# A Composable Coreset for $k$-Center in Doubling Metrics

Sepideh Aghamolaei* Mohammad Ghodsi†


## Abstract

A set of points $P$ in a metric space and a constant integer $k$ are given. The $k$-center problem finds $k$ points as *centers* among $P$, such that the maximum distance of any point of $P$ to their closest centers ($r$) is minimized.

Doubling metrics are metric spaces in which for any $r$, a ball of radius $r$ can be covered using a constant number of balls of radius $r/2$. Fixed dimensional Euclidean spaces are doubling metrics. The lower bound on the approximation factor of $k$-center is 1.822 in Euclidean spaces, however, $(1+\epsilon)$-approximation algorithms with exponential dependency on $\frac{1}{\epsilon}$ and $k$ exist.

For a given set of sets $P_1, \ldots, P_L$, a *composable coreset* independently computes subsets $C_1 \subset P_1, \ldots, C_L \subset P_L$, such that $\cup_{i=1}^{L} C_i$ contains an approximation of a measure of the set $\cup_{i=1}^{L} P_i$.

We introduce a $(1+\epsilon)$-approximation composable coreset for $k$-center, which in doubling metrics has size sublinear in $P$. This results in a $(2+\epsilon)$-approximation algorithm for $k$-center in MapReduce with a constant number of rounds and sublinear communications, which improves upon the previous 4-approximation algorithm. We also prove a trade-off between the size and the approximation factor of our coreset, and give a composable coreset for a related problem called dual clustering.


## 1 Introduction

Coresets are subsets of points that approximate a measure of the point set. A method of computing coresets on big data sets is composable coresets. Composable coresets [20] provide a framework for adapting constant factor approximation algorithms to streaming and MapReduce models. Composable coresets summarize distributed data so that the scalability is increased, while keeping the desirable approximation factor and time complexity.

There is a general algorithm for solving problems using coresets which known by different names in different settings: mergeable summaries [1] and merging in a tree-like structure [2] for streaming $(1+\epsilon)$-approximation algorithms, small space (divide and conquer) for constant factor approximations in streaming [15], and composable coresets in MapReduce [20]. A consequence of using constant factor approximations instead of $(1+\epsilon)$-approximations with the same merging method is that it can add a $O(\log n)$ factor to the approximation factor of the algorithm on an input of size $n$.

Composable coresets [20] require only a single round and sublinear communications in the MapReduce model, and the partitioning is done arbitrarily.

**Definition 1 (Composable Coreset)** *A composable coreset on a set of sets $\{S_i\}_{i=1}^{L}$ is a set of subsets $C(S_i) \subset S_i$ whose union gives an approximation solution for an objective function $f : (\cup_{i=1}^{L} S_i) \to \mathbf{R}$. Formally, a composable coreset of a minimization problem is an $\alpha$-approximation if*

$$f(\cup_i S_i) \leq f(\cup_i C(S_i)) \leq \alpha . f(\cup_i S_i),$$

*for a minimization problem. The maximization version is similarly defined.*

A *partitioned composable coreset* is a composable coreset in which the initial sets are a partitioning, i.e. sets $\{S_i\}_{i=1}^{L}$ are disjoint. Using Gonzalez's algorithm for $k$-center [14], Indyk, et al. designed a composable coreset for a similar problem known as the diversity maximization problem [20]. Other variations of composable coresets are randomized composable coresets and mapping coresets. Randomized composable coresets [26] share the same divide and conquer approach as other composable coresets and differ from composable coresets only in the way they partition the data. More specifically, randomized composable coresets, randomly partitioning the input, as opposed to other composable coresets which make use of arbitrary partitioning. Mapping coresets [5] extend composable coresets by adding a mapping between coreset points and other points to their coresets and keep almost the same amount of data in all machines. Algorithms for clustering in $\ell^p$ norms using mapping coresets are known [5]. Further improvements of composable coresets for diversity maximization [20] include lower bounds [3] and multi-round composable coresets in metrics with bounded doubling dimension [6].

