic approximation algorithms for some variants of center-based clustering and related problems in the fully dynamic setting, where the pointset evolves through an arbitrary sequence of insertions and deletions. Specifically, we target the following problems: k -center (with and without outliers), matroid-center, and diversity maximization. All algorithms employ a coresets-based strategy and rely on the same data structure, a suitably augmented cover tree, which can thus serve queries for different problems at the same time. For all of the aforementioned problems, our algorithms yield approximations comparable to those achievable in the off-line setting. For spaces of bounded doubling dimension, the running times are dramatically smaller than those that would be required to compute solutions on the entire pointset from scratch after each update. To the best of our knowledge, ours are the first solutions for the matroid-center and diversity maximization problems in the fully dynamic setting.

Keywords: dynamic algorithms · k -center · outliers · matroid center · diversity maximization · cover tree.

1 Introduction

Clustering, that is the task of partitioning a set of points according to some similarity metric, is one of the fundamental primitives in unsupervised learning and data mining, with applications in several fields [27]. A commonly used formulation is the k -center problem, where, given a set of points from a metric space and a parameter k , one seeks to select k points as cluster centers so that the maximum distance from any point to its closest center is minimized. Since finding the optimal solution is known to be NP-hard [20], in practice one has to settle for approximate solutions.

Several important variants of the k -center problem have been intensely studied over the years. Since the k -center's objective function involves a maximum, the optimal solution may be heavily affected by *outlier* points, which are markedly distant from all the other ones. A robust formulation of the problem, referred to as k -center with z outliers, has been introduced in [15], where the objective function disregards the z most distant points from the centers. Another popular variant of k -center is the *matroid center* problem [16], which, given a set

of points and a matroid defined on it, aims at finding a set of centers forming an independent set of the matroid, minimizing the maximum distance of any point from the closest center. The matroid can be employed to model a wide range of constraints on the solution (most notably, *fairness* [17]). As for k -center, matroid center admits a robust formulation with z outliers [16]. Finally, a problem closely related to k -center is *diversity maximization* [1], whose goal is somehow dual with respect to the one of k -center, as the k points to be selected should maximize a suitable notion of diversity among them. As for k -center, a variant of diversity maximization exists where the diverse point selection is subject to a further matroid constraint [5, 7].

Conceivably, in practical scenarios the inputs to the above problems are not static, but evolve with time, hence it is important to develop algorithms that can maintain good solutions under insertion and deletion of arbitrary points. Indeed, in recent years there has been a surge in research efforts in the development of *fully dynamic* clustering algorithms, whose main focus is on the efficient handling of arbitrary insertions and deletions [2, 12, 13, 21].

In this paper, we tackle all of the aforementioned problems in the fully dynamic setting, targeting solutions for general metric spaces of bounded doubling dimension, which is a widely used notion of dimensionality for general metrics.

1.1 Related work

In the offline setting, k -center can be approximated within a factor 2 in polynomial time, but there is no polynomial-time $(2 - \epsilon)$ -approximation unless $P=NP$ [20]. For the robust formulation with z outliers, there exists a simple, combinatorial 3-approximation algorithm [15], which has been extended to weighted pointsets in [6]. More complex LP-based 2-approximate algorithms have been developed [11, 23], but it is not clear whether they can be employed in the weighted case. For what concerns the matroid-center problem, [16] provides a 3-approximate algorithm for the standard formulation and a 7-approximate algorithm for the robust formulation, which has been improved to a 3-approximation in [23]. The diversity maximization problem admits several instantiations, depending on the specific diversity function embodied in its objective, which are all NP-hard but admit polynomial-time $O(1)$ -approximation algorithms (e.g., see [1, 7] and references therein).

In [12], the authors developed the first fully dynamic k -center algorithm, which is able to return $(2 + \epsilon)$ -approximate solutions under arbitrary insertions and deletions of a non-adaptive adversary in general metrics. The algorithm is randomized and has an amortized update time of $O(k^2 \epsilon^{-1} \log \Delta)$, where Δ denotes the *aspect ratio* of the input pointset, namely the ratio between the largest and the smallest distance between any two points. This approach has been recently improved in [2], where an algorithm with expected amortized $O((k + \log n) \epsilon^{-1} \log \Delta \log n)$ update time is presented, where n is the maximum number of points at any time. We remark that the data structures used in these works for storing the dynamically changing pointset, must be statically configured to deal with *fixed* values of k and ϵ , and answering queries for

different clustering granularities and/or accuracies would, in principle, require building the data structures from scratch. Also, both works use data structures whose size is superlinear in the pointset size by at least a factor $O(\log \Delta/\epsilon)$. Recently, [13] presented a randomized fully dynamic algorithm for k -center with z outliers that returns a bicriteria $(14 + \epsilon)$ -approximation solution when discarding at most $(1 + \lambda)z$ outliers. Again, their data structure works for fixed k and ϵ and requires superlinear size by a factor $O(\log \Delta/\epsilon)$.

In [21], the authors propose a fully dynamic k -center algorithm for metric spaces of bounded doubling dimension D based on the *navigating net* data structure [26], which affords insertions and deletions in $O((1/\epsilon)^{O(D)} \log \Delta \log \log \Delta \log \epsilon^{-1})$ time. While their approach allows clustering queries for arbitrary values of k , the data structure is built for a specific value of the accuracy parameter ϵ , and must be rebuilt from scratch if a different accuracy ϵ' is sought for. Moreover, the data structure requires superlinear space, by a factor $O((1/\epsilon)^{O(D)} \log \Delta \log \log \Delta \log \epsilon^{-1})$. Finally, k -center clustering has been studied in semi-dynamic frameworks, such as the insertion-only [6, 10, 14, 24] or the sliding window [4, 18, 29, 30] settings.

1.2 Our contributions

We present approximation algorithms for the problems introduced before, in the fully dynamic setting. All algorithms are deterministic and crucially rely on a core data structure aimed at maintaining the necessary information for solving the problem at hand. At any point in time, when a solution to any of the problems is required with a given accuracy, a small subset of representative points, dubbed *coreset*, is extracted from the data structure, and a sequential algorithm is run on such a coreset to compute the solution. This allows for fast execution times, which are independent of the current number of points stored in the tree. Our specific contributions are the following.

1. We define the *augmented cover tree* data structure to maintain the current set of points, by extending and improving the cover tree of [3] in several ways: (i) we store richer information at each node regarding the subset of points stored in its subtree, such as its cardinality and, for matroid center, a maximal independent set of the submatroid induced by the subset; (ii) we devise iterative algorithms to handle insertion and deletions maintaining the aforementioned richer information, which also afford simpler correctness proofs and simpler implementations compared to the original recursive ones [3]; (iii) we make the complexity analysis of the update operations parametric in the doubling dimension D and the aspect ratio Δ (Theorems 1 and 2), fixing some flaws in the original analysis, as pointed out in [19].
2. We devise a fully dynamic $(2 + \epsilon)$ -approximate algorithm for k -center. Unlike all previous works, our data structure allows to query for solutions for *arbitrary values* of k and ϵ . Compared to [21], our data structure has smaller (linear) size and can handle insertions and deletions at a lower asymptotic cost. Also, our time to compute the solution has only a linear dependency on k (Theorem 3).

