Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Master Thesis in Computer Engineering

Matroid-center clustering in sliding windows

Master Candidate

Lorenzo Cappellotto

Student ID 2044728

Supervisor

Prof. Andrea Alberto Pietracaprina

University of Padova

Supervisor

Prof. Geppino Pucci

University of Padova

Academic Year

2022/2023

To my mother and sister,
who shaped me into who I am today.

Abstract

In this thesis, we study the Matroid Center problem which, given a set W of points from a metric space and an integer $k < |W|$, requires to find a subset $S \subset W$ of k centers such that the maximum distance of a point of W from S is minimized, and S is an independent set of a specified matroid. In particular, we consider the partition matroid, which can be used to model fairness constraints, and the more general transversal matroid. For both matroids we devise the first approximation streaming algorithms under the sliding window model, which, at any time, are able to efficiently compute a solution to Matroid Center for the latest window of points of the stream. The algorithms exhibit a $(3 + \epsilon)$-approximate ratio, where 3 is the best approximation attainable in sequential polynomial time and $\epsilon \in (0, 1)$ is a user-defined accuracy parameter. The analysis of the algorithms is carried out in terms of the dimensionality of the data, and it shows that, for low dimensional data the required working memory and processing time are asymptotically and significantly smaller than the window size.

Sommario

In questa tesi, studiamo il problema del Matroid Center, il cui obiettivo è trovare, dato un insieme W di punti presi da uno spazio metrico e un intero $k < |W|$, un sottoinsieme $S \subset W$ di k centri tale per cui la distanza massima di un punto di W da S sia minimizzata e S sia un insieme indipendente di una matroide specificata. In particolare, esaminiamo i casi relativi alla matroide di partizione, che può essere utilizzata per modellare vincoli di equità, e alla più generale matroide trasversale. Per entrambe le matroidi costruiamo i primi algoritmi di approssimazione per il modello di calcolo a finestre scorrevoli, i quali, in qualsiasi momento, sono in grado di calcolare in modo efficiente una soluzione del Matroid Center per l'ultima finestra di punti dello stream. Questi algoritmi presentano un fattore di approssimazione pari a $(3 + \epsilon)$, dove 3 è la migliore approssimazione ottenibile sequenzialmente in tempo polinomiale e $\epsilon \in (0, 1)$ rappresenta un parametro di precisione definito dall'utente. L'analisi degli algoritmi viene effettuata in termini di dimensionalità dei dati e mostra che, per dati a bassa dimensionalità, la memoria di lavoro richiesta e il tempo di elaborazione sono asintoticamente e significativamente inferiori alla dimensione della finestra.

Contents

1 Introduction

In recent years, the explosive growth of data has presented both opportunities and challenges for various domains, such as finance, marketing, logistics and many others. With the increasing availability of large datasets and the uninterrupted growth in computing capabilities, the need to extract meaningful information and patterns has become more critical than ever.

Unsupervised learning, a branch of machine learning whose objective is to gain insights into the data without relying on labeled examples or predefined outputs, plays a crucial role in addressing these challenges. One of the fundamental primitives of unsupervised learning is clustering, which aims at partitioning data points into distinct groups such that points within the same cluster are more similar to each other than to those in other clusters. Clustering has found applications in various fields, including image analysis, customer segmentation, anomaly detection, recommender systems and many more.

With the expanding necessity of working with massive amounts of data, traditional clustering algorithms became impractical. Sequential clustering algorithms often require to work with the entire dataset, which is not feasible when dealing with vast volumes that exceed the memory capacity of a single machine. Moreover, big data is frequently generated in a continuous streaming fashion, with data arriving sequentially and requiring real-time processing, as opposed to being stored and processed in batches. This sparked the need to develop new techniques that could handle the huge amounts of data efficiently and adaptively.

The streaming model, as well-known in the literature [12], offers a powerful paradigm for processing massive, continuously arriving data streams. Unlike batch processing, where the entire dataset is available upfront, streaming algorithms operate on data that arrives in small portions, or mini-batches, and is processed in real-time. This model has gained significant attention due to its ability to handle efficiently the velocity and volume challenges of big data. By processing data in a streaming fashion, it became possible to analyze large-scale datasets incrementally, avoiding the need to store and load the entire dataset into memory.

In the context of clustering [1, 2], the streaming model is particularly relevant and valuable. Streaming clustering algorithms can handle continuous data streams, adapt to changes in the data distribution, and provide real-time insights into the evolving clusters. These algorithms typically employ approximation techniques and memory-efficient data structures to update the set of centers or other relevant parameters on-the-fly. By continuously processing the incoming data, streaming clustering algorithms can maintain up-to-date cluster representations and capture the underlying structures in real-time.

The sliding window model [9] is a variation of the streaming model that is particularly useful for handling big data in real-time analysis. In this model, data is processed as a continuous stream, but instead of considering it in its entirety, a fixed-size window is maintained, and only the most recent data within the window is considered for analysis. The window moves along with the arrival of new data, discarding older data points that fall outside the window. By applying clustering algorithms to the data within the sliding window, it becomes possible to identify and track evolving clusters or patterns over time. This is particularly relevant in domains where data evolves dynamically, such as social media analysis, sensor data processing, or network traffic monitoring. The sliding window model enables real-time clustering, facilitating the extraction of valuable insights and timely responses to changing data patterns in large-scale streaming data scenarios.

One clustering task that has been thoroughly researched both in the sequential and the big data settings is the *k-Center* Problem, also called *Facility Location* problem. Given a ground set of points W from a general metric space and an integer $k < |W|$, the objective of this task is to find a subset S of W of size k, called centers, which minimizes the maximum distance between the points in W and the set S. Since this problem is known to be NP-Hard, many approximation algorithms have been designed for the different computing models cited above [1, 8, 11, 14, 15].

As machine learning algorithms increasingly impact various aspects of our society, from hiring decisions to loan approvals and many others, it has become crucial to address the ethical implications and unintended consequences of these systems. For this reason, extensive research has gone into designing fairness-aware algorithms and into transforming already existing ideas into procedures capable of returning fair solutions. One area where fairness concerns have gained significant attention is algorithmic clustering, both in the sequential setting [5, 6] and in the case of massive amounts of data [13, 7]. Traditional clustering algorithms often overlook the potential for bias and unfairness in their results, since their only focus is optimizing a certain objective cost function. Fair clustering, instead, aims to ensure that clustering algorithms not only produce accurate and meaningful results but also consider fairness criteria and mitigate potential biases. This is usually done by introducing constraints in the kind of solutions that can be chosen.

A constrained version of the very well known k-Center problem is the *Matroid Center* problem, which uses the mathematical concept of matroid to model additional constraints and limit the possible solutions for a given instance. In particular, given a matroid $M_W = (W, I_W)$ with a set of points W from a metric space as ground set, the objective of this constrained task is to choose an independent set $S \in I_W$ of the selected matroid which minimizes the maximum distance between the points in W and S. The Matroid Center problem can work with all the different types of matroids, from the most specific to the most general one. If we decide to limit ourselves to the use of the *Partition Matroid*, where the ground set of points W is partitioned into a set of categories A and a subset S of W is considered an independent sets only if it contains at most a certain number of points for each category in A, then we can call this specialized case of Matroid Center problem also *Fair k-Center* (FKC) problem.

At the moment, for what concerns the sliding window model, the literature contains very good algorithms for the *k-Center* problem [8, 14, 15], but does not provide good approximations for the more general *Fair k-Center* and *Matroid Center* problems. For this reason, we decided to embark on the search of good solutions for both these problems. In order to facilitate ourselves, we will first investigate the FKC problem alone and then, once a good approximation has been found, try to adapt the solution obtained for that particular case to more general settings. As we will see later, this will translate to generalizing from the partition matroid case to the transversal matroid case.

It is in this context that we give some more details on the problems and the model used by looking at the state of the art for the various cases.

1.1 Previous work

As we mentioned previously, approximation algorithms for the Fair k-Center and the Matroid Center problems under the sliding window model have yet to be developed as of the time of writing. For this reason, we will not be able to compare directly the algorithms we designed with other state of the art solutions for the same problem and the same model. Still, we present here the best solutions known to date respectively for the unconstrained k-Center problem under the sliding window model and for the Matroid Center problem in streaming settings. This way, later in the chapters, we will be able to appreciate how close the results contained in this thesis are with respect to the approximation factor to the best solutions for the other models. Moreover, we will be able to compare in term of performances our solutions with the best known for the k-Center problem in the sliding window model and see how well we dealt with the additional constraints.

The k-Center clustering problem has been studied under multiple computational models. In the sequential model the problem is known to have an efficient polynomial 2-approximation and is known to be $(2-\epsilon)$-inapproximable unless $P = NP$ [11]. Regarding the streaming and MapReduce models, the best known solutions were developed by [1], which contains a $(2+\epsilon)$-approximation for both cases, with $\epsilon \in (0,1)$ a user-defined accuracy parameter. Instead, for what concerns the sliding window model, the state of the art was developed by [14], which contains a $(2+\epsilon)$-approximation with working memory $O\left(k\frac{\log(\Delta)}{\log(1+\beta)}\left(\frac{32(1+\beta)}{\epsilon}\right)^{D_{W_t}}\right)$, where $\beta > 0$ is a chosen parameter, Δ is the aspect ratio of the stream and D_{W_t} is the doubling dimension for the window. As we can see, in the last case the working memory is bounded by a function of the doubling dimension of the window D_{W_t}, which means that for low dimensionality data we can obtain very good solutions with high efficiency. It is important to note that [14] does not use the doubling dimension D_{W_t} in its procedures, but only to carry out the complexity analysis, which means that it can adapt obliviously to the parameter.

On the other hand, for what concerns the Matroid Center problem, in the sequential settings we know the problem is 3-approximable [3, 4]. Instead, regarding the streaming model, the $(3 + \epsilon)$-approximation algorithm contained in [2] represents the state of the art for the problem. As in the case of [14], the algorithm blindly adapts to the dimensionality of the data, while its analysis was carried out in terms of the doubling dimension of the stream.

1.2 Summary of contributions

The contributions of this work are the following:

- We review the known approximation algorithms present in the literature representing the state of the art for the k-Center problem under the sliding window model and the state of the art for the Matroid Center problem in streaming settings.
- We study the Matroid Center problem under the sliding window model and, in particular, we investigate the cases of partition and transversal matroids.
- For the partition matroid case, we design the first $(3 + \epsilon)$-approximation algorithm for the sliding window model capable of working with general metric spaces and adapting obliviously to the dimensionality of the data, as captured by the concept of doubling dimension. It is important to highlight that this solution not only represents the state of the art for the Matroid Center problem under partition matroid constraints in sliding windows, but is also the first algorithm ever to approximate the problem for this computational model.
- We formally prove the correctness of the proposed algorithm and demonstrate it achieves a $(3 + \epsilon)$ approximation factor, an accuracy comparable to the best known algorithm for the sequential settings. Furthermore, we present tight asymptotic bounds on both the working memory and the running time of its procedures in terms of the doubling dimension of the window, which means that the working memory for low dimensionality data is proven to be drastically inferior to the size of the window.

- We further improve the previous result by developing two additional ideas capable of dealing with a more general type of matroid called transversal matroid. In fact, we show that the first refined algorithm works correctly with the transversal matroid and obtains formally a $(3 + \epsilon)$-approximation at the cost of slightly worst time and space requirements. Finally, we present a second refined heuristic algorithm capable of working with the transversal matroid, obtaining the same approximation factor with asymptotically better performances and whose correctness has only been conjectured with the help of some empirical examples.

1.3 Structure of the thesis

The thesis is organized as follows. Chapter 2 describes the notions necessary to present the central results of this thesis. Chapter 3 describes and analyzes our $(3 + \epsilon)$-approximation algorithm for the k-Center problem under partition matroid constraints, also called Fair k-Center problem. Chapter 4 presents two improved versions of the standard algorithm, able to obtain the same approximation factor under the more general transversal matroid constraints. Finally, Chapter 5 terminates the thesis with some concluding remarks.

2 Preliminaries

Before moving to the actual algorithms proposed in the paper, we need to introduce some important concepts used in order to formalize the problems, the framework and the notation employed in the proofs. We will start by introducing the computational framework. Then, we will present some mathematical concepts that will be useful to formalize the problems and to carry out the proofs of correctness and complexity. Finally, we will introduce the actual problems we are trying to solve and present the state of the art under some interesting models, starting from the simpler k-Center clustering problem and then moving to more generalized versions.

2.1 Sliding Window Model

We start by introducing the computational scenario in which the algorithms were developed. It is an important variant of the streaming model, called sliding window model, presented in [9].