*Metric $k$-center* is a NP-hard problem for which 2-approximation algorithms that match the lower bound for the approximation factor of this problem are

*Department of Computer Engineering, Sharif University of Technology, aghamolaei@ce.sharif.edu

†Department of Computer Engineering, Sharif University of Technology, School of Computer Science, Institute for Research in Fundamental Sciences (IPM), ghodsi@sharif.edu

known [28, 14]. Among approximation algorithms for $k$-center is a parametric pruning algorithm, based on the minimum dominating set [28]. In this algorithm, an approximate dominating set is computed on the disk graph of the input points. The running time of the algorithm is $O(n^3)$. The greedy algorithm for $k$-center requires only $O(nk)$ time [14] and unlike the algorithm based on the minimum dominating set[28], uses $r$-nets [17]. A $(1+\epsilon)$-approximation coreset exists for $k$-center [4] with size exponentially dependent on $\frac{1}{\epsilon}$.

Let the optimal radius of $k$-center for a point set $P$ be $r$. The problem of finding the smallest set of points that cover $P$ using radius $r$ is known as the *dual clustering problem* [7].

Metric dual clustering (of $k$-center) has an unbounded approximation factor [7]. In Euclidean metric, there exists a streaming $O(2^d d \log d)$-approximation algorithm for this problem [7]. Also, any $\alpha$-approximation algorithm for the minimum disk/ball cover problem gives a 2-approximation coreset of size $\alpha k$ for $k$-center, so 2-approximation coresets of size $(1+\epsilon)k$ exist for this problem [23]. A greedy algorithm for dual clustering of $k$-center has also been used as a preprocessing step of density-based clustering (DBSCAN) [11]. Implementing DBSCAN efficiently in MapReduce is an important problem [18, 9, 13, 27, 21].

Randomized algorithms for metric $k$-center and $k$-median in MapReduce [10] exist. These algorithms take $\alpha$-approximation offline algorithms and return $(4\alpha+2)$-approximation and $(10\alpha+3)$-approximation algorithms for $k$-center and $k$-median in MapReduce, respectively. The round complexity of these algorithms depends on the probability of the algorithm for finding a good approximation.

Current best results on metric $k$-center in MapReduce have 2 rounds and give the approximation factor 4 [24]. However, a 2-approximation algorithm exists if the cost of the optimal solution is known [19]. Experiments in [25] suggest that running Gonzalez's algorithm on a random partitioning and an arbitrary partitioning results in the same approximation factor.

## Warm-Up

Increasing the size of coresets in the first step of computing composable coresets can improve the approximation factor of some problems. The approximation factor of $k$-median algorithm of [15] is $2c(1+2b)+2b$, where $b$ and $c$ are the approximation factors of $k$-median and weighted $k$-median, respectively. This algorithm computes a composable coreset, where a coreset for $k$-median is the set of $k$ medians weighted by the number of points assigned to each median.

A pseudo-approximation for $k$-median finds $k+O(1)$ median and has approximation factor $1+\sqrt{3}+\epsilon$ [22]. Using a pseudo-approximation algorithm in place of $k$-median algorithms in the first step of [15], it is possible to achieve a better approximation factor for $k$-median using the same proof as [15]. Since any pseudo-approximation has a cost less than or equal to the optimal solution; replacing them will not increase the cost of clustering.

The approximation factor using [8] as weighted $k$-median coresets is 91.66, while the best $k$-median algorithm would give a 99.33 factor using the same algorithm ($b = 1+\sqrt{3}$). The lower bound on the approximation factor of this algorithm using the same weighted $k$-median algorithm but without pseudo-approximation is 63.09 ($b = 1+\frac{2}{e}$).

## Contributions

We give a $(1+\epsilon)$-approximation coreset of size $(\frac{4}{\epsilon})^{1+2b}k$ for $k$-center in metric spaces with doubling dimension $b$. Using composable coresets, our algorithm generalizes to MapReduce setting, where it becomes a $(1+\epsilon)$-approximation coreset of size $(\frac{4}{\epsilon})^{1+2b}\frac{n}{m}k$, given memory $m$, which is sublinear in the input size $n$.

| Conditions | Approx. | Reference |
|---|---|---|
| Metric $k$-center: | | |
| $O(1)$-rounds | 4 | [24] |
| $O(\log^{\Delta}_{1+\epsilon})$ rounds | $2+\epsilon$ | [19] |
| Lower bound | 2 | offline [28] |
| Doubling metrics: | | |
| $O(1)$-rounds | $2+\epsilon$ | Theorem 7 |
| Lower bound | 1.822 | [12] |
| Dual clustering: | | |
| General metrics | $O(\log n)$ | min dominating set [28], composable coreset [20] |
| Doubling metrics | $O(1)$ | Theorem 3 |

Table 1: Summary of results for $k$-center and dual clustering in MapReduce. $\Delta$ is the diameter of the point-set.