3. We devise a fully dynamic $(3 + \epsilon)$ approximation algorithm for k -center with z outliers, which allows to choose k , ϵ , and z at query time. We remark that the only previously available algorithm for general metrics can only return a $(14 + \epsilon)$ bi-criteria solution (i.e. with an additional slackness on the number of outliers), and requires to fix k , z and ϵ beforehand (Theorem 4).
4. We present the first fully dynamic algorithm for matroid center. Our algorithm returns a $(3 + \epsilon)$ approximate solution (Theorem 5).
5. We present the first fully dynamic algorithms for diversity maximization. Our algorithms return $(\alpha_{\text{div}} + \epsilon)$ approximate solutions, where α_{div} is the best sequential approximation factor (Theorem 6).

An important feature of our algorithms is that they are oblivious to D , in the sense that the value D only influences the analysis, but must not be known by the algorithms. This is a very desirable feature, since, in practice, this value is difficult to estimate. Moreover, our algorithms are completely deterministic, while most previously available algorithms make use of randomization.

The rest of the paper is organized as follows. Section 2 provides the formal definitions of the problems and some technical facts used in the analysis. Section 3 is dedicated to describing the novel augmented cover tree, while Section 4 details how to maintain such data structure efficiently. In Section 5, we present the fully dynamic algorithms for the problems introduced before. Section 6 concludes the paper with some final remarks.

Due to space constraints, pseudocodes and some proofs are omitted in this extended abstract but can be found in [31].

2 Preliminaries

This section formally defines the problems studied in this paper, and states some important technical facts.

(Robust) k -center problem. Consider a metric space (U, dist) and a set $S \subseteq U$ of n points. For any $p \in U$ and any subset $C \subseteq S$, we use the notation $\text{dist}(p, C) = \min_{q \in C} \text{dist}(p, q)$, and define the *radius of C with respect to S* as

$$r_C(S) = \max_{p \in S} \text{dist}(p, C).$$

For a positive integer $k < n$, the k -center problem requires to find a subset $C \subseteq S$ of size at most k which minimizes $r_C(S)$. The points of the solution C are referred to as *centers*. Note that C induces a partition of S into $|C|$ clusters, by assigning each point to its closest center (with ties broken arbitrarily). We denote the radius of the optimal solution by $r_k^*(S)$. The seminal work by Gonzalez [20] presents a greedy algorithm that returns a 2-approximate solutions to k center in $O(nk)$ -time.

The algorithms presented in this paper crucially rely on confining the computation of the solution on a succinct *coreset* T , efficiently extracted from the (possibly large) input S , which contains a close enough “representative” for each point in S . The quality of a coreset T is captured by the following definition.

Definition 1. Given a pointset S and a value $\epsilon > 0$, a subset $T \subseteq S$ is an (ϵ, k) -coreset for S (w.r.t. the k -center problem) if $r_T(S) \leq \epsilon r_k^*(S)$.

In real world applications, large datasets often include noisy points which, if very distant from all other points, may severely distort the optimal center selection. To handle these scenarios, the following robust formulation of the k -center problem has been introduced [15]. For positive $k, z < n$, the k -center problem with z outliers (a.k.a. (k, z) -center) requires to find a subset $C \subseteq S$ of size k minimizing $r_C(S - Z_C)$, where Z_C is the set of z points in S with the largest distances from C , which are regarded as outliers. We denote the radius of the optimal solution of this problem by $r_{k,z}^*(S)$. Observe that the (k, z) -center problem reduces to the k -center problem for $z = 0$. Also, it is straightforward to argue that

$$r_{k+z}^*(S) \leq r_{k,z}^*(S). \quad (1)$$

A well known sequential 3-approximation algorithm for the k -center problem with z outliers, which runs in $O(n^2 \log n)$ time was devised in [15]. In this work, we will make use of a more general formulation of the problem, referred to as *weighted (k, z) -center*, where each point $p \in S$ carries a positive integer weight $w(p)$, and the desired set C of k centers must minimize $r_C(S - Z_C)$, where Z_C is the set of points with the largest distances from C , of maximum cardinality and aggregate weight at most z .

Matroid center problem. Another variant of the k -center problem requires the solution C to satisfy an additional constraint, specified through a matroid. A *matroid* [28] on a pointset S is a pair $M = (S, I)$, where I is a family of subsets of S , called *independent sets*, satisfying the following properties: (i) the empty set is independent; (ii) every subset of an independent set is independent (*hereditary property*); and (iii) if $A, B \in I$ and $|A| > |B|$, then there exists $x \in A \setminus B$ such that $B \cup \{x\} \in I$ (*augmentation property*). An independent set is *maximal* if it is not properly contained in another independent set. Given a matroid $M = (S, I)$, the *matroid center problem* on M requires to determine an independent set $C \in I$ minimizing the radius $r_C(S)$. We let $r^*(M) = \min_{C \in I} r_C(S)$ to denote the radius of the optimal solution. The augmentation property ensures that all maximal independent sets of M have the same size, which is referred to as *rank* of M , denoted by $\text{rank}(M)$. It is easy to argue that $r_{\text{rank}(M)}^*(S) \leq r^*(M)$, since each solution to the matroid center problem is also a solution to the k -center problem with $k = \text{rank}(M)$. The definition of matroid implies that any subset $S' \subseteq S$, induces a matroid $M' = (S', I')$, where $I' \subseteq I$ is the restriction of I to subsets of S' . The following fact, which will be exploited in the derivation of our results, is a consequence of the *extended augmentation property* proved in [8, Lemma 2.1] (see also [24]).

Fact 1 Let $M = (S, I)$ be a matroid, and let $S_1, \dots, S_h$ be a partition of S into h disjoint subsets. If $A_1 \subseteq S_1, \dots, A_h \subseteq S_h$ are maximal independent sets of the submatroids $M_1 = (S_1, I_1), \dots, M_h = (S_h, I_h)$, then $\cup_{i=1}^h A_i$ contains a maximal independent set of M .

Table 1. Diversity functions considered in this paper. $w(\text{MST}(X))$ (resp., $w(\text{TSP}(X))$) denotes the minimum weight of a spanning tree (resp., Hamiltonian cycle) of the complete graph induced by the points of X and their pairwise distances.