In the standard streaming framework, the computation is performed by a single processor with a small working memory and the input is provided as a continuous, possibly unbounded, stream of objects (points, in our case) arriving one at each time step, which is usually too large to fit in the working memory. Under the sliding window model, at each time t, a solution to the problem of interest should be computable for the pointset W_t represented by the last N points arrived in the stream, where N, referred to as window length, is a predetermined value known to the algorithm. More formally, for each input point p, let $t(p)$ denote its arrival time. At any time t, we have that $W_t = \{p \mid 0 \leq t - t(p) < N\}$. Since N can still be much larger than the working memory size, the goal in this setting is to guarantee the quality of the solution while storing an amount of data substantially smaller than the window length.

At this point we introduce two additional quantities for the stream S and the window W_t which will be essential in the analysis of the two algorithms. Consider now a stream S of points from a metric space with distance function $dist(\cdot,\cdot)$, and with a sliding window of length N. We define the aspect ratio Δ of S as the ratio between the maximum distance and the minimum distance of any two points of S ($\Delta = \frac{maxDist}{minDist}$). Similarly, at any time t, we define the aspect ratio Δ_{W_t} of the current window W_t as the ratio between the maximum and the minimum distance of any two distinct points in W_t ($\Delta_{W_t} = \frac{maxDist_{W_t}}{minDist_{W_t}}$).

The time and space performance of our algorithms will be analyzed in terms of k, N, Δ_{W_t}, ϵ, β and of the dimensionality of the points in the current window. Since the target applicability of our algorithms to arbitrary metric spaces, we will make use of the following, general notion of dimensionality.

2.1.1 Doubling Dimension

Definition 2.1 (Metric Space) *A metric space is an ordered pair (W, d) where W is a ground set of points and $dist(\cdot,\cdot)$ is a distance function on W, i.e. a function $dist : S \times S \longrightarrow \mathbb{R}$, such that the following properties are true $\forall x, y, z \in S$:*

- $dist(x, y) \geq 0$;
- $dist(x, y) = 0$ *only if* $x = y$;
- $dist(x, y) = dist(y, x)$ *(symmetry);*
- $dist(x, z) \leq dist(x, y) + dist(y, z)$ *(triangle inequality).*

Let W_t denote the set of points from a metric space. For any $x \in W_t$ and $r > 0$, let the ball of radius r centered at x, denoted as $B(x, r)$, be the subset of points of W_t at distance at most r from x. The doubling dimension of W_t is the smallest D such that any ball $B(x, r)$, with $x \in W_t$, is contained in the union of at most 2^D balls of radius $\frac{r}{2}$ suitably centered at points in W_t. The following important fact, which we will use in the analysis is:

Lemma 2.2 (Doubling Dimension) *Let W_t be a set of points from a metric space and let $Y \subseteq W_t$ be such that any two distinct points $a, b \in Y$ have pairwise distance $dist(a,b) > r$. If W_t has doubling dimension D, then for every $R \geq r$ and any point $x \in W_t$, we have $|B(x,R) \cap Y| \leq \left(\frac{4R}{r}\right)^D$.*

The doubling dimension D_{W_t} for the current window W_t will be required only in the analysis of time and space complexity for the algorithms, but will never be used explicitly by the proposed solutions.

2.2 Matroid, general definition and specializations

After introducing the computational model, we move to the definition of matroid. This mathematical concept is used to formalize the additional constraints added to the k-Center problem.

Let W be a ground set (a window in our case) of elements from a metric space with distance function $dist(\cdot,\cdot)$ satisfying the triangle inequality. A matroid on W is a pair $M_W = (W, I_W)$, where I_W is a family of subsets of W, called independent sets, satisfying the following properties:

- the empty set is independent ($\emptyset \in I_W$);
- every subset of an independent set is independent (*hereditary property*, $\forall X' \subseteq X : X \in I_W \implies X' \in I_W$);
- if $A, B \in I_W$ and $|A| > |B|$, then $\exists x \in A \setminus B$ such that $B \cup \{x\} \in I_W$ (*augmentation property*).

An independent set is maximal if it is not properly contained in another independent set ($A \in I_W, \forall x \in W \setminus A : A \cup \{x\} \notin I_W$). A basic consequence of the augmentation property is that all the maximal independent sets for a matroid $M_W = (W, I_W)$ must have the same size, which we denote as rank of the matroid $rank(M_W)$. This notion of maximality can be naturally extended to any subset of the ground set. Namely, for $W' \subseteq W$, an independent set $A \subseteq W'$ of maximum size among all independent sets contained in W' is called a maximal independent set of W' and all maximal independent sets of W' have the same size. We define the rank of a subset $W' \subseteq W$, denoted as $rank(W')$, to be the size of a maximal independent set in W'. An important property of the rank function is called submodularity: for any $A, B \subseteq W$ it holds that $rank(A \cup B) + rank(A \cap B) \leq rank(A) + rank(B)$. Given the previous notions, we can define more formally the rank of the matroid $rank(M_W)$ as the rank of its ground set $rank(W)$. Given a matroid $M_W = (W, I_W)$ and a subset $W' \subseteq W$, we define a restriction of M_W to W' as $M_{W'} = (W', I_{W'})$ where $I_{W'} = \{X \cap W' \mid X \in I_W\}$. It is easy to see that $M_{W'}$ is also a matroid.

Next, we present a lemma taken from [2] that provides a useful property of matroids which will be used to prove the correctness of the proposed algorithms. Its proof is contained in the paper cited above.

Lemma 2.3 (Extended augmentation property) *Let $M_W = (W, I_W)$ be a matroid. Consider the independent set $A \in I_W$, a subset $W' \subseteq W$ and an independent set $B \subseteq W'$ which is maximal within W'. If $\exists y \in W' \setminus A$ such that $A \cup \{y\} \in I_W$, then $\exists x \in B \setminus A$ such that $A \cup \{x\} \in I_W$.*

Finally, before moving to the presentation of the problems, we need to introduce two particular types of matroid called partition matroid and transversal matroid [10]. They will be used to generalize the k-Center problem by constraining the possible subsets of points we can chose as centers. In fact, by using the partition matroid we will be able to allow only solutions that select the set of centers in a fair and distributed fashion between a set of given categories. Similarly, by using the transversal matroid we will allow only solutions that take at most one point for each category in a set of (possibly non-disjoint) categories and do not take the same point as representative of two different groups simultaneously.

Definition 2.4 (Partition matroid) *Let $W_1, ..., W_\ell$ be a partition of the ground set W ($\cup_{i=1}^{\ell} W_i = W, \forall 1 \le i < j \le \ell : W_i \cap W_j = \emptyset$). Moreover, let $k_1, ..., k_\ell$ be positive integers. Then the family*

$$I_W = \{X \subseteq W : \forall_{i=1}^{\ell} |X \cap W_i| \le k_i\} \tag{2.1}$$

satisfies the independence axioms. The corresponding matroid $M_W = (W, I_W)$ is called a partition matroid.

Definition 2.5 (Transversal matroid) *Let $A = \{A_1, ..., A_\ell\}$ be a family of (possibly non-disjoint) categories of the ground set W, with $W = \cup_{i=1}^{\ell} A_i$, and consider the bipartite graph $(W, A; E)$ where E consists of all edges $\{s_i, A_j\}$ with $s_i \in A_j$, for $1 \le i \le |W|$ and $1 \le j \le \ell$. Define I_W as the family of subsets $X \subseteq W$ corresponding to the left endpoints of some matching in the above graph. Then, $M_W = (W, I_W)$ is a transversal matroid based on W.*

It is easy to see that the partition matroid can be viewed as a specific case of transversal matroid, where, for each category $i = 1, \ldots, \ell$ of the partition matroid, we have k_i equivalent categories of points in A.

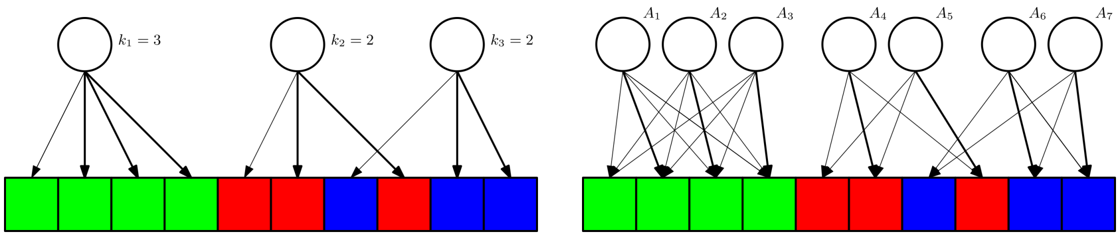


Figure 2.1: On the left the partition matroid with colored categories, on the right the equivalent transversal matroid.

2.3 Definitions of the problems

At this point we have all the notions needed in order to introduce the actual problems we are trying to solve. We start by presenting the basic k-Center clustering problem.

2.3.1 k-Center problem

Consider a pointset W from some metric space with distance function $dist(\cdot, \cdot)$. For any point $p \in W$ and any subset $C \subseteq W$ we use the notation

$$dist(p, C) = \min_{q \in C} dist(p, q) \tag{2.2}$$

and define the radius of C with respect to W as

$$r_C(W) = \max_{p \in W} dist(p, C). \tag{2.3}$$

For a positive integer $k < |W|$, the k-Center problem requires to find a subset $C \subseteq W$ of size k which minimizes $r_C(W)$. Note that any subset $C \subseteq W$ of size k induces immediately a partition of W into k clusters by assigning each point to its closest center (with ties broken arbitrarily). We say that $r_C(W)$ is the radius of such clustering and define

$$OPT_{k,W} = \min_{C \subseteq W, |C| = k} r_C(W) \tag{2.4}$$

to denote the radius achieved by an optimal solution to the problem.

In the standard sequential setting, it is well-known that for general metric spaces the k-Center problem is NP-Hard, admits a 2-approximation algorithm and, for any $\epsilon > 0$, it is not $(2-\epsilon)$-approximable unless $P = NP$. The well-known greedy sequential algorithm by Gonzalez [11], provides a 2-approximation to the k-Center problem running in $O(|W|k)$ time.

The current state of the art for sliding window, developed by [14], maintains information about a carefully selected subset of points of the current window W_t, from which, at any time t, a coreset $T \subseteq W_t$ can be extracted and a solution to the k-Center problem can be efficiently computed by running a sequential (approximation) algorithm on T.

2.3.2 Robust Matroid Center problem

The Robust Matroid Center problem is a variant of the k-Center problem with outliers. Its definition makes use of the concept of matroid previously introduced and can be formalized as follows.

Let $M_W = (W, I_W)$ be a matroid with rank k defined over the set of points W and let z be an integer with $0 \le z \le |W|$. The *Robust Matroid Center* (RMC) problem on M_W with parameter z requires determining an independent set $S \in I_W$ minimizing

$$r(S, W, z) = \min_{X \subseteq W : |X| \ge |W| - z} \max_{i \in X} dist(i, S) \tag{2.5}$$

We use the tuple $(M_W = (W, I_W), z)$ to denote an instance of RMC and let $OPT_{M_W, z}$ denote the cost of its optimal solution. It is immediate to see that the objective function $r(S, W, z)$ corresponds to the $(|W| - z)$-th smallest distance of a point of W from S. In other words, the best solution is allowed to ignore the contribution of the z most distant points, which can be regarded as outliers. Note that if the matroid (W, I_W) has a rank k, any feasible solution $S \in I_W$ has size at most k. In the case where $z = 0$ we denote $OPT_{M_W} = \min_{S \in I_W} r(S, W, 0)$ as the cost of the optimal solution. Also, note that the standard k-Center problem is a special case of the RMC problem where $z = 0$ and the set I_W of independent sets consists of all subsets of size at most k ($I_W = \{X \subseteq W : |X| \le k\}$).

The state of the art on sequential solutions for the problem is the 3-approximation algorithm presented in [3, 4].

2.3.3 Fair k-Center problem, partition and transversal matroids

The special case of *Partition Matroid* can be used to model fairness constraints where the points of W are naturally subdivided into $m \leq k$ groups and fair solutions to k-Center are sought which include at most k_i points from the i-th group, for given k_i's such that $\sum_{i=1}^{m} k_i = k$. The *Fair k-Center* (FKC) problem is a special case of the RMC problem where $z = 0$ and the partition matroid $M_W = (W, I_W)$ is used.

It is important to note that we will also investigate another special case of the RMC problem similar to FKC where, instead of the partition matroid, the transversal matroid is used, since it models an even more general case of k-Center clustering and we believe it is analogously approximable in the sliding window model.