Using the composable coreset for dual clustering, we find a $(2+\epsilon)$-approximation composable coreset for $k$-center, which has a sublinear size in metric spaces with constant doubling dimension. More specifically, if an $\alpha$-approximation exists for doubling metrics, our algorithm provides $(\alpha+\epsilon)$-approximation factor. It improves the previous 4-approximation algorithm [24, 25] in MapReduce. A summary of results on $k$-center is shown in Table 1. Note that for MapReduce model, each round can take a polynomial amount of time, however, the space available to each machine is sublinear.

Our algorithm achieves a trade-off between the approximation factor and the size of coreset (see fig. 1). The approximation factor of our algorithm and the size of the resulting composable coreset for $L$ input sets are

$\alpha = 2 + \epsilon$ and $kL\beta$, respectively. This trade-off is the main idea of our algorithm.

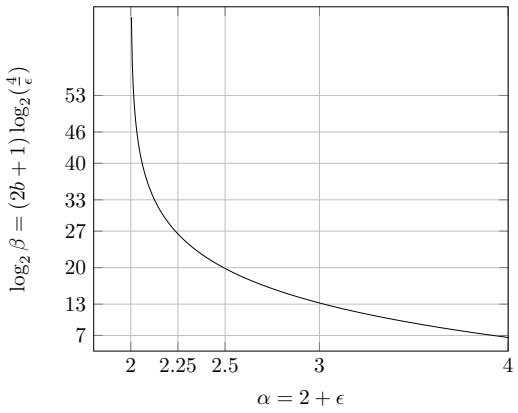


Figure 1: Space-approximation factor trade-off of our $\alpha$-approx. coreset of size $\beta kL$ for $k$-center in Euclidean plane.

Our composable coresets give single pass streaming algorithms and 1-round approximation algorithms in MapReduce with sublinear communication, since each coreset is communicated once, and the size of the coreset is constant.

## 2 Preliminaries

First we review some basic definitions, models and algorithms in computational geometry and MapReduce.

### 2.1 Definitions

Some geometric definitions and notations are reviewed here, which have been used in the rest of the paper.

**Definition 2 (Metric Space)** *A (possibly infinite) set of points $P$ and a distance function $d(.,.)$ create a metric space if the following three conditions hold:*

- $\forall p, q \in P \quad d(p,q) = 0 \Leftrightarrow p = q$
- $\forall p, q \in P \quad d(p,q) = d(q,p)$
- $\forall p, q, t \in P \quad d(p,q) + d(q,t) \geq d(p,t)$, *known as triangle inequality*

Metrics with bounded doubling dimension are called *doubling metrics*. Constant dimension Euclidean spaces under $\ell^p$ norms and Manhattan distance are examples of doubling metrics.

Doubling constant [16] of a metric space is the number of balls of radius $r$ that lie inside a ball of radius $2r$. The logarithm of doubling constant in base 2 is called *doubling dimension*. Many algorithms have better approximation factors in doubling metrics compared to general metric spaces. The doubling dimension of Euclidean plane is $\log_2 7$.

**Definition 3 (Doubling Dimension [16])** *For any point $x$ in a metric space and any $r \geq 0$, if the ball of radius $2r$ centered at $x$ can be covered with at most $2^b$ balls of radius $r$, we say the doubling dimension of the metric space is $b$.*

$k$-Center is a NP-hard clustering problem with clusters in shapes of $d$-dimensional balls.

**Definition 4 (Metric $k$-Center [28])** *Given a set $P$ of points in a metric space, find a subset of $k$ points as cluster centers $C$ such that*

$$\forall p \in P, \min_{c \in C} d(p,c) \leq r$$

*and $r$ is minimized.*

The best possible approximation factor of metric $k$-center is 2 [28].

Geometric intersection graphs represent intersections between a set of shapes. For a set of disks, their intersection graph is called a disk graph.