Name	Diversity measure $\text{div}(X)$	Sequential approx. α_{div}
remote-edge	$\min_{p,q \in X} d(p, q)$	2
remote-clique	$\sum_{p,q \in X} d(p, q)$	2
remote-star	$\min_{c \in X} \sum_{q \in X \setminus \{c\}} d(c, q)$	2
remote-bipartition	$\min_{Q \subset X, Q = \lfloor X /2 \rfloor} \sum_{q \in Q, z \in X \setminus Q} d(q, z)$	3
remote-tree	$w(\text{MST}(X))$	4
remote-cycle	$w(\text{TSP}(X))$	3

As in previous works [1, 7, 24], in the paper we will assume that a constant-time oracle is available to check independence of any subset of S . A combinatorial 3-approximation algorithm for the matroid center problem in general metrics is presented in [16], which runs in time polynomial in $|S|$ and $\text{rank}(M)$.

An important instantiation of the problem is based on the *partition matroid* $M_P = (S, I_P)$, where each point in S is associated to one of $m \leq k$ of categories, and I_P consists of all subsets with at most k_i points of the i -th category, with $\sum_{i=1}^m k_i = k$. For instance, this matroid can be employed to model fairness constraints [17, 25].

Diversity maximization Let $\text{div} : 2^S \rightarrow \mathbb{R}$ be a *diversity function* that maps any subset $X \subset S$ to some non-negative real number. For a specific diversity function div and a positive integer $k \leq n$, the goal of the *diversity maximization problem* is to find a set $C \subseteq S$ of size k that maximizes $\text{div}(C)$. We denote the optimal value of the objective function as $\text{div}_k^*(S) = \max_{C \subseteq S, |C|=k} \text{div}(C)$. In this paper, we will focus on several variants of the problem based on different diversity functions amply studied in the previous literature, which are reported in Table 1. All variants are NP-hard, and Table 1 lists the best known approximation ratios attainable in polynomial time (see [1, 10] and references therein).

Doubling dimension We will relate the performance of our algorithms to the dimensionality of the data which, for a general metric space (U, dist) , can be captured by the notion of doubling dimension, reviewed below. For any $p \in U$ and $r > 0$, the *ball of radius r centered at p* , denoted as $B(p, r)$, is the subset of all points of U at distance at most r from p . The *doubling dimension* of U is the minimum value D such that, for all $p \in U$, any ball $B(p, r)$ is contained in the union of at most 2^D balls of radius $r/2$ centered at points of U . The notion of doubling dimension has been used extensively for a variety of applications (e.g., see [9, 10, 22, 29] and references therein). The following fact is a simple consequence of the definition.

Fact 2 *Let X be a set of points from a metric space of doubling dimension D , and let $Y \subseteq X$ be such that any two distinct points $a, b \in Y$ have pairwise*

distance $\text{dist}(a, b) > r$. Then for every $R \geq r$ and any point $p \in X$, we have $|B(p, R) \cap Y| \leq 2^{\lceil \log_2(2R/r) \rceil \cdot D} \leq (4R/r)^D$. If $R/r = 2^i$, then the bound can be lowered to $|B(p, R) \cap Y| \leq (2R/r)^D$.

3 Augmented Cover Trees

Let S be a set of n points from a metric space (U, dist) of doubling dimension D . Our algorithms employ an augmented version of the cover tree data structure of [3]. For completeness, we first review the original data structure and then discuss the augmentation, providing simpler update algorithms and a new correct analysis. Conceptually, a *cover tree* T for S is an infinite tree, where each node corresponds to a single point of S , while each point of S is associated to one or more nodes. The levels of the tree are indexed by integers, decreasing from the root towards the leaves. For $\ell \in (-\infty, +\infty)$, we let T_ℓ to be the set of nodes of level ℓ , and let $\text{pts}(T_\ell)$ be the points associated with the nodes of T_ℓ , which are required to be distinct. For each node $u \in T$ we maintain: its associated point ($u.\text{point}$); a pointer to its parent ($u.\text{parent}$); the list of pointers to its children ($u.\text{children}$); and its level in T ($u.\text{level}$). For brevity, in what follows, for any two nodes $u, v \in T$ (resp., any point $p \in U$ and node $u \in T$), we will use $\text{dist}(u, v)$ (resp., $\text{dist}(p, u)$) to denote $\text{dist}(u.\text{point}, v.\text{point})$ (resp., $\text{dist}(p, u.\text{point})$). For each level ℓ , the set T_ℓ must satisfy the following three properties:

1. $\text{pts}(T_\ell) \subseteq \text{pts}(T_{\ell-1})$;
2. for each $u \in T_\ell$, $\text{dist}(u, u.\text{parent}) \leq 2^{\ell+1}$;
3. for all $u, v \in T_\ell$, $\text{dist}(u, v) > 2^\ell$.

For every $p \in S$, let $\ell(p)$ denote the largest index such that $p \in \text{pts}(T_{\ell(p)})$. The definition implies that for every $\ell < \ell(p)$, $p \in \text{pts}(T_\ell)$, and the node u in T_ℓ with $u.\text{point} = p$ is a *self-child* of itself, in the sense that $u.\text{parent}.\text{point} = p$, since for every other $v \in T_{\ell+1}$, with $v.\text{point} \neq p$, $\text{dist}(u, v) > 2^{\ell+1}$. Let $d_{\min}$ and $d_{\max}$ denote, respectively, the minimum and maximum distances between two points of S , and define $\Delta = d_{\max}/d_{\min}$ as the *aspect ratio* of S . It can be easily seen that for every $\ell < \log_2 d_{\min}$, $\text{pts}(T_\ell) = S$, and for every $\ell \geq \log_2 d_{\max}$, $|\text{pts}(T_\ell)| = 1$. We define $\ell_{\min}$ (resp., $\ell_{\max}$) as the largest (resp., smallest) index such that $\text{pts}(T_\ell) = S$ (resp., $|\text{pts}(T_\ell)| = 1$), and note that every node u in a level T_j with $j > \ell_{\max}$ or $j \leq \ell_{\min}$ has only the self-child in $u.\text{children}$. Therefore, we will consider only the portion of the tree constituted by the levels T_ℓ with $\ell \in [\ell_{\min}, \ell_{\max}]$ and will regard the unique node $r \in T_{\ell_{\max}}$ as the root of the tree. The above observations imply that the number of levels in this portion of the tree is $O(\log \Delta)$.

Update operations on a cover tree crucially rely on the notion of *cover set* [3]. For any point $p \in U$, and for every index $\ell \leq \ell_{\max}$ the *cover set* (for p at level ℓ) $Q_\ell^p \subseteq T_\ell$ is defined inductively as follows:

$$\begin{aligned} Q_{\ell_{\max}}^p &= \{r\} \\ Q_\ell^p &= \{u \in T_\ell : u.\text{parent} \in Q_{\ell+1}^p \wedge \text{dist}(u, p) \leq 2^{\ell+1}\} \quad \text{for } \ell < \ell_{\max} \end{aligned} \tag{2}$$

A simple inductive argument proves that for any node $u \in T_\ell - Q_\ell^p$, $\text{dist}(p, u) > 2^{\ell+1}$. Thus, intuitively, Q_ℓ^p contains all points of T_ℓ somewhat “close” to p , at the scale prescribed by the level.