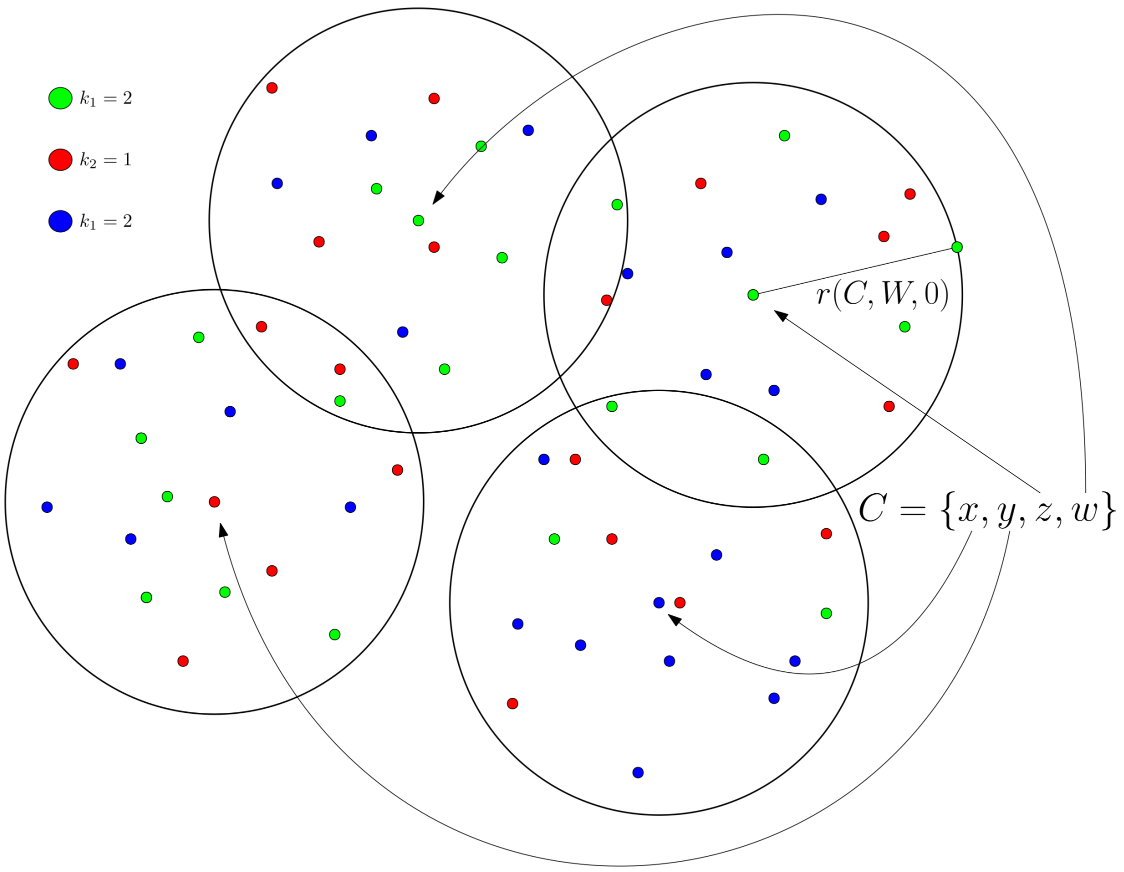


Figure 2.2: Example of Fair k-Center clustering with centers C.

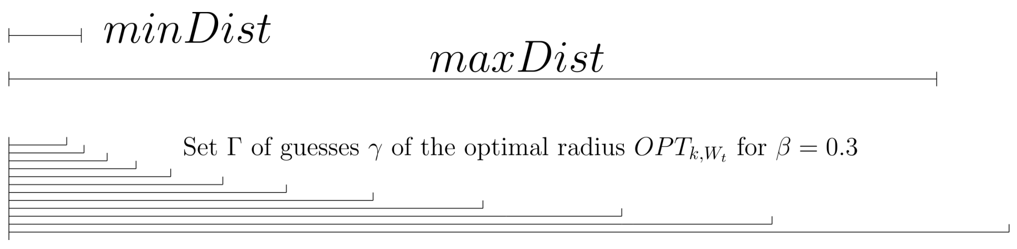
2.4 Relevant existing algorithms

In this section, we will provide a short overview of the current state of art solutions for the k-Center problem under the sliding window model and the Matroid Center problem in streaming settings. The main reasons we present these solutions is that they were a great source of inspiration for our proposed algorithms and were especially helpful in understanding how to deal with the constraints of both the model and the problem.

2.4.1 k-Center in sliding windows

There are two pivotal works related to the approximation of the k-Center problem under the sliding window model. The first of the two papers, written by Cohen-Addad [8], provided a $(6 + \epsilon)$-approximation and presented novel techniques for dealing with sliding windows and guaranteeing a maximum distance between the points in the window and the set of centers. The second article, developed by Pellizzoni-Pietracaprina-Pucci [14], built upon those ideas and was capable of improving the final result to a $(2 + \epsilon)$-approximation which is, as of today, the state of the art for k-Center clustering. This improvement was obtained at the expense of a small bump in storage needs and running time performances, as highlighted by the experiments contained in the paper. Since the latter is an improvement of the former, we will limit ourselves to the presentation of the most comprehensive one [14].

The paper used the system of guessing of the optimal radius for the problem initially presented by [8].


$minDist$
$maxDist$
Set Γ of guesses γ of the optimal radius OPT_{k,W_t} for $\beta = 0.3$

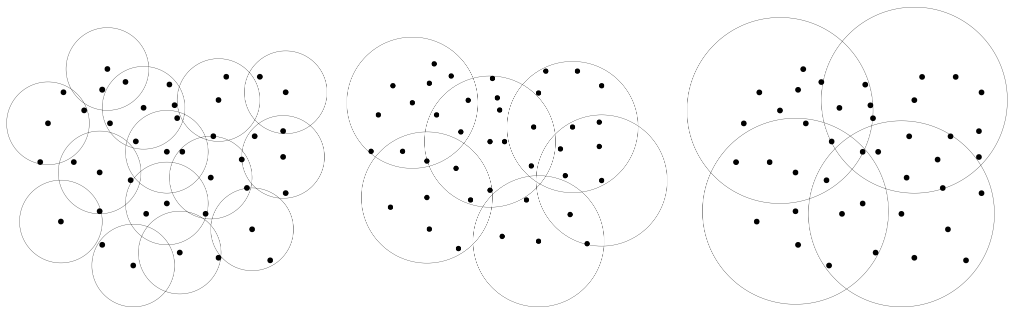
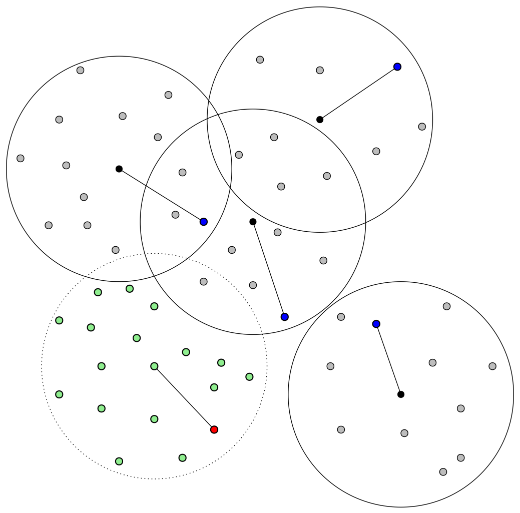


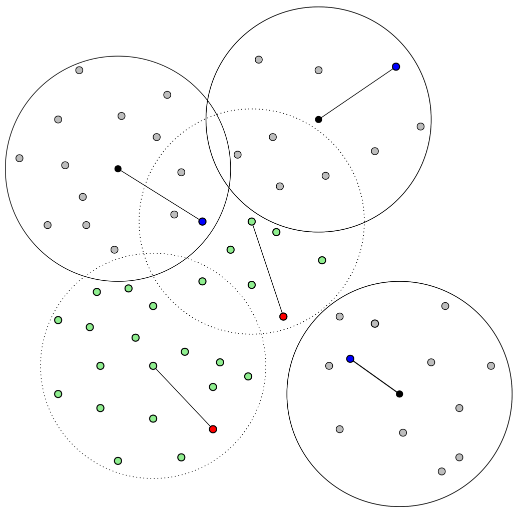
Figure 2.3: Example of the geometric progression Γ with parameter $\beta = 0.3$ used in [8, 14] to provide a set of guesses of the optimal radius OPT_{k,W_t} for the k-Center problem on the window W_t.

For each guess of the optimal radius the algorithm kept six different support structures. Three validation sets AV_γ, RV_γ and OV_γ and three additional coreset sets. These validation sets were used to decide if their guess γ was in fact a good one or a bad one and, in the second case, from which time step the guess could be taken into consideration again knowing the number of centers k. The coreset sets A_γ, R_γ and O_γ, instead, were used to compute the final solution.

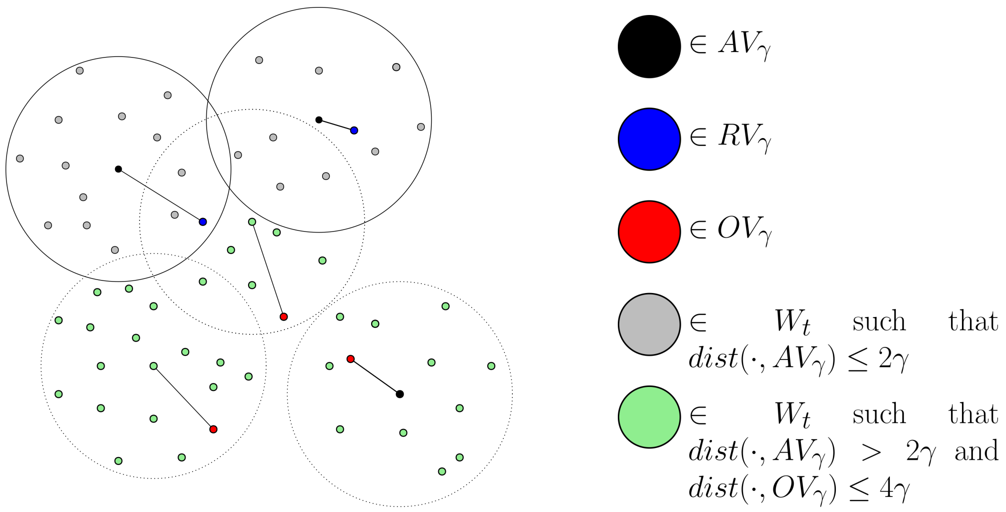
We conclude this section with a graphical example on how the three structures AV_γ, RV_γ and OV_γ used by [8] and [14] should look like when the guess γ is a good one i.e. $|AV_\gamma| \leq k$.



(a) Initially, only one point exited AV_γ, all the points in W_t are all still at distance at most 4γ from $RV_\gamma \cup OV_\gamma$



(b) Another point exits AV_γ, its representative becomes an orphan, the points are still covered by $RV_\gamma \cup OV_\gamma$, a representative is swapped



$\in AV_\gamma$
$\in RV_\gamma$
$\in OV_\gamma$
$\in W_t$ such that $dist(\cdot, AV_\gamma) \leq 2\gamma$
$\in W_t$ such that $dist(\cdot, AV_\gamma) > 2\gamma$ and $dist(\cdot, OV_\gamma) \leq 4\gamma$

(c) A third point exits AV_γ and another point becomes the new representative for an attraction v-point still alive

Figure 2.4: Example of clustering for a given γ using the algorithm proposed by Cohen-Added [8] and subsequently improved by Pellizzoni-Pietracaprina-Pucci [14]

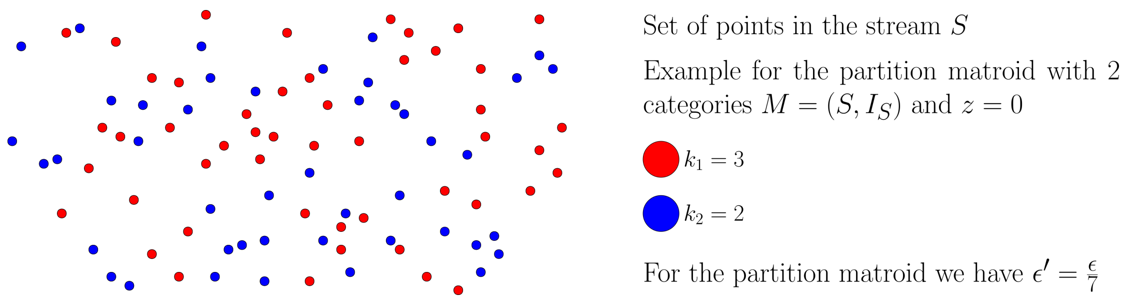
2.4.2 Matroid Center in streaming settings

For what concerns the streaming model, the best approximation algorithm for the Matroid Center problem is given by [2]. In particular, in this paper they obtained a $(3 + \epsilon)$ approximation factor for the *Robust Matroid Center* problem, which is a generalized version of the problem we are interested in also capable of working with outliers.