**Definition 5 (Disk Graph)** *For a set of points $P$ in a metric space with distance function $d(.,.)$ and a radius $r$, the disk graph of $P$ is a graph whose vertices are $P$, and whose edges connect points with distance at most $2r$.*

**Definition 6 (Dominating Set)** *Given a graph $G = (V, E)$, the smallest subset $Q \subset V$ is a minimum dominating set, if $\forall v \in V, v \in Q \vee \exists u \in Q : (v,u) \in E$.*

We define the following problem as a generalization of the dual clustering of [7] by removing the following two conditions: the radius of balls is 1, and the set of points are in $\mathbf{R}^d$.

**Definition 7 (Dual Clustering)** *Given a set of points $P$ and a radius $r$, the dual clustering problem finds the smallest subset of points as centers $(C), C \subset P$ such that the distance from each point to its closest center is at most $r$.*

### 2.2 An Approximation Algorithm for Metric $k$-Center

Here, we review the parametric pruning algorithm of [28] for metric $k$-center.

**Algorithm 1** Parametric Pruning for $k$-Center [28]

**Input:** A metric graph $G = (V, E)$, an integer $k$
**Output:** A subset $C \subset V, |C| \leq k$
  Sort $E$ such that $e_1 \leq e_2 \leq \cdots \leq e_{|E|}$.
  $G' = (V, E') \leftarrow (V, \emptyset)$
  **for** $i = 1, \ldots, |E|$ **do**
    $E' \leftarrow E' \cup \{e_i\}$
    Run algorithm 2 on $G'$.
    **if** $|IS| \leq k$ **then return** $IS$

Using this algorithm on a metric graph $G$, a 2-approximation for the optimal radius $r$ can be determined. In algorithm 1, edges are added by increasing order of their length until reaching $r$. Given this radius, another graph ($G'$) is built, where edges exist between points within distance at most $r$ of each other.

**Algorithm 2** Approximate dominating set of $G$ [28]

**Input:** A metric graph $G' = (V, E)$
**Output:** A subset $C \subset V$
  $G'^2 \leftarrow G'$
  **for** $\forall (u,t), (t,v) \in E$ **do**
    Add $(u, v)$ to $G'^2$.
  Find a maximal independent set $IS$ of $G'^2$
  **return** $IS$

Hence, by definition, a minimum dominating set of $G'$ is an optimal $k$-center of $G$. Every cluster is a star in $G'$ which turns into a clique in $G'^2$. Therefore, a maximal independent set of $G'^2$ chooses at most one point from each cluster. Algorithm 2 computes $G'^2$ and returns a maximal independent set of $G'^2$.

Computing a maximal independent set takes $O(|E|)$ time. The graph $G'^2$ in Algorithm 2 only changes in each iteration of Algorithm 1 around the newly added edge, so, updating the previous graph and $IS$ takes $O(n)$ time. Therefore, the time complexity of Algorithm 1 is $O(|E| \cdot n) = O(n^3)$.

## 3 A Coreset for Dual Clustering in Doubling Metrics

In this section, we prove a better approximation offline coreset for the dual clustering problem. Our method is based on Algorithm 1 which first builds the disk graph with radius $r$, then covers this graph using a set of stars. We prove the maximum degree of those stars is $D^2$, where $D$ is the doubling constant. The result is an approximation algorithm for dual clustering in doubling metrics.

### 3.1 Algorithm

We add a preprocessing step to Algorithm 1 to find a better approximation factor for $k$-center and dual clustering problems.

**Algorithm 3** A Coreset for $k$-Center

**Input:** A set of points $P$, an integer $k$ or a radius $r$
**Output:** A subset $C \subset P, |C| \leq k$
  **if** $k$ is given in the input **then**
    Compute a 2-approximation solution for $k$-center (radius $r$).
  $E \leftarrow$ all pairs of points with distance at most $r/2$.
  Run algorithm 2 on $G = (P, E)$ to compute $IS$.
  **return** $IS$

### 3.2 Analysis

Unlike in general metric spaces, $k$-center in doubling metrics admits a space-approximation factor trade-off. More specifically, doubling or halving the radius of $k$-center changes the number of points in the coreset by a constant factor, since the degrees of vertices in the minimum dominating set are bounded in those metric spaces.