We now relate the size of the cover sets to the doubling dimension D of (U, dist) . We wish to remark that, when the doubling dimension D_S of the subspace (S, dist) is lower than D , e.g. if the current points belong to a lower-dimensional manifold of U , D_S can replace D in the subsequent analyses, yielding tighter results.

Lemma 1. *For every point $p \in U$ and $\ell \leq \ell_{\max}$, we have that $|Q_\ell^p| \leq 4^D$ and $\sum_{u \in Q_\ell^p} |u.\text{children}| \leq 12^D$.*

Proof. For each $u_1, u_2 \in T_\ell$, we have that $\text{dist}(u_1, u_2) > 2^\ell$. Also, $\text{pts}(Q_\ell^p) \subseteq B(p, 2^{\ell+1}) \cap T_\ell$. Then, by Fact 2, it follows that $|Q_\ell^p| \leq (2 \cdot 2^{\ell+1}/2^\ell)^D = 4^D$. Moreover, for each $u \in Q_\ell^p$ and each $u' \in u.\text{children}$, we have that $\text{dist}(u', p) \leq \text{dist}(u', u) + \text{dist}(u, p) \leq 2^\ell + 2^{\ell+1} = 3 \cdot 2^\ell$, whence $u'.\text{point} \in B(p, 3 \cdot 2^\ell) \cap \text{pts}(T_{\ell-1})$. Then, again by Fact 2, it follows that $|\{u' \in u.\text{children} \text{ s.t. } u \in Q_\ell^p\}| \leq (2 \cdot 3 \cdot 2^\ell / 2^{\ell-1})^D = 12^D$.

We maintain a pointer to the root r of T , and recall that $r.\text{level} = \ell_{\max}$. Note that the naive representation of T , referred to as *implicit representation* in [3], would require $O(n \log \Delta)$ space. In order to save space, a more compact representation of T , referred to as *explicit representation* in [3], is used, where chains of 1-child nodes, which correspond to instances of the same point, are represented solely by the first node of the chain, which inherits the children of the last node of the chain. It is easy to argue that this compact representation takes only $O(n)$ space. Given a point $p \in U$, the cover sets Q_ℓ^p for $\ell \leq \ell_{\max}$ can be constructed in a top-down fashion from the explicit representation of T by simply recreating the contracted chains of implicit nodes. By Lemma 1, Q_ℓ^p can be constructed from $Q_{\ell+1}^p$ in $O(12^D)$ time.

Augmenting the basic structure. We augment the cover tree data structure so to maintain at its nodes two additional data fields, which will be exploited in the target applications presented later. Suppose that a matroid $M = (U, I)$ is defined over the universe U . An *augmented cover tree* T for S (with respect to M) is a cover tree such that each node $u \in U$ stores the following two additional fields: a positive weight $u.\text{weight} = |S_u|$, where S_u is the subset of points of S associated with nodes in the subtree rooted at u , and a set of points $u.\text{mis}$, which is a maximal independent set of the submatroid $M_u = (S_u, I_u)$ spanned by the subsets of S_u . The size of T becomes $O(n \cdot \text{rank}(M))$, where $\text{rank}(M)$ denotes the size of a maximum independent set of S . For the applications where the matroid information is not needed, the fields $u.\text{mis}$ will be always set to null (as if $M = (U, \emptyset)$), and, in this case, the size of T will be $O(n)$.

4 Dynamic maintenance of augmented cover trees

Let T be an augmented cover tree for a set S of n points. In this section, we show how to update T efficiently when a point p is added to or deleted from S .

Insertion. Let p be a new point to be inserted in T . The insertion of p is accomplished as follows. First, if p is very far from the root r , namely $\text{dist}(p, r) > 2^{\ell_{\max}}$, then both $\ell_{\max}$ and $r.\text{level}$ are increased to $\lfloor \log_2 \text{dist}(p, r) \rfloor$. Then, an explicit node u is created with $u.\text{point} = p$, $u.\text{weight} = 1$ and $u.\text{mis} = \{p\}$. In order to determine the level $\ell(p)$ where u must be placed, all cover sets Q_ℓ^p are computed, as described above, for every $\ell \in [\bar{\ell}, \ell_{\max}]$, where $\bar{\ell}$ is the largest index in $(-\infty, \ell_{\max}]$ such that $Q_{\bar{\ell}}^p = \emptyset$. Note that such empty cover set must exist and it is easy to see that $\bar{\ell} \geq \lceil \log_2 \text{dist}(p, S) \rceil - 2$. Then, $u.\text{level}$ is set to the smallest index $\ell(p) \geq \bar{\ell}$ such that $\text{dist}(p, \text{pts}(Q_{\ell(p)}^p)) > 2^{\ell(p)}$ and $\text{dist}(p, \text{pts}(Q_{\ell(p)+1}^p)) \leq 2^{\ell(p)+1}$. At this point, an arbitrary node $v \in Q_{\ell(p)+1}^p$ such that $\text{dist}(p, v) \leq 2^{\ell(p)+1}$ (which must exist for sure) is determined. Let $q = v.\text{point}$. If v has no explicit self-child at level $\ell(p)$, a new node w with $w.\text{point} = q$, $w.\text{level} = \ell(p)$, and $w.\text{children} = v.\text{children}$, is created, and $v.\text{children}$ is set to $\{u, w\}$. If instead such an explicit self-child w of v exists at level $\ell(p)$, then u is simply added as a further child of v . Finally, the path from the newly added node u to r is traversed, and for every ancestor v of u , $v.\text{weight}$ is increased by 1 and p is added to the independent set $v.\text{mis}$, if $v.\text{mis} \cup \{p\}$ is still an independent set.

In [31], we provide a fully detailed pseudocode for the insertion procedure.

Theorem 1. *Let T be an augmented cover tree for a set S of n points, with respect to a matroid $M = (U, I)$. The insertion algorithm described above yields an augmented cover tree for $S \cup \{p\}$ in time $O(12^D \log \Delta)$ where D is the doubling dimension of the metric space and Δ is the aspect ratio of S .*

Proof. It is easy to see that the insertion algorithm enforces, for every level ℓ , Properties 1, 2, 3 of the definition of cover tree, restricted to the nodes in Q_ℓ^p plus the new node created for p (for $\ell = \ell(p)$). These properties immediately extend to the entire level ℓ , since, as observed before, $\text{dist}(p, u) > 2^{\ell+1}$, for every $u \in T_\ell - Q_\ell^p$. For what concerns the update of the weight and mis fields, correctness is trivially argued for the weight fields, while Fact 1 ensures correctness of the updates of the mis fields. can be argued as for the insertion algorithm. The complexity bounds follow by observing that there are $O(\log \Delta)$ levels in the explicit representation of T and that, at each such level ℓ , the algorithm performs work linear in the number of children of Q_ℓ^p , which are at most 12^D , by Lemma 1.