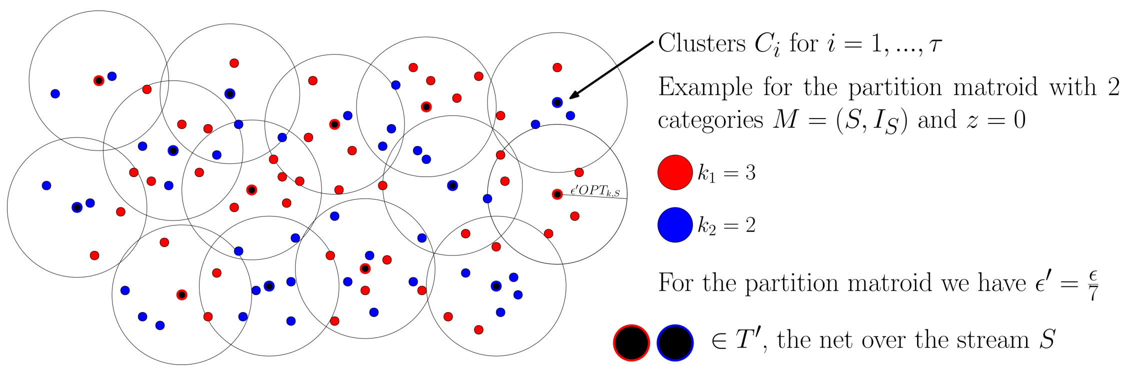
In order to better understand the solution, we will now terminate this section with a short presentation of the algorithm designed by [2]. The proposed algorithm can be divided in two separate parts. In the first part, the aim is finding an ϵ-coreset for the stream on which, in the second part, a good sequential solution can be computed. Since the sequential computation of the solution requires simply to run an efficient 3-approximation algorithm [3, 4] on the ϵ-coreset, we will focus on how this coreset is obtained.

Let us now consider the simplest case of partition matroid as presented in Definition 2.4 and let us use some graphical examples to facilitate the understanding of the various steps needed to create the ϵ-coreset:

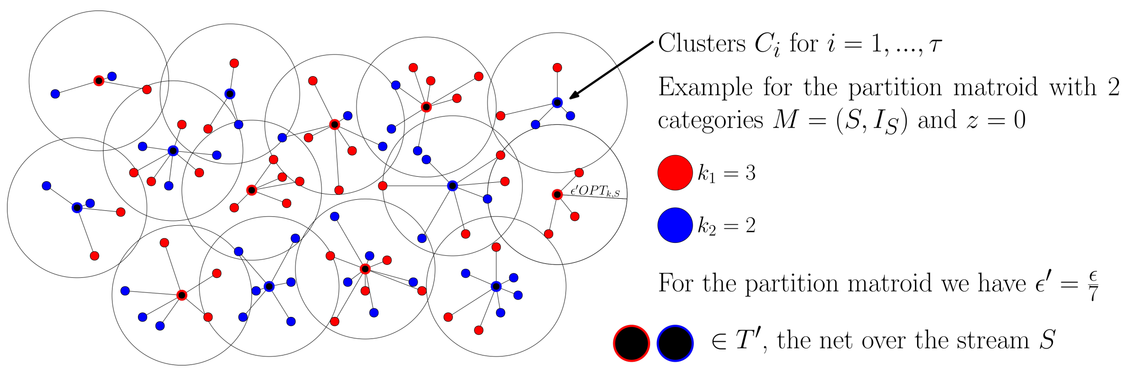
- Initially, given the stream S of points, we find an estimate of the optimal radius $OPT_{k,S}$ for the standard k-Center problem;


Set of points in the stream S
Example for the partition matroid with 2 categories $M = (S, I_S)$ and $z = 0$
$k_1 = 3$
$k_2 = 2$
For the partition matroid we have $\epsilon' = \frac{\epsilon}{7}$

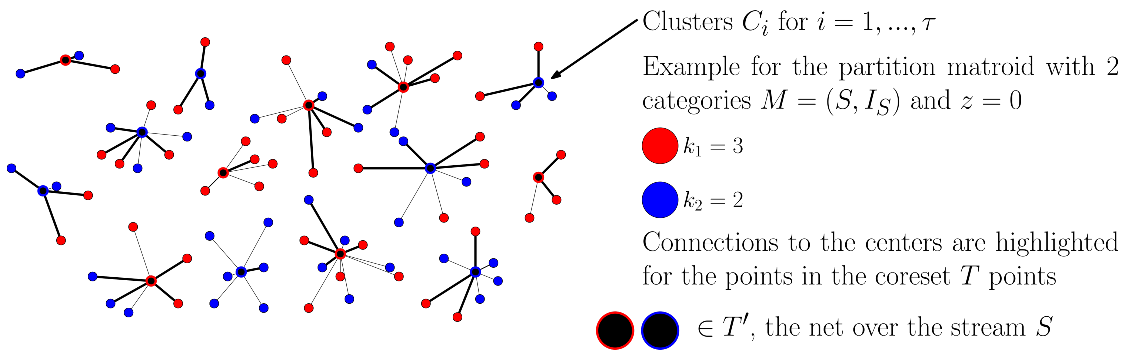
- Once $OPT_{k,S}$ is found, we chose a subset (a net) T' of points at distance greater than $\epsilon' OPT_{k,S}$ from each other, where ϵ' is a parameter chosen shrewdly to obtain the ϵ-coreset ($\epsilon' = \frac{\epsilon}{1+2\cdot\alpha}$ where $\alpha = 3$);


Clusters C_i for $i = 1, ..., \tau$
$\epsilon' OPT_{k,S}$
Example for the partition matroid with 2 categories $M = (S, I_S)$ and $z = 0$
$k_1 = 3$
$k_2 = 2$
For the partition matroid we have $\epsilon' = \frac{\epsilon}{7}$
$\in T'$, the net over the stream S

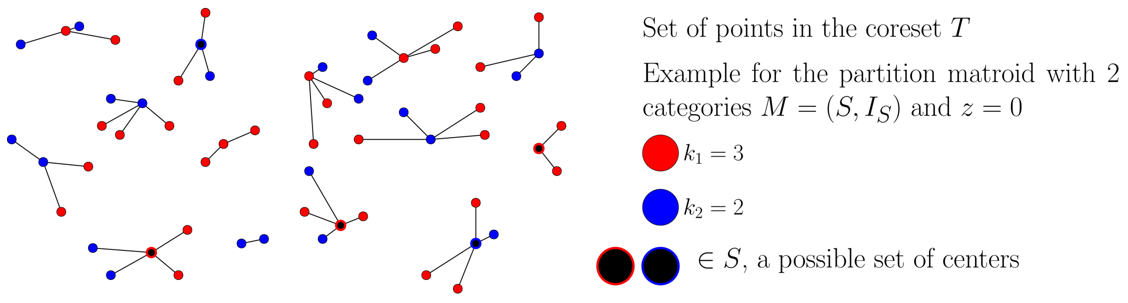
- We cluster the points in the stream using as centers the subset T' chosen previously;


Clusters C_i for $i = 1, ..., \tau$
$\epsilon' OPT_{k,S}$
Example for the partition matroid with 2 categories $M = (S, I_S)$ and $z = 0$
$k_1 = 3$
$k_2 = 2$
For the partition matroid we have $\epsilon' = \frac{\epsilon}{7}$
$\in T'$, the net over the stream S

- For each cluster of points, we compute a maximal independent set for the (partition) matroid;


Clusters C_i for $i = 1, ..., \tau$
Example for the partition matroid with 2 categories $M = (S, I_S)$ and $z = 0$
● $k_1 = 3$
● $k_2 = 2$
Connections to the centers are highlighted for the points in the coreset T points
● ● $\in T'$, the net over the stream S

- We return as ϵ-coreset T the union of these maximal independent sets.


Set of points in the coreset T
Example for the partition matroid with 2 categories $M = (S, I_S)$ and $z = 0$
● $k_1 = 3$
● $k_2 = 2$
● ● $\in S$, a possible set of centers

3 Sliding window algorithm for k-Center under partition matroid constraint

3.1 Description of the algorithm

The solution we developed for the FKC problem is built upon previous results already present in the literature. In particular, we based our work on [14] for what concerns the actual procedures to run and the support structures used. Moreover, we exploited the result presented by [2] in order to generalize the solution from the k-Center to the Matroid Center problem.

As in [14], our algorithm works on guesses of the optimal radius OPT_{k,W_t} for the k-Center problem on the window W_t. In order to limit the space and time required to run, these guesses are chosen from a geometric progression of type $\Gamma = \{(1+\beta)^i : \lfloor \log_{(1+\beta)} minDist \rfloor \leq i \leq \lceil \log_{(1+\beta)} maxDist \rceil\}$ for a given parameter $\beta > 0$. For each of these guesses $\gamma \in \Gamma$ we use six support structures, which are divided in:

- three validation sets AV_γ, RV_γ and OV_γ, used to decide which γ is the best guess of OPT_{k,W_t};
- three coreset sets A_γ, R_γ and O_γ, used to obtain a good representative of W_t on which the sequential algorithm $\mathcal{A}$ for the FKC problem can be run and return a $(3+\epsilon)$-approximate solution for W_t.

With the intention of facilitating the description of the procedures, we will refer to the points contained within the validation sets as *v-points*, while we will refer to the ones in the coreset sets as *c-points*.

Among the validation sets, we have:

- The set AV_γ of *attraction v-points*, which contains a covering net of points at distance greater than 2γ and which, when γ is a valid guess of the optimal radius OPT_{k,W_t} for the k-Center problem, represents a possible solution of radius at most 2γ ($r_{AV_\gamma}(W_t) \leq 2\gamma$);
- The set RV_γ of *representative v-points* $repV_\gamma(v)$, one for each $v \in AV_\gamma$, representing the newest points at distance not greater than 2γ from their respective v ($dist(v, repV_\gamma(v)) \leq 2\gamma$);
- The set OV_γ containing old representative v-points, also called *orphan v-points*, whose respective attraction v-points expired or got deleted from AV_γ at time steps previous to the current one.

On a similar note, all the coreset sets A_γ, R_γ and O_γ closely mirror the behaviour of their respective validation sets, but with a few important distinctions. In particular, as was done in [14], instead of requiring the attraction c-points to be at distance greater than 2γ, we require them to be at distance exceeding $\frac{\delta\gamma}{2}$, where δ is a chosen parameter. By setting δ cleverly, we can obtain, at the expense of a slight deterioration in performance, a much finer coverage for the points in the window W_t and consequently a much better coreset. The second important difference is related to the meaning of the coreset sets R_γ and O_γ. In our case, in contrast with what was done in [14] where, for each $a \in A_\gamma$, they keep a representative c-point $repC_\gamma(a)$, we maintain a set of points at distance at most $\frac{\delta\gamma}{2}$ from each attraction c-point. This is done so to have, not a single good alternative close to each $a \in A_\gamma$, but a good set of them. Specifically, with inspiration from [2], we decided to keep an independent set of points close to each of these $a \in A_\gamma$ from which later a coreset can be extracted and a solution computed sequentially. Since all the points can be divided into categories, we will denote $repC^i_\gamma(a')$ as the subset of representatives $repC_\gamma(a')$ for $a' \in A_\gamma$ of category i (note that $\forall i = 1, ..., \ell : |repC^i_\gamma(a')| \leq k_i$).

To sum up, our proposal consists of two procedures:

- UPDATE(p), which describes the processing that needs to be done at each time step t when a new point p enters the window W_t and another point expires;

- QUERY(), which, if invoked at time t, returns a $(3+\epsilon)$-approximate solution for the FKC problem on W_t by running the sequential algorithm $\mathcal{A}$ on the coreset $T = R_\gamma \cup O_\gamma$.