**Lemma 1** *For each cluster $C_i$ of Algorithm 3 with radius $r'$, the maximum number of points $(\Delta + 1)$ from $C_i$ that are required to cover all points inside $C_i$ with radius $r'/2$ is at most $D^2$, i.e.*

$$(\Delta + 1) \leq D^2,$$

*where $D$ is the doubling constant of the metric space.*

**Proof.** Assume a point $p \in IS$ returned by Algorithm 3. By the definition of doubling metrics, there are $D$ balls of radius $r'/2$ centered at $b_1, \ldots, b_D$ called $B_1, \ldots, B_D$ that cover the ball of radius $r'$ centered at $p$, called $B$.

$$\forall q \in B, \exists B_i, i = 1, \ldots, D : d(p, b_i) \leq r'/2$$

Repeating this process for each ball $B_i$ results in a set of at most $D$ balls $(B'_{i,1}, \ldots, B'_{i,D})$ of radius $r'/4$ centered at $b'_{i,1}, \ldots, b'_{i,D}$.

$$\forall q \in B'_{i,j}, d(b'_{i,j}, q) \leq r'/4$$

Choose a point $p_{i,j} \in P \cap B'_{i,j}$. Using triangle inequality,

$$\begin{aligned} \forall q \in B'_{i,j}, d(p_{i,j}, q) &\leq d(p_{i,j}, b'_{i,j}) + d(b'_{i,j}, q) \\ &\leq r'/4 + r'/4 = r'/2. \end{aligned}$$

We claim any minimal solution needs at most one point from each ball $B'_{i,j}$. By contradiction, assume there are two point $p_{i,j}, q'$ in the minimal solution that lie inside a ball $B'_{i,j}$. After removing $q'$, the ball with radius $r'/2$ centered at $p_{i,j}$ still covers $B'_{i,j}$, since:

$$\begin{aligned} \forall q \in P, \exists B_i, B'_{i,j} \ni q, p_{i,j} \\ d(q, p_{i,j}) &\leq d(q, b'_{i,j}) + d(b'_{i,j}, p_{i,j}) \\ &\leq r'/4 + r'/4 = r/2'. \end{aligned}$$

Then we have found a point $(q')$ whose removal decreases the size of the solution, which means the solution was not minimal. So the size of any minimal set of points covering $B$ is at most $D^2$. □

**Lemma 2** *In a metric space with doubling constant $D$, if a dual clustering with radius $r$ has $k$ points, then a dual clustering with radius $r/2$ exists which has $D^2k$ points.*

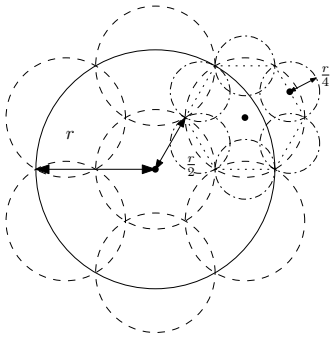


Figure 2: Applying the doubling dimension bound twice (Lemma 1).

**Proof.** Let $p$ be a center in the $k$-center problem. Based on the proof of Lemma 1, there are $\Delta$ vertices adjacent to $p$ that cover the points inside the ball of radius $r$ centered at $p$, using balls of radius $r/2$ and a ball of radius $r/2$ centered at $p$. By choosing all these vertices as centers, it is possible to cover all input points $P$ with radius $r/2$. Using the same reasoning for all clusters, it is possible to cover all points using $(\Delta+1)k$ centers. Using the bound in Lemma 1, these are $D^2k$ centers. □

**Theorem 3** *The approximation factor of Algorithm 3 is $D^2$ for the dual clustering.*

**Proof.** Since the radius of balls in Lemma 2 is at most the optimal radius for $k$-center, the approximation factor of dual clustering is the number of points chosen as centers divided by $k$, which is $D^2$. □

**Theorem 4** *The approximation factor of the coreset for $k$-center in Algorithm 3 is $2^{-R}$ and its size is $D^{2(R+1)}k$.*

**Proof.** Applying Lemma 2 halves the radius and multiplies the number of points by $D^2$. So, applying this lemma $R$ times gives $(D^2)^{R+1}k$ points, since it might be the case that in the first step of the algorithm the optimal radius was found, and we divided it by 2. The radius remains $\frac{r}{2^R}$ because of the case where we had found a 2-approximation. □