Deletion. Let $p \in S$ be the point to be removed. We assume that p is not the only point in S , otherwise the removal is trivial. The deletion of p is accomplished as follows. In the first, top-down phase, all cover sets Q_ℓ^p are computed, for every $\ell \in [\bar{\ell}, \ell_{\max}]$, where $\bar{\ell} \leq \ell_{\max}$ is the level of the leaf node corresponding to p in the explicit tree. Also, a list $R_{\bar{\ell}}$ of explicit nodes at level $\bar{\ell}$ to be relocated is created and initialized to the empty list. In the second, bottom-up phase, the following operations are performed iteratively, for every $\ell = \bar{\ell}, \bar{\ell}+1, \dots, \ell_{\max}-1$.

- If Q_ℓ^p contains a node u with $u.\text{point} = p$ and $u.\text{level} = \ell$, the following additional operations are performed. Let v be the parent of u , and observe that all children of v must also be explicit nodes at level ℓ . If u is the self child of v (i.e., $v.\text{point} = u.\text{point} = p$) u is removed from T , u 's siblings are detached from v and added to R_ℓ (v will be later removed at iteration $v.\text{level}$). If instead u is not the self child of v , but it is the only child of v besides the self-child, u is removed from T and v and its self-child are merged together in the explicit tree.
- An empty list $R_{\ell+1}$ is created. Then, R_ℓ is scanned sequentially, and, for every $w \in R_\ell$, a node $w' \in Q_{\ell+1}^p \cup R_{\ell+1}$ is searched for such that $d(w, w') \leq 2^{\ell+1}$. If no such node exists, then w is added to $R_{\ell+1}$, raising $w.\text{level}$ to $\ell+1$. Otherwise, if w' is found, it becomes parent of w as follows. If w' is internal and its children are at level ℓ , w is simply added as a further child. Otherwise, a new explicit node z , is created with $z.\text{point} = w'.point$, $z.\text{level} = \ell$, and $z.children = w'.children$, and $w'.children$ is set to $\{z, w\}$.
- For all nodes in $w \in Q_{\ell+1}^p \cup R_{\ell+1}$ with $w.\text{level} = \ell+1$, their $.weight$ and $.mis$ fields are updated based on the values of the corresponding fields of their children. The update of the $.weight$ fields is straightforward, while, based on Fact 1, the update of the $.mis$ field of one such node w can be accomplished by computing a maximal independent set in the union of the elements of the $.mis$ fields of w 's children.

Once the above operations are performed, a slightly more complex iteration is required for $\ell = \ell_{\max}$, since it might be necessary to modify $\ell_{\max}$ and/or create a new root, in case $p = r.\text{point}$. A fully detailed pseudocode for the deletion procedure is provided in [31].

Theorem 2. *Let T be an augmented cover tree for a set S of n points, with respect to a matroid $M = (U, I)$. The deletion algorithm described above yields an augmented cover tree for $S - \{p\}$ in time $O((16^D + 12^D \text{rank}(M)) \log \Delta)$ where D is the doubling dimension of the metric space and Δ is the aspect ratio of S .*

Proof. It is easy to see that the bottom-up phase of the deletion algorithm enforces, for every level ℓ , Properties 1, 2, 3 of the definition of cover tree, restricted to the nodes in $Q_\ell^p \cup R_\ell$. As argued before, these properties immediately extend to the entire level ℓ . Finally, correctness of the update of the $.weight$ and $.mis$ fields can be argued as for the insertion algorithm. For what concerns the complexity bound, first observe that for every level ℓ , the nodes in $Q_\ell^p \cup R_\ell$ represent the new coversets $\hat{Q}_\ell^p$ associated to p after its deletion from T , thus Lemma 1 holds. As a consequence, the work needed to process the nodes in R_ℓ is $\Theta(|R_\ell|(|Q_{\ell+1}^p \cup R_{\ell+1}|)) = O(16^D)$, while, by Fact 1, recreating the $.weight$ and $.mis$ fields for all nodes in $Q_{\ell+1}^p \cup R_{\ell+1}$ is upper bounded by the number of their children multiplied by $(1 + \text{rank}(M))$. The final bound follows by observing that there are $O(\log \Delta)$ levels in the explicit representation of T .

5 Extracting solutions from the augmented cover tree

We show how to employ the augmented cover tree presented before to extract accurate solutions to the various problems introduced in Section 2. For all these problems, we rely on the extraction from the cover tree of a small (ϵ, k) -coreset (see Definition 1), for suitable values of ϵ and k .

Let T be an augmented cover tree for a set S of n points from a metric space of doubling dimension D . Given ϵ and k , an (ϵ, k) -coreset for S can be constructed as follows. Let $T_{\ell(k)}$ be the level of largest index (in the implicit representation of T) such that $|T_{\ell(k)}| \leq k$ and $|T_{\ell(k)-1}| > k$. Then, define

$$\ell^*(\epsilon, k) = \max\{\ell_{\min}, \ell(k) - \lceil \log_2(8/\epsilon) \rceil\}. \quad (3)$$

(For ease of notation, in what follows we shorthand $\ell^*(\epsilon, k)$ with ℓ^* whenever the parameters are clear from the context.) We have:

Lemma 2. *The set of points $\text{pts}(T_{\ell^*})$ is an (ϵ, k) -coreset for S of size at most $k(64/\epsilon)^D$ and can be constructed in time $O(k((64/\epsilon)^D + \log \Delta))$.*