In order to run correctly the procedures need some variables. These are set to $\delta = \frac{\epsilon'}{(1+\beta)}$ and $\epsilon' = \frac{\epsilon}{(1+2\alpha)}$, given the parameter $\epsilon \in (0,1)$ and given an α-approximation sequential algorithm for the FKC problem. In conclusion, the pseudocode for the two procedures is the following:

Algorithm 1 UPDATE(p)

```
1:  Let $p \in W_i$ and let $repC^i_\gamma(p) = repC_\gamma(p) \cap W_i$
2:  for each $\gamma \in \Gamma$ do
3:      for each expired $v \in AV_\gamma$ do
4:          $AV_\gamma = AV_\gamma \setminus \{v\}$
5:          $RV_\gamma = RV_\gamma \setminus \{repV_\gamma(v)\}$
6:          $OV_\gamma = OV_\gamma \cup \{repV_\gamma(v)\}$
7:      for each expired $a \in A_\gamma$ do
8:          $A_\gamma = A_\gamma \setminus \{a\}$
9:          $R_\gamma = R_\gamma \setminus repC_\gamma(a)$
10:         $O_\gamma = O_\gamma \cup repC_\gamma(a)$
11:     for each expired $q \in OV_\gamma$ do
12:         $OV_\gamma = OV_\gamma \setminus \{q\}$
13:     for each expired $o \in OV_\gamma$ do
14:         $O_\gamma = O_\gamma \setminus \{o\}$
15:     $EV = \{v \in AV_\gamma \mid dist(p,v) \leq 2\gamma\}$
16:     $E = \{a \in A_\gamma \mid dist(p,a) \leq \frac{\delta\gamma}{2}\}$
17:     if $EV == 0$ then
18:         $AV_\gamma = AV_\gamma \cup \{p\}$
19:         $repV_\gamma(p) = p$
20:         $RV_\gamma = RV_\gamma \cup \{repV_\gamma(p)\}$
21:         if $|AV_\gamma| > k+1$ then
22:             $v_{old} = \arg\min_{v \in AV_\gamma} TTL(v)$
23:             $AV_\gamma = AV_\gamma \setminus \{v_{old}\}$
24:             $RV_\gamma = RV_\gamma \setminus \{repV_\gamma(v_{old})\}$
25:             $OV_\gamma = OV_\gamma \cup \{repV_\gamma(v_{old})\}$
```

```
26:         **if** $|AV_\gamma| > k$ **then**
27:             $t_{min} = \min_{v \in AV_\gamma} TTL(v)$
28:             **for each** $a \in A_\gamma$ **do**
29:                 **if** $TTL(a) < t_{min}$ **then**
30:                     $A_\gamma = A_\gamma \setminus \{a\}$
31:                     $R_\gamma = R_\gamma \setminus repC_\gamma(a)$
32:                     $O_\gamma = O_\gamma \cup repC_\gamma(a)$
33:             **for each** $q \in OV_\gamma$ **do**
34:                 **if** $TTL(q) < t_{min}$ **then**
35:                     $OV_\gamma = OV_\gamma \setminus \{q\}$
36:             **for each** $o \in O_\gamma$ **do**
37:                 **if** $TTL(o) < t_{min}$ **then**
38:                     $O_\gamma = O_\gamma \setminus \{o\}$
39:     **else**
40:         **for each** $v \in EV$ **do**
41:             Set $repV_\gamma(v) = p$ in $RV_\gamma$
42:     **if** $E == 0$ **then**
43:         $A_\gamma = A_\gamma \cup \{p\}$
44:         $repC_\gamma(p) = \{p\}$
45:         $R_\gamma = R_\gamma \cup repC_\gamma(p)$
46:     **else**
47:         $a_{add} = \arg\min_{a \in E} |repC^i_\gamma(a)|$ (break ties arbitrarily)
48:         Set $repC_\gamma(a_{add}) = repC_\gamma(a_{add}) \cup \{p\}$ in $R_\gamma$
49:         **if** $|repC^i_\gamma(a_{add})| > k_i$ **then**
50:             $o_{rem} = \arg\min_{o \in repC^i_\gamma(a_{add})} TTL(o)$
51:             Remove $o_{rem}$ from $repC_\gamma(a_{add})$ in $R_\gamma$
```

Algorithm 2 QUERY()

```
1: for increasing values of $\gamma \in \Gamma$ with $|AV_\gamma| \le k$ do
2:     $C = \emptyset$
3:     for each $q \in AV_\gamma \cup RV_\gamma \cup OV_\gamma$ do
4:         if $(C == \emptyset)$ or $dist(q, C) > 2\gamma$ then
5:             $C = C \cup \{q\}$
6:         if $|C| > k$ then
7:             Break and move to the next guess
8:     if $|C| \le k$ then
9:         return $\mathcal{A}(R_\gamma \cup O_\gamma, k)$
```

3.2 Analysis

In this section we will develop the proofs needed in order to guarantee that the algorithm is correct and that it returns a $(3 + \epsilon)$-approximation for the FKC problem. In addition, at the end of this section we will analyze the space and time complexity of the algorithm.

In order to simplify its analysis, we first introduce a general notation that helps us characterize the evolution over time of all the support structures used by the algorithm. We will denote a general structure X at the end of the execution of UPDATE(p) at time step t as X_t. For example, we will specify $AV_{\gamma,t}$, EV_t and $repC_{\gamma,t}(v)$ as the structures AV_γ, EV and $repC_\gamma(v)$ at time t.

Now that the final piece of notation is fixed, we will present first a simple property for the sets of attraction points and then, secondly, a set of invariants needed in order to prove that the distance of each point from $RV_\gamma \cup OV_\gamma$ and from $R_\gamma \cup O_\gamma$ is bounded under some conditions.

Lemma 3.1 *Let $v' = \arg\min_{v \in AV_{\gamma,t}} t(v)$ for a given $\gamma \in \Gamma$ (resp. $a' = \arg\min_{a \in A_{\gamma,t}} t(a)$) and let W_t be the window at time t. Then, we have that $\forall q \in W_t$ with $t(q) \ge t(v')$ (resp. $t(q) \ge t(a')$): $AV_{\gamma,t(q)} \subseteq AV_{\gamma,t} \cup AV_{\gamma,t(v')-1}$ (resp. $A_{\gamma,t(q)} \subseteq A_{\gamma,t} \cup A_{\gamma,t(a')-1}$).*

Proof Since the same argument can be made for both $AV_{\gamma,t(q)}$ and $A_{\gamma,t(q)}$, we only carry out the proof for the former. Consider a generic $\gamma \in \Gamma$. The main reason we can deduce the property specified in the lemma is that the behavior of $AV_{\gamma,t'}$ (how points enter and exit the set) over time t' is equivalent to the behaviour of a queue. In fact, we always remove only the oldest point in $AV_{\gamma,t'}$ at time step t' and add only points newer than all the points already present in the set $AV_{\gamma,t'}$. Thus, since the set of points in a queue at an intermediate time $t(v') - 1 < t(q) \le t$ is a subset of the union of the set of points in the queue at two different instants, we can conclude that $AV_{\gamma,t(q)} \subseteq AV_{\gamma,t} \cup AV_{\gamma,t(v')-1}$. ■

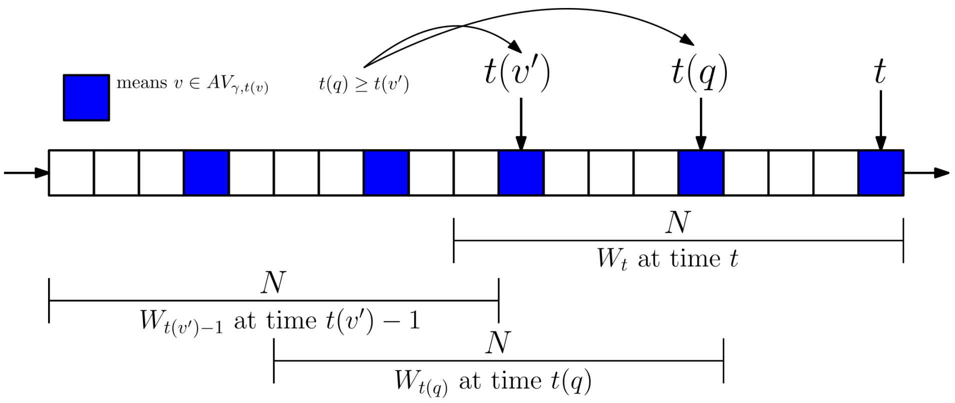

means $v \in AV_{\gamma,t(v)}$
$t(q) \geq t(v')$
$t(v')$
$t(q)$
t
N
W_t at time t
N
$W_{t(v')-1}$ at time $t(v') - 1$
N
$W_{t(q)}$ at time $t(q)$

Figure 3.1: A visual example of the lemma described above

Lemma 3.2 *Let $v' = \arg\min_{v \in AV_\gamma} t(v)$ be the oldest point in AV_γ, if it exists. For every $\gamma \in \Gamma$, the following properties hold after the end of each execution of procedure UPDATE(p), with respect to the window W_t at time step t:*

(1) $\forall q \in W_t$ with $t(q) \ge t(v')$ we have that $dist(q, RV_\gamma \cup OV_\gamma) \le 4\gamma$;

(2) $\forall q \in W_t$ with $t(q) < t(v')$ we have that if $|AV_\gamma| \le k$ then $dist(q, RV_\gamma \cup OV_\gamma) \le 4\gamma$.

Proof Let γ be a generic guess in Γ.

We prove the two properties presented in the lemma by distinguishing on the size of the set $AV_{\gamma,t}$ at time step t:

- If $|AV_{\gamma,t}| = 0$, then it is certain that $W_t = \emptyset$ since the first point in the window is always added to $AV_{\gamma,t}$. In this case the properties are true ($\forall q \in W_t$ with $W_t = \emptyset$).
- If $|AV_{\gamma,t}| \ge 1$, consider $v' = \arg\min_{v \in AV_{\gamma,t}} t(v)$ the oldest point in $AV_{\gamma,t}$ in the window W_t at time t. In this case we can partition the points in the window W_t given the time $t(q)$ they entered:

- $\forall q \in W_t$ with $t(q) \geq t(v')$ we have that $dist(q, RV_\gamma \cup OV_\gamma) \leq 4\gamma$. Assume, by contradiction, that there exists $q' \in W_t$ with $t(q') > t(v')$ such that $dist(q', RV_{\gamma,t} \cup OV_{\gamma,t}) > 4\gamma$ (the point cannot be trivially $q' = v'$). This implies that:

  (A) we have $dist(q', RV_{\gamma,t}) > 4\gamma$ and in turn $dist(q', AV_{\gamma,t}) > 2\gamma$ at time t;

  (B) we have $dist(q', OV_{\gamma,t}) > 4\gamma$ at time t, which means that at time $t(v')-1$ we have $dist(q', RV_{\gamma,t(v')-1}) > 4\gamma$ and $dist(q', AV_{\gamma,t(v')-1}) > 2\gamma$. (points can be in $OV_{\gamma,t}$ only if they where one of the representatives, or their substitutions in close proximity, of the last $k+1$ points that exited $AV_{\gamma,t}$. These points must have been in $AV_{\gamma,t(v')-1}$).

  Since $AV_{\gamma,t(q')-1} \subseteq AV_{\gamma,t} \cup AV_{\gamma,t(v')-1}$ then we have that $\forall v \in AV_{\gamma,t(q')-1}$: $dist(q', v) > 2\gamma$ at time $t(q')$. This implies that at time $t(q')$ the point q' should have been added to $AV_{\gamma,t(q')}$. Finally we can conclude that, since $t(q') \geq t(v')$, q' would still be in $AV_{\gamma,t}$ (an absurdity, it cannot happen).

- $\forall q \in W_t$ with $t(q) < t(v')$ we have that $dist(q, RV_\gamma \cup OV_\gamma) \leq 4\gamma$ if $|AV_\gamma| \leq k$. Assume, by contradiction, that there exists $q' \in W_t$ with $t(q') < t(v')$ such that $dist(q', RV_{\gamma,t} \cup OV_{\gamma,t}) > 4\gamma$. This implies that $dist(q, OV_{\gamma,t}) > 4\gamma$. Since $|AV_{\gamma,t}| \leq k$ we have that $\nexists v \in AV_{\gamma,t(v')}$ with $t(v') > t(v) > t-N$ otherwise it would be in $AV_{\gamma,t}$ and v' would not be the oldest point in AV_γ in the window W_t (no point is eliminated from $AV_{\gamma,t}$ in the window W_t since $|AV_{\gamma,t}| \leq k$). This implies that all the orphans in $OV_{\gamma,t}$ were representatives of points in $AV_{\gamma,t(q')-1}$. In turn it means that $dist(q', RV_{\gamma,t(q')-1}) > 4\gamma$ and $dist(q', AV_{\gamma,t(q')-1}) > 2\gamma$. The point q' should have been added to $AV_{\gamma,t(q')}$ at time $t(q')$ and, since $|AV_{\gamma,t}| \leq k$ would still be in $AV_{\gamma,t}$ at time t (an absurdity, it cannot happen).