**Theorem 5** *Algorithm 3 given $(\frac{4}{\epsilon})^{2\log_2 D}k$ as input, is a $(1+\epsilon)$-approximation coreset of size $(\frac{4}{\epsilon})^{2\log_2 D}k$ for the $k$-center problem.*

**Proof.** For $R = \lceil \log_2 \frac{2}{\epsilon} \rceil$, the proof of Theorem 4 gives $(\frac{4}{\epsilon})^{2\log_2 D}$ points and radius $r\epsilon$. Assume $O$ is the set of $k$ centers returned by the optimal algorithm for point-set $P$, and $C$ is the set of centers returned by running the optimal algorithm on the coreset of $P$. For any point $p \in P$, let $o$ be the center that covers $p$ and $c$ be the point that represents $o$ in the coreset. Using triangle inequality:

$$d(p,c) \leq d(p,o) + d(o,c) \leq r + r\epsilon = (1+\epsilon)r$$

So, computing a $k$-center on this coreset gives a $(1+\epsilon)$-approximation. □

## 4 A Composable Core-Set for $k$-Center in Doubling Metrics

Our general algorithm for constructing coresets based on dual clustering has the following steps:

- Compute the cost of an approximate solution ($X$).
- Find a composable coreset for dual clustering with cost $X$.
- Compute a clustering on the coreset.

In this section, we use this general algorithm for solving $k$-center.

### 4.1 Algorithm

Knowing the exact or approximate value of $r$, we can find a single-round $(2+\epsilon)$-approximation for metric $k$-center in MapReduce. Although the algorithm achieves the aforementioned approximation factor, the size of the coreset and the communication complexity of the algorithm depend highly on the doubling dimension.

**Algorithm 4** $k$-Center

**Input:** A set of sets of points $\cup_{i=1}^{L} S_i$, a $k$-center algorithm

**Output:** A set of $k$ centers

1: Run a $k$-center algorithm on each $S_i$ to find the radius $r_i$.
2: Run Algorithm 2 on the disk graph of each set $S_i$ with radius $\frac{\epsilon r_i}{2}$ locally to find $C(S_i)$.
3: Send $C(S_i)$ to set 1 to find the union $\cup_i C(S_i)$.
4: Run a 2-approximation $k$-center algorithm on $\cup_{i=1}^{L} C(S_i)$ to find the set of centers $C$.
5: **return** $C$.

Based on the running time of Algorithm 2 and Gonzalez's algorithm, the running time of Algorithm 4 is $\sum_i [O(k \cdot |S_i|) + O(|S_i|^2)] + O(k \sum_i |C(S_i)|) = O(kn)$. Since the sum of running times of machines is of the same order as the best sequential algorithm, Algorithm 4 is a work-efficient parallel algorithm.

We review the following well-known lemma:

**Lemma 6** *For a subset $S \subset P$, the optimal radius of the $k$-center of $S$ is at most twice the radius of the $k$-center of $P$.*

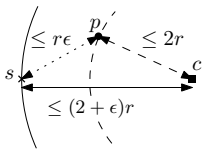


Figure 3: The dominating set on $\cup_i C(S_i)$ covers $\cup_i S_i$ with radius $(2+\epsilon)$(Theorem 7).

**Proof.** Consider the set of clusters $O_i$ in the optimal $k$-center of $P$ centered at $c_i, i = 1, \ldots, k$ with radius $r$. If $c_i \in S$, then the points of $O_i \cap S$ are covered by $c_i$ with radius $r$, as before. Otherwise, select an arbitrary point in $O_i \cap S$ as the new center $c_i'$. Using the triangle inequality on $c_i, c_i'$ and any point $p \in O_i \cap S$:

$$d(p, c_i') \leq d(p, c_i) + d(c_i, c_i') \leq r + r = 2r$$

Since $c_i'$ was covered using $c_i$ with radius $r$. So the set $S \cap O_i$ can be covered with radius $2r$. Note that since we choose at most one point from each set, the number of new centers is at most $k$. □