Proof. We first show that $\text{pts}(T_{\ell^*})$ is an (ϵ, k) -coreset for S . If $\ell^* = \ell_{\min}$, we have $\text{pts}(T_{\ell^*}) = S$, so the property is trivially true. Suppose that $\ell^* = \ell(k) - \lceil \log_2(8/\epsilon) \rceil > \ell_{\min}$ and consider an arbitrary point $p \in S$. There must exist some node $u \in T_{\ell_{\min}}$ such that $p = u.\text{point}$. Letting v be the ancestor of u in T_{ℓ^*} , by the cover tree properties, we know that $\text{dist}(v.\text{point}, p) \leq 2^{\ell^*+1} \leq (\epsilon/4)2^{\ell(k)}$. Also, since all pairwise distances among points of $\text{pts}(T_{\ell(k)-1})$ are greater than $2^{\ell(k)-1}$ and $|T_{\ell(k)-1}| \geq k+1$, there must be two points $q, q' \in \text{pts}(T_{\ell(k)-1})$ which belong to the same cluster in the optimal k -center clustering of S . Therefore, $2^{\ell(k)-1} < \text{dist}(q, q') \leq 2r_k^*(S)$, which implies that $2^{\ell(k)} \leq 4r_k^*(S)$. Putting it all together, we have that for any $p \in S$, $\text{dist}(p, \text{pts}(T_{\ell^*})) \leq \epsilon r_k^*(S)$. Let us now bound the size of T_{ℓ^*} . By construction $|T_{\ell(k)}| \leq k$, and we observe that T_{ℓ^*} can be partitioned into $|T_{\ell(k)}|$ subsets $T_{\ell^*}^u$, for every $u \in T_{\ell(k)}$, where $T_{\ell^*}^u$ is the set of descendants of u in T_{ℓ^*} . The definition of cover tree implies that for each $u \in T_{\ell(k)}$ and $v \in T_{\ell^*}^u$, $\text{dist}(u, v) \leq 2^{\ell(k)+1}$. Moreover, since the pairwise distance between points of $\text{pts}(T_{\ell^*}^u)$ is greater than 2^{ℓ^*} , by applying Fact 2 with $Y = \text{pts}(T_{\ell^*}^u)$, $R = 2^{\ell(k)+1}$ and $r = 2^{\ell^*}$, we obtain that $|T_{\ell^*}^u| \leq 2^{(\lceil \log_2(8/\epsilon) \rceil + 2) \cdot D} \leq (64/\epsilon)^D$, and the bound on $|T_{\ell^*}|$ follows. T_{ℓ^*} can be constructed on the explicit tree through a simple level-by-level visit up to level ℓ^* , which can be easily determined from $\ell(k)$ and the fact that $\ell_{\min}$ is the largest level for which all nodes in the explicit representation of T are leaves. The construction time is linear in $\sum_{\ell=\ell^*}^{\ell_{\max}} |T_{\ell}| = \sum_{\ell=\ell^*}^{\ell(k)-1} |T_{\ell}| + \sum_{\ell=\ell(k)}^{\ell_{\max}} |T_{\ell}|$. The second summation is clearly upper bounded by $k \log \Delta$, while, using again Fact 2 it is easy to argue that $|T_{\ell}| \leq k \cdot 2^{(\ell(k)+2-\ell) \cdot D}$, for every $\ell^* \leq \ell \leq \ell(k) - 1$, whence the first sum is $O(k(64/\epsilon)^D)$.

5.1 Solving k -center

Suppose that an augmented cover tree T for S is available. We can compute an $O(2 + O(\epsilon))$ -approximate solution C to k -center on S as follows. First, we

extract the coreset $Q = \text{pts}(T_{\ell^*})$, where $\ell^* = \ell^*(\epsilon, k)$ is the index defined in Equation 3, and then run a sequential algorithm for k -center on Q . To do so, we could use Gonzalez's 2-approximation algorithm. However, this would contribute an $O(k|Q|)$ term to the running time, which, based on the size bound stated in Lemma 2, would yield a quadratic dependency on k . The asymptotic dependency on k can be lowered by computing the solution through an adaptation of the techniques presented in [21], as explained below. Let us define a generalization of the cover tree data structure, dubbed (α, β) -cover tree, where the three properties that each level ℓ must satisfy are rephrased as follows:

1. $\text{pts}(T_\ell) \subseteq \text{pts}(T_{\ell-1})$;
2. for each $u \in T_\ell$, $\text{dist}(u, u.\text{parent}) \leq \beta \cdot \alpha^{\ell+1}$;
3. for all $u, v \in T_\ell$, $\text{dist}(u, v) > \beta \cdot \alpha^\ell$.

By adapting the insertion procedure and its analysis, it is easily seen that the insertion of a new point in the data structure can be supported in $O(12^D \cdot \log_\alpha \Delta)$ time. For a given integer parameter m , we construct m generalized cover trees for the coreset Q , namely an $(\alpha, \alpha^{i/m})$ -cover tree $T^{(i)}$ for every $1 \leq i \leq m$. Each cover tree is constructed by inserting one point of Q at a time. Let ℓ_i be the smallest index such that level $T_{\ell_i}^{(i)}$ in $T^{(i)}$ has at most k nodes. The returned solution C is the set $\text{pts}(T_{\ell_i}^{(i)})$ such that $T_{\ell_i}^{(i)}$ minimizes $\alpha^{\ell_i + i/m}$. By selecting $\alpha = 2/\epsilon$ and $m = O(\epsilon^{-1} \ln \epsilon^{-1})$, and by using the argument of [21], it can be shown that C is a $(2 + O(\epsilon))$ -approximation for k -center on Q . The following theorem is a consequence of the above discussion, Lemma 2, and the definition of (ϵ, k) -coreset.

Theorem 3. *Given an augmented cover tree T for S , the above procedure returns a $(2 + O(\epsilon))$ -approximation C to the k -center problem for S , and can be implemented in time $O((k/\epsilon)(768/\epsilon)^D \log \Delta)$.*

Proof. By Lemma 2, Q is an (ϵ, k) -coreset for S . Let $C^* = \{c_1, c_2, \dots, c_k\}$ be an optimal solution for k -center on S . The coreset property of Q ensures that for each c_i there is a point $c'_i \in Q$ such that $d(c_i, c'_i) \leq \epsilon r_k^*(S)$. This implies that the set $C' = \{c'_1, c'_2, \dots, c'_k\}$ is a solution to k -center on Q with $r_{C'}(Q) \leq (1+\epsilon)r_k^*(S)$, hence $r_k^*(Q) \leq (1+\epsilon)r_k^*(S)$. Suppose that the $(2+O(\epsilon))$ -approximation algorithm outlined above is used in Phase 2 to compute the solution C on Q . Then, $r_C(Q) \leq (2 + O(\epsilon))r_k^*(Q) \leq (2 + O(\epsilon))(1 + \epsilon)r_k^*(S) = (2 + O(\epsilon))r_k^*(S)$. By the coreset property and the triangle inequality, it follows that $r_C(S) \leq (2 + O(\epsilon))r_k^*$. For what concerns the running time, we have that the construction of the (ϵ, k) -coreset Q requires $O(k((64/\epsilon)^D + \log \Delta))$ time (see Lemma 2), while the running time of Phase 2 is dominated by the construction of the $m = O(\epsilon^{-1} \ln \epsilon^{-1})$ $(\alpha, \alpha^{p/m})$ -cover trees $T^{(p)}$, for $1 \leq p \leq m$, by successive insertions of the elements of Q . As observed above, an insertion takes $O(12^D \log_\alpha \Delta)$ time, hence the cost for constructing each $T^{(p)}$ is $O(|Q|12^D \log_\alpha \Delta)$. Since $\alpha = 2/\epsilon$ and $|Q| \leq k(64/\epsilon)^D$, the total cost is thus $O((k/\epsilon)(768/\epsilon)^D \log \Delta)$, which dominates over the cost of Phase 1.