■

An argument, with reasoning similar to the previous one for $RV_\gamma \cup OV_\gamma$, can also be made for $R_\gamma \cup O_\gamma$. The resulting statement is presented, without proof, in the following lemma:

Lemma 3.3 *Let $a' = \arg\min_{a \in A_\gamma} t(a)$ be the oldest point in A_γ, if it exists. For every $\gamma \in \Gamma$, the following properties hold after the end of each execution of procedure UPDATE(p), with respect to the window W_t at time step t:*

(1) $\forall q \in W_t$ with $t(q) \geq t(a')$ we have that $dist(q, R_\gamma \cup O_\gamma) \leq \delta\gamma$;

(2) $\forall q \in W_t$ with $t(q) < t(a')$ we have that if $|AV_\gamma| \leq k$ then $dist(q, R_\gamma \cup O_\gamma) \leq \delta\gamma$.

After introducing two sets of very important invariants, we present here an adapted version of [[2], Lemma 3], whose proof is a straightforward refitting of the one already present in the previous paper and which we have consequently decided to omit.

Lemma 3.4 *Let W_t be the window of points at time step t. Suppose that the coreset Q with proxy function $p : W_t \longrightarrow Q$ satisfies the following conditions, for a given $\gamma \in \Gamma$:*

(C1) For each $q \in W_t, dist(q, p(q)) \leq \delta\gamma$;

(C2) For each independent set $X \in I_{W_t}$ there exists an injective mapping $\pi_X : X \longrightarrow Q$ such that:

- *$\{\pi_X(i) : i \in X\} \subseteq Q$ is an independent set ($\in I_Q$);*
- *for each $i \in X, dist(i, \pi_X(i)) \leq \delta\gamma$*

Then:

(P1) There exists a solution of the Matroid Center problem on $M_Q = (Q, I_Q)$ of cost at most $OPT_{M_{W_t}} + 2\delta\gamma$;

(P2) Every solution F of the Matroid Center problem on M_Q of cost r_F is also a solution of cost at most $r_F + \delta\gamma$ on M_{W_t}.

Since the FKC problem is just a specialization of the Matroid Center problem, this lemma also works in our case. However, it is presented in the more general setting of the Matroid Center problem because it will be used also for the case with the transversal matroid in Chapter 4.

In order to facilitate our work in the next chapter and allow ourselves to create a more general theorem capable of guaranteeing the correctness and the approximation factor in both the partition matroid and the transversal matroid cases, let us now introduce two additional lemmas which will be proved separately for the two types of matroid but that can be used in the same fashion in a unique general theorem of correctness based on the work developed by [2].

Lemma 3.5 *Let p be the point that enters the window W_t at time step t. We have that p always enters $R_{\gamma,t}$ at time $t = t(p)$, for all $\gamma \in \Gamma$.*

Proof Consider a generic $\gamma \in \Gamma$. When p enters the window W_t ($t = t(p)$) we have $E_t = \{a \in A_{\gamma,t-1} : dist(p,a) \leq \frac{\delta\gamma}{2}\}$. Let us distinguish two cases depending on the size of E_t:

- If $E_t == \emptyset$, then the point p is added to $A_{\gamma,t}$ and its set of representatives is initialized to $repC_{\gamma,t}(p) = \{p\} \subseteq R_{\gamma,t}$.
- If $E_t \neq \emptyset$, the point p is at distance $dist(p,a) \leq \frac{\delta\gamma}{2}$ from all $a \in E_t$. Let us consider the attraction c-point $a_{add} = \arg\min_{a \in E_t} |repC^i_{\gamma,t-1}(a)|$ chosen by the procedure UPDATE(p) at time step t and its set of representatives $repC_{\gamma,t}(a_{add})$. Independently of what happens at lines 49-51 of procedure UPDATE(p) we know that p, being the latest point to enter the window, will never be the one eliminated. As a consequence we know that it will still be in $repC_{\gamma,t}(a_{add}) \subseteq RV_{\gamma,t}$ at the end of the procedure.

At this point, we can conclude that p is always added to $R_{\gamma,t}$ at time $t = t(p)$. ■

Lemma 3.6 *Let W_t be the window at time t. For each $\gamma \in \Gamma$, the following holds. Let $a \in S$ be a point that entered A_γ at some point in time, stated $a \in A_{\gamma,t(a)}$ (we denote a as a point in the stream S and not in the window W_t because we need to specify the behaviour of all sets $repC_{\gamma,t}(a)$, even those orphaned whose $a \notin W_t$). For $t \geq t(a)$ we have:*

- *If $a \in A_{\gamma,t}$, then $repC_{\gamma,t}(a)$ is a maximal independent set for $W_t(a) = \{x \in W_t : (dist(a,x) \leq \frac{\delta\gamma}{2}) \wedge (t(a) \leq t(x))\}$.*

- *If $a \notin A_{\gamma,t}$, then $repC_{\gamma,t}(a)$ is a maximal independent set for $W_t(a) = \{x \in W_t : (dist(a,x) \leq \frac{\delta\gamma}{2}) \wedge (t(a) \leq t(x) \leq \hat{t})\}$ where $\hat{t} \leq t$ is the time step a was removed from A_γ ($a \in A_{\gamma,\hat{t}-1}$ but $a \notin A_{\gamma,\hat{t}}$). Note that $repC_{\gamma,t}(a)$ is a subset of $repC_{\gamma,\hat{t}-1}(a)$ of points still not expired.*

Proof Consider a generic $\gamma \in \Gamma$. This lemma will be proved by induction on the time step t.

- Base case, $t = t(a)$.
  By the design of procedure UPDATE(a), we known that $a \in A_{\gamma,t}$ and that $repC_{\gamma,t}(a) = \{a\}$. Furthermore, given that $a \in A_{\gamma,t}$ is the first point $x \in W_t$ to enter at time $t \geq t(a)$ and at distance $dist(a,x) \leq \frac{\delta\gamma}{2}$ from a, we have $W_t(a) = \{a\}$. As a result of these observations, we can conclude that $repC_{\gamma,t}(a)$, being equal to $W_t(a)$ and containing only one point, is a maximal independent set for it.

- Inductive case 1, $t > t(a)$ and $a \in A_{\gamma,t}$.
  By inductive hypothesis, we assume $repC_{\gamma,t-1}(a)$ to be a maximal independent set for $W_{t-1}(a)$. Our objective is to prove that, at the end of time step t, $repC_{\gamma,t}(a)$ remains a maximal independent set for $W_t(a)$.
  Let u be the point that expired from W_t and p the point that entered W_t at time step t. If $a \in A_{\gamma,t}$, then we must have that $u \notin repC_{\gamma,t-1}(a)$ and that $u \notin W_t(a)$ otherwise a would have expired at time t and not be in $A_{\gamma,t}$ ($u \in W_t(a)$ implies $t(u) \geq t(a)$).
  We can distinguish 2 cases:
  - If $p \notin W_t(a)$, then we have that $W_t(a) = W_{t-1}(a)$ and that $repC_{\gamma,t}(a) = repC_{\gamma,t-1}(a)$. Clearly, given that at time step t nothing changes for the two sets connected to a, we can conclude that $repC_{\gamma,t}(a)$ is still a maximal independent set for $W_t(a)$.
  - If $p \in W_t(a)$, then we have that $W_t(a) = W_{t-1}(a) \cup \{p\}$ and that $repC_{\gamma,t}(a) \subseteq repC_{\gamma,t-1}(a) \cup \{p\}$. As we can see from lines 49-51 of procedure UPDATE(p), we remove a point $o \in repC_{\gamma,t-1}(a)$ only if we have k_i points of its category newer than him. Since we can never have more than k_i points in the same category $i = 1, \ldots, \ell$ inside an independent set, it follows that $repC_{\gamma,t}(a)$ remains maximal for $W_t(a)$.

- Inductive case 2, $t > t(a)$ and $a \notin A_{\gamma,t}$.
  By inductive hypothesis, we assume $repC_{\gamma,t-1}(a)$ to be a maximal independent set for $W_{t-1}(a)$. Our objective is to prove that, at the end of time step t, $repC_{\gamma,t}(a)$ remains a maximal independent set for $W_t(a)$.
  Let u be the point that expired from W_t and p the point that entered W_t at time step t. Since $a \notin A_\gamma$ $(t = t(p) \geq \hat{t})$, we are sure that $p \notin W_t(a)$, therefore $W_t(a) = W_{t-1}(a) \setminus \{u\}$.
  We can have 2 different situations:
  - If $u \notin repC_{\gamma,t-1}(a)$, then $repC_{\gamma,t}(a) = repC_{\gamma,t-1}(a)$. In turn this means that the maximal independent set $repC_{\gamma,t-1}(a)$ for $W_{t-1}(a)$ still exists in $W_t(a)$ $(rank(W_t(a)) = rank(W_{t-1}(a)) = \left|repC_{\gamma,t}(a)\right|)$. As a result, we can say that $repC_{\gamma,t}(a)$ remains a maximal independent set for $W_t(a)$.
  - If $u \in repC_{\gamma,t-1}(a)$, then $repC_{\gamma,t}(a) = repC_{\gamma,t-1}(a) \setminus \{u\}$. In this case, given that we always kept the newest points for each category $i = 1, ..., \ell$ (at most k_i of them), it means that $repC_{\gamma,t}(a) \cap W_i = W_t(a) \cap W_i$. Since the number of points in $W_t(a)$ for category i is smaller than $\left|repC_{\gamma,t-1}(a) \cap W_i\right|$, it logically follows that $rank(W_t(a)) = rank(W_{t-1}(a)) - 1$. Therefore, having $|repC_{\gamma,t}(a)| = |repC_{\gamma,t-1}(a)| - 1$, we can conclude that the set $repC_{\gamma,t}(a)$ continues to be a maximal independent set for $W_t(a)$.

In summary, we can state that $repC_{\gamma,t}(a)$ is a maximal independent set for $W_t(a)$ for every $t \geq t(a)$. ■

At this point, we have all the necessary concepts needed in order to prove the core results for the paper where we show the algorithm is correct and returns a $(3 + \epsilon)$-approximation for the FKC problem. As done for Lemma 3.4, the following results are presented in the more general setting of the Matroid Center problem because they will be used also for the case with the transversal matroid in Chapter 4.

Theorem 3.7 *Let $\epsilon \in (0, 1)$ and $\beta > 0$. Suppose that an α-approximation algorithm $\mathcal{A}$ for the Matroid Center problem is available. When procedure QUERY() is run at time step t, the selected coreset $T = R_\gamma \cup O_\gamma$ exhibits properties (P1) and (P2) of Lemma*

3.4 with $\delta = \frac{\epsilon'}{(1+\beta)}$ and $\epsilon' = \frac{\epsilon}{(1+2\alpha)}$. Hence, the solution F returned by running $\mathcal{A}$ on instance M_T of the Matroid Center problem is an $(\alpha + \epsilon)$-approximate solution to instance M_{W_t} of the Matroid Center problem.

Proof In order to prove this theorem we will first prove that the coreset $T = R_\gamma \cup O_\gamma$ chosen by procedure QUERY() at time step t is of the type required by Lemma 3.4. Then, we will use the properties of that lemma to bound the solution returned by the algorithm and insure that it is actually an $(\alpha + \epsilon)$-approximation to instance M_{W_t} of the Matroid Center problem.

We start by proving that coreset $T = R_\gamma \cup O_\gamma$ satisfies condition (C1) of Lemma 3.4. By design of procedure QUERY() at time step t, the coreset T is returned only if $|AV_\gamma| \leq k$ and $|C| \leq k$. Knowing that $|AV_\gamma| \leq k$ and taking advantage of both point (1) and point (2) of Lemma 3.3, we can say that

$$\forall q \in W_t : dist(q, R_\gamma \cup O_\gamma) \leq \delta\gamma \tag{3.1}$$

and conclude that condition (C1) is satisfied by T.

To prove condition (C2) of Lemma 3.4 for $T = R_\gamma \cup O_\gamma$, consider an arbitrary $X \in I_{W_t}$. We must show that there exists an injective mapping $\pi_X : X \longrightarrow T$ such that:

- $\{\pi_X(o) \mid o \in X\} \in I_T$ (the set of mapped points form an independent set of I_{W_t} and are a subset of T);
- $\forall o \in X : dist(o, \pi_X(o)) \leq \delta\gamma$ (equivalent to say that o and $\pi_X(o)$ are at distance $\leq \frac{\delta\gamma}{2}$ to at least one attraction c-point in common).