**Theorem 7** *The approximation factor of Algorithm 4 is $2+\epsilon$ for metric $k$-center.*

**Proof.** Let $r$ be the optimal radius of $k$-center for $\cup_i S_i$. Since $\cup_i C(S_i) \subset \cup_i S_i$, using Lemma 6, the radius of $k$-center for $\cup_i C(S_i)$ is at most $2r$. The radius of $k$-center inside each set $S_i$ is at most $2r$ for the same reason. The algorithm computes a covering $S_i$ with balls of radius $r_i\epsilon/2$. Based on the fact that offline $k$-center has 2-approximation algorithms and the triangle inequality, the approximation factor of the algorithm proves to be $(2+\epsilon)$-approximation (Figure 3). Let $p = \arg\min_{p \in \cup_i C(S_i)} dist(s, p)$, then

$$\begin{aligned} \forall s \in S_i \exists c \in C, d(s, c) &\leq d(s, p) + d(p, c) \leq r' + r_i\epsilon/2 \\ &\leq 2r + 2r\epsilon/2 = (2+\epsilon)r \end{aligned}$$

where $r'$ is the radius of the offline $k$-center algorithm on $C$. □

### 4.2 Analysis

**Lemma 8** *In a metric space with doubling constant $D$, the union of dual clusterings of radius $r$ computed on sets $S_1, \ldots, S_L$ is a $(L \times D^{2\log_2 \frac{8}{\epsilon}})$-approximation for the dual clustering of radius $r(1+\epsilon)$ of their union $(\cup_{i=1}^{L} S_i)$.*

**Proof.** Each center in the dual clustering with radius $r$ of $P = (\cup_{i=1}^{L} S_i)$ has at most $\Delta$ adjacent vertices covered by this center. Consider a point $p \in P$ covered by center $c$ in a solution for $P$. If $p$ and $c$ belong to the same set $S_i$, assign $p$ to $c$. Otherwise, pick any point that was previously covered by $c$ as the center that covers $p$.

While this might increase the radius by a factor 2, it does not increase the number of centers in each set. Since the algorithm uses radius $\epsilon.r/2$, it increases the number of centers to $D^{2\log_2 \frac{8}{\epsilon}} k$ (based on Theorem 4 for $R = \frac{4r}{\epsilon r/2}$) but keeps the approximation factor of the radius to $1+\epsilon$. There are $L$ such sets, so the size of the coreset is $L \times D^{2\log_2 \frac{8}{\epsilon}} k$. □

**Theorem 9** *Algorithm 4 returns a coreset of size $O(kL)$ for $k$-center in metric spaces with fixed doubling dimension.*

**Proof.** The coreset of each set $S_i$ has a radius $r_i$ varying from the optimal radius ($r = r_i$) to $2\beta.r$, where $\beta$ is the approximation factor of the offline algorithm for $k$-center. Clearly, the lower bound holds because any radius is at least as much as the optimal (minimum) radius, which means $r \leq r_i$; and Lemma 6 when applied to $S_i \subset \cup_i S_i$, yields the upper bound.

$$r \leq r_i \leq 2\beta.r \Rightarrow \frac{r\epsilon}{4\beta} \leq \frac{r_i\epsilon}{4\beta} \leq \frac{\epsilon r}{2}$$

Reaching value $r\epsilon$ requires applying Theorem 7 at most $\log_2 \frac{4\beta}{\epsilon}$ times.

The size of the resulting coreset is therefore at most

$$(4^{\log_2 D})^{\log_2 \frac{4\beta}{\epsilon}} kL = (\frac{4\beta}{\epsilon})^{2(\log_2 D)} kL.$$

Here, we use the best approximation factor for metric $k$-center ($\beta = 2$), which gives a coreset of size $(\frac{8}{\epsilon})^{2(\log_2 D)} kL = O(kL)$ for fixed $\epsilon$. □

## 5 Conclusions

We proved a trade-off between the approximation factor and the number of centers for the $k$-center problem in doubling metrics. To improve the trade-off in MapReduce, local partitioning methods such as grid-based or locality sensitive hashing, or degree based partitioning of disk graph with lower radius might be effective.

Gonzalez's algorithm [14] is a version of parametric pruning algorithm [28] in which the greedy maximal independent set computation prioritizes the points with maximum distance from the currently chosen points. Our algorithm and trade-off partially answers the open question of [25] about comparing and improving these two algorithms in MapReduce.

Our composable coreset for dual clustering gives constant factor approximation for minimizing the size of DBSCAN cluster representatives if half the input radius is used, and the dominating set subroutine is replaced with the connected dominating set.