5.2 Solving k -center with z outliers

For the k -center problem with z outliers, an approach similar to the one adopted for k -center can be employed. Let T be an augmented cover tree for S . We can compute a $(3 + O(\epsilon))$ -approximation to k -center with z outliers on S , by proceeding as follows. First, we extract the coreset $Q = \text{pts}(T_{\ell^*})$, where $\ell^* = \ell^*(k+z)$ is the index defined in Equation 3. Each point $q \in Q$ is associated to the weight $w_q = u.\text{weight}$, where $u \in T_{\ell^*}$ is such that $u.\text{point} = q$. Then, we extract the solution C from this weighted coreset Q using the techniques from [6]. We have:

Theorem 4. *Given an augmented cover tree T for S , the above procedure returns a $(3 + O(\epsilon))$ -approximation C to the k -center problem with z outliers for S , and can be implemented in time $O((k+z)^2(64/\epsilon)^{2D}(1/\epsilon)\log \Delta)$.*

Proof. By Lemma 2, $Q = \text{pts}(T_{\ell^*})$ is an $(\epsilon, k+z)$ -coreset for S . Consider a point $q \in Q$ and let $u \in T_{\ell^*}$ be such that $u.\text{point} = q$. It is easy to see that each of the $u.\text{weight} = w_q$ points associated with the descendants of u in T (forming set $S_u \subseteq S$) is at distance at most $\epsilon r_{k+z}^*(S)$ from q . Since the S_u 's form a partition of S , we have that each point of S can be associated to a point of Q at distance at most $\epsilon r_{k+z}^*(S) \leq r_{k,z}^*(S)$, referred to as its *proxy*. Suppose that algorithm OUTLIERSCLUSTER described in [6] is run on the weighted coreset Q with parameters k , r , and ϵ , where r is a guess of the optimal radius. The analysis in [6] shows that the algorithm returns two subsets $X, Q' \subseteq Q$ such that

- $|X| \leq k$
- For every $p \in S$ whose proxy is in $Q - Q'$, $\text{dist}(p, X) \leq \epsilon r_{k,z}^*(S) + (3 + 4\epsilon)r$;
- if $r \geq r_{k,z}^*(S)$, then $\sum_{q \in Q'} w_q \leq z$.

Then, we can repeatedly run OUTLIERSCLUSTER for $r = 2^{\ell_{\max}}/(1+\epsilon)^i$, for $i = 0, 1, \dots$, stopping at the smallest guess r which returns a pair (X, Q') where Q' has aggregate weight at most z and returning $C = X$ as the final solution. Therefore, the bound on the approximation ratio immediately follows from the properties of OUTLIERSCLUSTER. The running time is dominated by the repeated executions of OUTLIERSCLUSTER. Each execution of OUTLIERSCLUSTER can be performed in $O(|Q|^2 + k|Q|) = O((k+z)^2(64/\epsilon)^{2D})$ time. The bound on the running time follows by observing that $2^{\ell_{\max}}/r_{k,z}^*(S) = O(\Delta)$, whence $O(\log_{1+\epsilon} \Delta) = O((1/\epsilon)\log \Delta)$ executions suffice.

5.3 Solving matroid center

Consider a matroid $M = (S, I)$ defined on a set S , and suppose that an augmented cover tree T for S w.r.t. M is available. We can compute an $O(3 + O(\epsilon))$ -approximate solution C to the matroid center problem on M as follows. First we determine a coreset Q as the union of the independent sets associated with the nodes of level T_{ℓ^*} where $\ell^* = \ell^*(\epsilon, \text{rank}(M))$ is the index defined in Equation 3. Namely, $Q = \bigcup_{u \in T_{\ell^*}} u.\text{mis}$. (Note that $\text{rank}(M)$ is easily obtained as the size of

$r.\text{mis}$, where r is the root of T .) Then, solution C is computed by running the 3-approximation algorithm by [16] on Q . The resulting performance is summarized by following theorem, whose proof can be found in [31].

Theorem 5. *Given an augmented cover tree T for S w.r.t. $M = (S, I)$, the above procedure returns a $(3 + O(\epsilon))$ -approximation C to the matroid center problem on M , and can be implemented in time $O(\text{poly}(\text{rank}(M), (64/\epsilon)^D) + \text{rank}(M) \log \Delta)$.*

5.4 Solving diversity maximization

In [10], the authors present a coreset-based approach to yield approximate solutions to all variants of the diversity maximization problem listed in Table 1. Specifically, for a pointset S and a diversity function $\text{div}(\cdot)$, the polynomial-time sequential algorithm A_{div} yielding the α_{div} approximation mentioned in the table, is run on a coreset $Q' \subseteq S$ which is derived from an (ϵ, k) -coreset Q for k -center on S , as follows. For the remote edge and the remote cycle variants, simply setting $Q' = Q$ suffices, while for all the other variants, Q' is obtained by selecting, for each $p \in Q$, $\min\{k, |S_p|\}$ points from the subset S_p of a partition $\{S_p : p \in Q\}$ of S into disjoint subsets, where each S_p contains points $q \in S$ with $\text{dist}(p, q) \leq \epsilon r_k^*$. It is shown in [10] that running A_{div} on Q' yields an $(\alpha_{\text{div}} + O(\epsilon))$ -approximate solution for S . Now, given an (augmented) cover tree T for S , Q' can be easily constructed as follows. For the remote edge and remote clique variants, Q' is set to $\text{pts}(T_{\ell^*})$, where $\ell^* = \ell^*(\epsilon, k)$ is the index defined in Equation 3. The other variants require the use of an augmented cover tree w.r.t. to the simple k -bounded cardinality matroid $M_{k,S}$, whose independent sets are all subsets of S of at most k points, so that each node of T will store a set of (at most) k points from its subtree. Then, $Q' = \cup_{u \in T_{\ell^*}} u.\text{mis}$. The resulting performance is summarized by following theorem, where $t_{A_{\text{div}}}(\cdot)$ denotes the running time of A_{div} . The proof of the theorem can be found in [31].

Theorem 6. *Consider a cover tree T for S (augmented w.r.t. the k -bounded cardinality matroid $M_{k,S}$, when necessary). For each diversity variant in Table 1, running A_{div} on the coreset Q' extracted from T returns an $(\alpha_{\text{div}} + O(\epsilon))$ -approximate solution to the diversity maximization problem in time $O(t_{A_{\text{div}}}(k(64/\epsilon)^D) + k \log \Delta)$ for the remote edge and cycle variants, and time $O(t_{A_{\text{div}}}(k^2(64/\epsilon)^D) + k \log \Delta)$ for the other variants.*

6 Conclusions

In this paper, we provided novel coreset-based fully dynamic algorithms for (robust) k -center, matroid center and diversity maximization. All algorithms are deterministic and run on the same augmented cover tree data structure, which can thus serve queries for different problems at the same time. It is important to remark that for all problems, when the input S is large and both the spread Δ and the doubling dimension D of the metric are small, the dynamic maintenance

of the data structure and the extraction of solutions can be accomplished in time dramatically smaller than the time that would be required to compute solutions on the entire pointset from scratch.