Let $X = \{x_u \mid 1 \leq u \leq |X|\}$. We will define the mapping π_X incrementally one element at the time. Suppose that we have fixed the mapping for the first $h \geq 0$ elements of X and assume, inductively, that $Y(h) = \{\pi_X(x_u) \mid 1 \leq u \leq h\} \cup \{x_u \mid h < u \leq |X|\}$ is an independent set ($\in I_{W_t}$) of size $|Y(h)| = |X|$. For $1 \leq u \leq h$ we have that $dist(x_u, \pi_X(x_u)) \leq \delta\gamma$.

Consider now x_{h+1}, by design of the procedure UPDATE(p) and Lemma 3.5, when the point x_{h+1} entered the window at some time $t(x_{h+1})$, it was certainly added to $R_{\gamma,t(x_{h+1})}$ (it enters at least one $repC_{\gamma,t(x_{h+1})}(a')$ for a certain point $a' \in A_{\gamma,t(x_{h+1})}$).

At this point, let us distinguish between two cases:

- If $x_{h+1} \in R_{\gamma,t} \cup O_{\gamma,t}$, then we can simply set $\pi_X(x_{h+1}) = x_{h+1}$. Hence, $Y(h+1) = Y(h)$ and $dist(x_{h+1}, \pi_X(x_{h+1})) = 0 \leq \delta\gamma$.

- If $x_{h+1} \notin R_{\gamma,t} \cup O_{\gamma,t}$, then we still have that, at time $t(x_{h+1})$, the point entered a certain $repC_{\gamma,t(x_{h+1})}(a')$ for some $a' \in A_{\gamma,t(x_{h+1})}$ with $dist(a', x_{h+1}) \leq \frac{\delta\gamma}{2}$ ($a' \in E_{t(x_{h+1})}$). Using the result that $repC_{\gamma,t}(a')$ is a maximal independent set for $W_t(a')$ of Lemma 3.6 combined with the extended augmentation property (Lemma 2.2), we can find $\pi_X(x_{h+1})$. In particular, let us set: $A = Y(h) \setminus \{x_{h+1}\}$; $y = x_{h+1}$; $W' = W_t(a')$; $B = repC_{\gamma,t}(a')$. Since $\exists x_{h+1} \in W' \setminus A$ such that $A \cup \{x_{h+1}\} \in I_{W'}$ and B is maximal for W', then $\exists \pi_X(x_{h+1}) \in B \setminus A$ such that $A \cup \{\pi_X(x_{h+1})\} \in I_{W'}$. Hence, $Y(h+1) = (Y(h) \setminus \{x_{h+1}\}) \cup \{\pi_X(x_{h+1})\}$ and, given that $dist(x_{h+1}, a') \leq \frac{\delta\gamma}{2}$, we also have that $dist(x_{h+1}, \pi_X(x_{h+1})) \leq dist(x_{h+1}, a') + dist(a', \pi_X(x_{h+1})) \leq \frac{\delta\gamma}{2} + \frac{\delta\gamma}{2} \leq \delta\gamma$.

After $|X|$ iterations of the above inductive argument, we have that the mapping π_X is completely specified and exhibits the following properties:

- π_X is injective because each x_u has been substituted by another specific point in $R_\gamma \cup O_\gamma$ (using the extended augmentation property);
- $\{\pi_X(x_u) \mid 1 \leq u \leq |X|\} \subseteq T = R_\gamma \cup O_\gamma$ is an independent set ($\in I_T$) since it is constructed to still be an independent set at each $h \geq 0$;
- for $1 \leq u \leq |X|$ we have $dist(x_u, \pi_X(x_u)) \leq \delta\gamma$ since at each $h \geq 0$ the two points x_{h+1} and $\pi_X(x_{h+1})$ either represent the same point or are at distance $\leq \frac{\delta\gamma}{2}$ from the same attraction c-point.

This concludes the proof of condition (C2) for coreset T.

Now, given that the coreset $T = R_\gamma \cup O_\gamma$ chosen by procedure QUERY() at time step t satisfies conditions (C1) and (C2) of Lemma 3.4, we can imply that it also exhibits properties (P1) and (P2) of that lemma.

By property (P1) of Lemma 3.4, we can say that the optimal solution to M_T has cost at most $OPT_{M_{W_t}} + 2\delta\hat{\gamma}$ where $\hat{\gamma}$ is the guess chosen by the procedure QUERY(). In the worst case the procedure returns $T = R_\gamma \cup O_\gamma$ for a guess $\hat{\gamma}$ such that $OPT_{k,W_t} < \hat{\gamma} \leq OPT_{k,W_t}(1+\beta) \leq OPT_{M_{W_t}}(1+\beta)$. This is true because the algorithm stops when it finds not more than k points at distance greater than 2γ for a certain guess γ and this is certain to happen when the guess is greater than the optimal radius OPT_{k,W_t}. In fact, we cannot find $k+1$ points at distance greater than $2OPT_{k,W_t}$ from one another, otherwise we would have that in the optimal clustering at least two of these points would be in the same cluster, which is impossible given the triangular inequality. Moreover, by the design of the geometric progression Γ, the first guess $\hat{\gamma}$ greater than OPT_{k,W_t} is always at most $OPT_{k,W_t}(1+\beta)$ and, since a solution for the Matroid Center problem on M_{W_t} is also a solution for the k-Center problem on W_t, we have that $\hat{\gamma} \leq OPT_{k,W_t}(1+\beta) \leq OPT_{M_{W_t}}(1+\beta)$.

This implies that in the worst case the cost of the optimal solution for the Matroid Center problem on M_T is at most $OPT_{M_{W_t}} + 2\delta\hat{\gamma} \leq (1+2\epsilon')OPT_{M_{W_t}}$. Hence, the solution F computed using algorithm $\mathcal{A}$ has always cost $r_F \leq \alpha(1+2\epsilon')OPT_{M_{W_t}} = (\alpha + \epsilon\frac{2\alpha}{1+2\alpha})OPT_{M_{W_t}}$. By property (P2) of Lemma 3.4 the solution F to M_T is also a solution to M_{W_t} for the Matroid Center problem with cost at most $r_F + \delta\hat{\gamma} \leq r_F + \frac{\epsilon}{1+2\alpha}OPT_{M_{W_t}} \leq (\alpha+\epsilon)OPT_{M_{W_t}}$ which concludes the proof. ■

Corollary 3.8 *procedure QUERY() can be used to compute a* $(3+\epsilon)$*-approximate solution to any instance* M_W*, for any fixed* $\epsilon \in (0,1)$ *and* $\beta > 0$.

Proof Using the sequential 3-approximation algorithm for the Matroid Center problem by [3] and using Theorem 3.7 we obtain a $(3+\epsilon)$-approximate solution to M_W for a fixed $\epsilon \in (0,1)$ and $\beta > 0$. ■

Now that the correctness and the approximation ratio for the algorithm have been formally proven, we move to the study of its time and space complexity. In order to do so, we will use two separate theorems.

Theorem 3.9 *At any time t during the processing of the stream S, the sets stored in the working memory (i.e* AV_γ*,* RV_γ*,* OV_γ*,* A_γ*,* R_γ *and* O_γ *for every guess* γ*) contain*

$$O\left(k^2 \frac{\log(\Delta)}{\log(1+\beta)}\left(\frac{32}{\delta}\right)^{D_{W_t}}\right) \tag{3.2}$$

points, overall, where D_{W_t} *is the doubling dimension of the current window* W_t *and* Δ *is the aspect ratio of the stream S.*

Proof Consider an arbitrary time t. Since $|\Gamma| = O\left(\frac{\log(\Delta)}{\log(1+\beta)}\right)$, it is sufficient to show that for every $\gamma \in \Gamma$ the aggregate size of the sets of validation points is $O(k)$, instead for the sets of coreset points is $O\left(k^2\left(\frac{32}{\delta}\right)^{D_{W_t}}\right)$.

We first show that $|AV_\gamma| = |RV_\gamma| \leq k+1$. By construction, at each time step t the procedure UPDATE(p) keeps the size of AV_γ under control by eliminating the oldest point in the set if it contains more than $k+1$ points. Moreover, we know that RV_γ contains exactly one representative $repV_\gamma(v)$ for each attraction v-point $v \in AV_\gamma$, which limits the number of points it can contain to the cardinality of AV_γ (at most $k+1$). Indeed, when a point is removed from AV_γ, its representative is moved to OV_γ. After this we can conclude that $|RV_\gamma| = |AV_\gamma| \leq k+1$.

For what concerns the bound on OV_γ, let us consider the sets $OV_{\gamma,t}$ and $AV_{\gamma,t}$ at time step t and let us distinguishing three cases given the size of the attraction set:

- If $|AV_{\gamma,t}| = 0$, then it is certain that $W_t = \emptyset$ since the first point in the window is always added to $AV_{\gamma,t}$. In this case $|OV_{\gamma,t}| = 0 \leq k+1$.
- If $|AV_{\gamma,t}| \geq 1$, consider $v' = \arg\min_{v \in AV_{\gamma,t}} t(v)$ the oldest point in $AV_{\gamma,t}$ in the window W_t at time t.
  - If $|AV_{\gamma,t}| \leq k$, then $\nexists v \in AV_{\gamma,t(v')-1}$ also in W_t, otherwise it would have been added to $AV_{\gamma,t}$ and v' would not be the oldest point in $AV_{\gamma,t}$ in the window W_t. In this case all points in $OV_{\gamma,t}$ must have been representative for points not in W_t anymore and, since we always have at each time step at most $k+1$ representatives, we must have that $|OV_{\gamma,t}| \leq k+1$.
  - If $|AV_{\gamma,t}| = k+1$, then $\nexists q \in OV_{\gamma,t}$ such that $t(q) < t(v')$ (all points older than v' are eliminated during the procedure UPDATE(p)). In this case, since $|AV_{\gamma,t(v')}| \leq k+1$ and $v' \in AV_{\gamma,t(v')}$, we know that there could have been at most k representatives newer than v' (with $t(\cdot) > t(v')$) for the points in $AV_{\gamma,t(v')} \setminus \{v'\}$. The number of $OV_{\gamma,t}$ then must be limited to k.

Thus, we can conclude that $|OV_\gamma| \leq k+1$.

Next, we show that $|A_\gamma| \leq 2(k+1)\left(\frac{32}{\delta}\right)^{D_{W_t}}$ and $|R_\gamma| = |O_\gamma| \leq 2k(k+1)\left(\frac{32}{\delta}\right)^{D_{W_t}}$, where D_{W_t} is the doubling dimension of the current window W_t. From the proof above we know that there are at most $k+1$ points in each of the sets AV_γ, RV_γ and OV_γ. By construction, we also know that the distance between any two points in A_γ is greater or equal to $\frac{\delta\gamma}{2}$. Given point (1) and point (2) of Lemma 3.2, we have that $\forall a \in A_\gamma$: $dist(a, RV_\gamma \cup OV_\gamma) \leq 4\gamma$, which means that the attraction c-points in A_γ can be all enclosed in at most $2(k+1)$ balls of radius 4γ. By Lemma 2.2, in each of these $2(k+1)$ balls, there can be at most $\left(\frac{32}{\delta}\right)^{D_{W_t}}$ points of A_γ, which implies that $|A_\gamma| \leq 2(k+1)\left(\frac{32}{\delta}\right)^{D_{W_t}}$.

Next, we show that $|R_\gamma| \leq 2k(k+1)\left(\frac{32}{\delta}\right)^{D_{W_t}}$. This bound on R_γ follows from the fact that R_γ contains one set of representatives $repC_\gamma(a)$ for each $a \in A_\gamma$ and from the fact that, given the partition matroid constraints, the maximal rank of an independent set of W_t is $\sum_{i=1}^{m} k_i = k$.

We are left to show that $|O_\gamma| \leq 2k(k+1)(\frac{32}{\delta})^{D_{W_t}}$. To do so we can use an argument completely analogous to the one for $|OV_\gamma|$, with the only exception that now we are now dealing with sets of representative of size at most k instead of single points. In this spirit we decided to omit the actual step by step reasoning that can be easily derived from the previous one for OV_γ and make the dissertation shorter.