## References

[1] P. K. Agarwal, G. Cormode, Z. Huang, J. M. Phillips, Z. Wei, and K. Yi. Mergeable summaries. *ACM Transactions on Database Systems (TODS)*, 38(4):26, 2013.

[2] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Approximating extent measures of points. *Journal of the ACM (JACM)*, 51(4):606–635, 2004.

[3] S. Aghamolaei, M. Farhadi, and H. Zarrabi-Zadeh. Diversity maximization via composable coresets. In *Canadian Conference on Computational Geometry (CCCG)*, 2015.

[4] M. Bādoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 250–257. ACM, 2002.

[5] M. Bateni, A. Bhaskara, S. Lattanzi, and V. Mirrokni. Distributed balanced clustering via mapping coresets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2591–2599, 2014.

[6] M. Ceccarello, A. Pietracaprina, G. Pucci, and E. Upfal. Mapreduce and streaming algorithms for diversity maximization in metric spaces of bounded doubling dimension. *Proceedings of the VLDB Endowment*, 10(5):469–480, 2017.

[7] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. *SIAM Journal on Computing*, 33(6):1417–1440, 2004.

[8] M. Charikar, S. Guha, E. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem (extended abstract). In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC '99, pages 1–10, New York, NY, USA, 1999. ACM.

[9] B.-R. Dai and I.-C. Lin. Efficient map/reduce-based dbscan algorithm with optimized data partition. In *2012 IEEE 5th International Conference on Cloud Computing (CLOUD)*, pages 59–66. IEEE, 2012.

[10] A. Ene, S. Im, and B. Moseley. Fast clustering using mapreduce. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD)*, pages 681–689. ACM, 2011.

[11] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD)*, volume 96, pages 226–231, 1996.

[12] T. Feder and D. Greene. Optimal algorithms for approximate clustering. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 434–444. ACM, 1988.

[13] Y. X. Fu, W. Z. Zhao, and H. F. Ma. Research on parallel dbscan algorithm design based on mapreduce. In *Advanced Materials Research*, volume 301, pages 1133–1138. Trans Tech Publ, 2011.

[14] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science (TCS)*, 38:293–306, 1985.

[15] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: Theory and practice. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 15(3):515–528, 2003.

[16] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 534–543. IEEE, 2003.

[17] S. Har-Peled and M. Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM Journal on Computing*, 35(5):1148–1184, 2006.

[18] Y. He, H. Tan, W. Luo, S. Feng, and J. Fan. Mr-dbscan: a scalable mapreduce-based dbscan algorithm for heavily skewed data. *Frontiers of Computer Science*, 8(1):83–99, 2014.

[19] S. Im and B. Moseley. Brief announcement: Fast and better distributed mapreduce algorithms for k-center clustering. In *Proceedings of the 27th ACM symposium on Parallelism in Algorithms and Architectures*, pages 65–67. ACM, 2015.

[20] P. Indyk, S. Mahabadi, M. Mahdian, and V. S. Mirrokni. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, pages 100–108. ACM, 2014.

[21] Y. Kim, K. Shim, M.-S. Kim, and J. S. Lee. Dbcure-mr: an efficient density-based clustering algorithm for large data using mapreduce. *Information Systems*, 42:15–35, 2014.

[22] S. Li and O. Svensson. Approximating k-median via pseudo-approximation. *SIAM Journal on Computing*, 45(2):530–547, 2016.

[23] C. Liao and S. Hu. Polynomial time approximation schemes for minimum disk cover problems. *Journal of combinatorial optimization*, 20(4):399–412, 2010.

[24] G. Malkomes, M. J. Kusner, W. Chen, K. Q. Weinberger, and B. Moseley. Fast distributed k-center clustering with outliers on massive data. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1063–1071, 2015.

[25] J. McClintock and A. Wirth. Efficient parallel algorithms for k-center clustering. In *Parallel Processing (ICPP), 2016 45th International Conference on*, pages 133–138. IEEE, 2016.

[26] V. Mirrokni and M. Zadimoghaddam. Randomized composable core-sets for distributed submodular maximization. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing (STOC)*, pages 153–162. ACM, 2015.

[27] M. Noticewala and D. Vaghela. Mr-idbscan: Efficient parallel incremental dbscan algorithm using mapreduce. *International Journal of Computer Applications (IJCA)*, 93(4), 2014.

[28] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013.