We remark that the coresets-based approaches developed in [8] for robust matroid center, and in [7] for diversity maximization under matroid constraints, can be integrated with the approach presented in this paper, to yield fully dynamic approximations for these more general versions of the problems, with similar accuracy-performance tradeoffs.

Acknowledgments. This work was supported, in part, by MUR of Italy, under Projects PRIN 20174LF3T8 (AHeAD: Efficient Algorithms for HARnessing Networked Data), and PNRR CN00000013 (National Centre for HPC, Big Data and Quantum Computing), and by the University of Padova under Project SID 2020 (RATED-X: Resource-Allocation Tradeoffs for Dynamic and eXtreme data).

References

1. Abbassi, Z., Mirrokni, V.S., Thakur, M.: Diversity maximization under matroid constraints. In: Proc. ACM KDD. pp. 32–40 (2013)
2. Bateni, M., Esfandiari, H., Fichtenberger, H., Henzinger, M., Jayaram, R., Mirrokni, V., Wiese, A.: Optimal fully dynamic k-center clustering for adaptive and oblivious adversaries. In: Proc. ACM-SIAM SODA. pp. 2677–2727 (2023)
3. Beygelzimer, A., Kakade, S., Langford, J.: Cover trees for nearest neighbor. In: Proc. ICML. pp. 97–104 (2006)
4. Borassi, M., Epasto, A., Lattanzi, A., Vassilvitskii, S., Zadimoghaddam, M.: Better sliding window algorithms to maximize subadditive and diversity objectives. In: Proc. ACM PODS. pp. 254–268 (2019)
5. Ceccarello, M., Pietracaprina, A., Pucci, G.: Fast coresets-based diversity maximization under matroid constraints. In: Proc. ACM WSDM. pp. 81–89 (2018)
6. Ceccarello, M., Pietracaprina, A., Pucci, G.: Solving k-center clustering (with outliers) in MapReduce and streaming, almost as accurately as sequentially. PVLDB **12**(7), 766–778 (2019)
7. Ceccarello, M., Pietracaprina, A., Pucci, G.: A general coresets-based approach to diversity maximization under matroid constraints. ACM Trans. Knowl. Discov. Data **14**(5), 60:1–60:27 (2020)
8. Ceccarello, M., Pietracaprina, A., Pucci, G., Soldà, F.: Scalable and space-efficient robust matroid center algorithms. Journal of Big Data **10**(49) (2023)
9. Ceccarello, M., Pietracaprina, A., Pucci, G., Upfal, E.: A practical parallel algorithm for diameter approximation of massive weighted graphs. In: Proc. IEEE IPDPS. pp. 12–21 (2016)
10. Ceccarello, M., Pietracaprina, A., Pucci, G., Upfal, E.: MapReduce and streaming algorithms for diversity maximization in metric spaces of bounded doubling dimension. Proc. VLDB Endow. **10**(5), 469–480 (2017)
11. Chakrabarty, D., Goyal, P., Krishnaswamy, R.: The non-uniform k-center problem. ACM Trans. on Algorithms **16**(4), 46:1–46:19 (2020)
12. Chan, T.H., Guerin, A., Sozio, M.: Fully dynamic k-center clustering. In: Proc. WWW. pp. 579–587 (2018)

13. Chan, T.H., Lattanzi, S., Sozio, M., Wang, B.: Fully dynamic k-center clustering with outliers. In: Proc. COCOON. pp. 150–161 (2023)
14. Charikar, M., Chekuri, C., Feder, T., Motwani, R.: Incremental clustering and dynamic information retrieval. In: Proc. ACM STOC. pp. 626–635 (1997)
15. Charikar, M., Khuller, S., Mount, D., Narasimhan, G.: Algorithms for facility location problems with outliers. In: Proc. ACM-SIAM SODA. pp. 642–651 (2001)
16. Chen, D., J.Li, H.L., Wang, H.: Matroid and knapsack center problems. *Algorithmica* **75**(1), 27–52 (2016)
17. Chiplunkar, A., Kale, S., Ramamoorthy, S.: How to solve fair k-center in massive data models. In: Proc. ICML. pp. 1877–1886 (2020)
18. Cohen-Addad, V., Schwiegelshohn, C., Sohler, C.: Diameter and k-center in sliding windows. In: Proc. ICALP (2016)
19. Elkin, Y., Kurlin, V.: Counterexamples expose gaps in the proof of time complexity for cover trees introduced in 2006. In: Proc. IEEE TopoInVis. pp. 9–17 (2022)
20. Gonzalez, T.: Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science* **38**, 293–306 (1985)
21. Goranci, G., Henzinger, M., Leniowski, D., Schulz, C., Svozil, A.: Fully dynamic k-center clustering in low dimensional metrics. In: Proc. ALENEX 2021. pp. 143–153 (2021)
22. Gottlieb, L.A., Kontorovich, A., Krauthgamer, R.: Efficient classification for metric data. *IEEE Trans. Information Theory* **60**(9), 5750–5759 (2014)
23. Harris, D., Pensyl, T., Srinivasan, A., Trinh, K.: A lottery model for center-type problems with outliers. *ACM Trans. on Algorithms* **15**(3), 36:1–36:25 (2019)
24. Kale, S.: Small space stream summary for matroid center. In: Proc. AP-PROX/RANDOM. pp. 20:1–20:22 (2019)
25. Kleindessner, M., Awasthi, P., Morgenstern, J.: Fair k-center clustering for data summarization. In: Proc. ICML. pp. 3448–3457 (2019)
26. Krauthgamer, R., Lee, J.: Navigating nets: simple algorithms for proximity search. In: Proc. ACM-SIAM SODA. pp. 798–807 (2004)
27. Leskovec, J., Rajaraman, A., Ullman, J.: *Mining of Massive Data Sets*. Cambridge University Press (2014)
28. Oxley, J.: *Matroid Theory*. Oxford University Press (2006)
29. Pellizzoni, P., Pietracaprina, A., Pucci, G.: Dimensionality-adaptive k-center in sliding windows. In: Proc. DSAA. pp. 197–206 (2020)
30. Pellizzoni, P., Pietracaprina, A., Pucci, G.: k-center clustering with outliers in sliding windows. *Algorithms* **15**(2), 52 (2022)
31. Pellizzoni, P., Pietracaprina, A., Pucci, G.: Fully dynamic clustering and diversity maximization in doubling metrics. arXiv preprint arXiv:2302.07771 (2023)