■

Theorem 3.10 *Procedure UPDATE(p) runs in time*

$$O\left(k^2 \frac{\log(\Delta)}{\log(1+\beta)} \left(\frac{32}{\delta}\right)^{D_{W_t}}\right) \tag{3.3}$$

while procedure QUERY() runs in time

$$O\left(k^2 \frac{\log(\Delta)}{\log(1+\beta)} + k^2 \left(\frac{32}{\delta}\right)^{D_{W_t}} + T_{\mathcal{A}}(R_\gamma \cup O_\gamma)\right) \tag{3.4}$$

Proof The complexity of UPDATE(p) is dominated by the computation at lines 36-38 where, for each guess $\gamma \in \Gamma$, we remove all the orphans in O_γ older than the oldest attraction v-point (in the worst case we could empty the entire O_γ). The claimed bound follows from Theorem 3.9.

For what concerns QUERY(), we observe that the complexity is dominated by the size of $T = R_\gamma \cup O_\gamma$, the time required by $\mathcal{A}(R_\gamma \cup O_\gamma)$ to run and the calculation of C where, for each point $q \in AV_\gamma \cup OV_\gamma \cup RV_\gamma$ we perform the calculation $dist(q, C)$, which can be accomplished in time $O(k^2)$ (for at most $O(k)$ points we compute $dist(\cdot,\cdot)$ in time $O(k)$) for a given guess $\gamma \in \Gamma$. ■

4 Adaptation to the transversal matroid constraint

After finishing the last chapter, where we laid out the main concepts and techniques used to deal with Fair k-Center clustering under sliding window constraints, we now move to the more general k-Center problem under transversal matroid constraints, where we will slightly change our approach but retain the same structure of the analysis. In order to investigate more broadly the problem, we decided to divide the presentation in two parts, one for each proposed solution. The two ideas can be summarized as follows:

- The first one, under acceptable assumptions, keeps, for each $a \in A_\gamma$, a sufficient number of points of each category in its vicinity, avoiding the actual computation of the maximal independent sets (maximum matchings in this case) but relying on the fact that they always exist within the set of points kept;
- The second, under different reasonable assumptions, tries instead to incrementally keep in memory a maximal independent set for each attraction c-point. While we are not able yet to formally proof the correctness of this second solution, we conjecture it and show a number of empirical examples supporting our conjecture.

In the next two sections we will present only the changes that must be applied in the two cases in order to make the procedure UPDATE(p) work with the more general transversal matroid, instead of the partition matroid. Furthermore, we will see how generalizing Lemma 3.5 and Lemma 3.6 is sufficient to make the updated procedures correct and maintain the same approximation factor without formally re-proving the central results given by Theorem 3.7 and Corollary 3.8. Finally, we will provide updated time and space complexity bounds similar to the ones in Theorems 3.9 and 3.10. This choice was made so to have a more compact presentation of the generalized solutions, avoid repetitions and keep the dissertation shorter.

4.1 First proposed solution

As briefly described before, our first idea on how to solve the k-Center problem under transversal matroid constraints is based on keeping a certain number of points for each attraction c-point and each category in the window W_t. In order to set ourselves in a situation similar to the previous chapter, let us reasonably assume that for the window W_t and more generally for the stream S the number of (possibly non-disjoint) categories $|A|$ in the transversal matroid $M_{W_t} = (W_t, I_{W_t})$ is at most k. This implies that the set of edges E in the connected bipartite graph $(W_t, A; E)$ has size $|E| \leq k|W_t|$. By remembering the technique used in the previous chapter to keep a maximal independent set for each attraction c-point and by remembering the assumption we made on the rank of the transversal matroid ($rank(M_{W_t}) = k$), we understood that keeping k points for each category and each $a \in A_\gamma$ is sufficient to find a good approximate solution for the entire window. In particular, we saw that, in order to guarantee that a maximal independent set is always contained in the set of points we keep over time, we need to store the *latest* k points that entered the window in each case. We will structure $repC_\gamma(a)$ as a hash map with a bucket for each category $A_i \in A$, which in turn will contain duplicates of points associated to that specific category.

With this concepts in mind, we present now the generalized version of procedure UPDATE(p):

```
Algorithm 3 UPDATE($p$)
  for each $\gamma \in \Gamma$ do
      Omitting pseudocode ...
      Consider the following lines as numbered from 42 to 55
```

```
if $E == 0$ then
    $A_\gamma = A_\gamma \cup \{p\}$
    for each $A_i \in A$ do
        if $p \in W_t \cap A_i$ then
            Add $p$ to the bucket of $repC_\gamma(a)$ associated to $A_i$ in $R_\gamma$
    $R_\gamma = R_\gamma \cup repC_\gamma(p)$
else
    for each $a \in E$ do
        for each $A_i \in A$ do
            if $p \in W_t \cap A_i$ then
                Add $p$ to the bucket of $repC_\gamma(a)$ associated to $A_i$ in $R_\gamma$
                if $repC_\gamma(a) \cap A_i > k$ then
                    $o_{rem} = \arg\min_{o \in repC_\gamma(a) \cap A_i} TTL(o)$
                    Remove $o_{rem}$ from the bucket of $repC_\gamma(a)$ associated
                    with $A_i$ in $R_\gamma$
```

At this point, in order to argue for the correctness and the approximation ratio of the proposed solution, we only need to present the updated versions of Lemma 3.5 and Lemma 3.6. Once those two results are formally proven, we will be obtain what we want by re-using Theorem 3.7 and Corollary 3.8 with the updated versions of the two lemmas.

Lemma 4.1 *Let p be the point that enters the window W_t at time step t. We have that p always enters $R_{\gamma,t}$ at time $t = t(p)$, for all $\gamma \in \Gamma$.*

Proof Consider a generic $\gamma \in \Gamma$. As for the partition matroid case, when p enters the window W_t ($t = t(p)$) we have $E_t = \{a \in A_{\gamma,t-1} : dist(p,a) \leq \frac{\delta\gamma}{2}\}$. Let's distinguish two cases depending on the size of E_t:

- If $E_t == \emptyset$, then the point p is added to $A_{\gamma,t}$ and its set of representatives $repC_{\gamma,t}(p)$ contains p in each of the buckets associated with one of its categories. Knowing there must be at least one category for each point, we can trivially conclude that $p \in R_{\gamma,t}$.
- If $E_t \neq \emptyset$, the point p is at distance $dist(p,a) \leq \frac{\delta\gamma}{2}$ from all $a \in E_t$. This means that there exists at least one attraction c-point $a \in A_{\gamma,t}$ at distance not more than $\frac{\delta\gamma}{2}$. Let's consider this generic $a \in A_{\gamma,t}$ (it will be true for

all $a \in E_t$) and its set of representatives $repC_{\gamma,t}(a)$. Independently of what happens at lines 53-55 of procedure UPDATE(p) we know that p, being the latest point to enter the window, will never be the one eliminated. As a consequence we know that it will still be in $repC_{\gamma,t}(a) \subseteq RV_{\gamma,t}$ at the end of the procedure.

At this point, we can conclude that p is always added to $R_{\gamma,t}$ at time $t = t(p)$. ■

Lemma 4.2 *Let W_t be the window at time t. For each $\gamma \in \Gamma$, the following holds. Let $a \in S$ be a point that entered A_γ at some point in time, namely, $a \in A_{\gamma,t(a)}$ (we denote a as a point in the stream S and not in the window W_t because we need to specify the behaviour of all sets $repC_{\gamma,t}(a)$, even those orphaned whose $a \notin W_t$). For $t \geq t(a)$ we have:*

- *If $a \in A_{\gamma,t}$, then $repC_{\gamma,t}(a)$ contains a maximal independent set for $W_t(a) = \{x \in W_t : (dist(a,x) \leq \frac{\delta\gamma}{2}) \wedge (t(a) \leq t(x))\}$.*
- *If $a \notin A_{\gamma,t}$, then $repC_{\gamma,t}(a)$ contains a maximal independent set for $W_t(a) = \{x \in W_t : (dist(a,x) \leq \frac{\delta\gamma}{2}) \wedge (t(a) \leq t(x) \leq \hat{t})\}$ where $\hat{t} \leq t$ is the time step a was removed from A_γ ($a \in A_{\gamma,\hat{t}-1}$ but $a \notin A_{\gamma,\hat{t}}$). Note that $repC_{\gamma,t}(a)$ is a subset of $repC_{\gamma,\hat{t}-1}(a)$ of points still not expired.*

Proof Consider a generic $\gamma \in \Gamma$. This lemma will be proved by induction on the time step t.

- Base case, $t = t(a)$.
  By the design of procedure UPDATE(a), we known that $a \in A_{\gamma,t}$ and that $repC_{\gamma,t}(a)$, if considered as a set, contains only a. Furthermore, we know that $W_t(a) = \{a\}$. If follows that $repC_{\gamma,t}(a)$, being equal to $W_t(a)$ and containing only one point, contains a maximal independent set for it or, equivalently, a maximum matching between the points in $W_t(a)$ and A in the connected bipartite graph (in practice a single edge between a and one of its categories).
- Inductive case 1, $t > t(a)$ and $a \in A_{\gamma,t}$.
  By inductive hypothesis, we assume $repC_{\gamma,t-1}(a)$ contains, when considered as a set, a maximal independent set for $W_{t-1}(a)$. Our objective is

to prove that, at the end of time step t, $repC_{\gamma,t}(a)$ contains a maximal independent set for $W_t(a)$.
Let u be the point that expired from W_t and p the point that entered W_t at time step t. If $a \in A_{\gamma,t}$, then we must have that $u \notin repC_{\gamma,t-1}(a)$ and that $u \notin W_t(a)$ otherwise a would have expired at time t and not be in $A_{\gamma,t}$ ($u \in W_t(a)$ implies $t(u) \geq t(a)$).
We can distinguish 2 cases:

– If $p \notin W_t(a)$, then we have that $W_t(a) = W_{t-1}(a)$ and that $repC_{\gamma,t}(a) = repC_{\gamma,t-1}(a)$. Clearly, given that at time step t nothing changes for the two sets connected to a, we can conclude that $repC_{\gamma,t}(a)$ still contains a maximal independent set for $W_t(a)$.

– If $p \in W_t(a)$, then we have that $W_t(a) = W_{t-1}(a) \cup \{p\}$ and that $repC_{\gamma,t}(a)$, if considered as a set, is a subset of $repC_{\gamma,t-1}(a) \cup \{p\}$. As we can see from lines 53-55 of procedure UPDATE(p), we remove a duplicate of point $o \in repC_{\gamma,t-1}(a)$ for one of its categories (in one bucket) only if we have k newer points in $repC_{\gamma,t-1}(a) \cup \{p\}$ for that specific category. Since we can never have more than k edges in a maximum matching between the points in $W_t(a)$ and A (we assumed $rank(W_t) = k$ for all time step t), it follows that storing at most k edges connected to each category $A_i \in A$ and each attraction c-point is sufficient to always find the maximum matching possible in $W_t(a)$. We can conclude that $repC_{\gamma,t}(a)$ contains a maximal independent set for $W_t(a)$.

- Inductive case 2, $t > t(a)$ and $a \notin A_{\gamma,t}$.
By inductive hypothesis, we assume $repC_{\gamma,t-1}(a)$ contains, when considered as a set, a maximal independent set for $W_{t-1}(a)$. Our objective is to prove that, at the end of time step t, $repC_{\gamma,t}(a)$ contains a maximal independent set for $W_t(a)$.
Let u be the point that expired from W_t and p the point that entered W_t at time step t. Since $a \notin A_\gamma$ ($t = t(p) \geq \hat{t}$), we are sure that $p \notin W_t(a)$, therefore $W_t(a) = W_{t-1}(a) \setminus \{u\}$.
We can have 2 different situations:

– If $u \notin repC_{\gamma,t-1}(a)$, then $repC_{\gamma,t}(a) = repC_{\gamma,t-1}(a)$. In turn this means that the maximal independent set contained in $repC_{\gamma,t-1}(a)$ continues to exist in $W_t(a)$ ($rank(W_t(a)) = rank(W_{t-1}(a)) = \left|repC_{\gamma,t}(a)\right|$). As a result, we can say that $repC_{\gamma,t}(a)$ still contains a maximal independent set for $W_t(a)$.

– If $u \in repC_{\gamma,t-1}(a)$, then, when considered as a set, we have that $repC_{\gamma,t}(a) = repC_{\gamma,t-1}(a) \setminus \{u\}$. In this case, given that we always kept the newest points (at most k of them) for each category $i = 1, ..., \ell$ and each attraction c-point, it means that all the points in $W_t(a)$ with categories associated to u that entered after $t(u)$ are stored in the buckets of $repC_{\gamma,t}(a)$ reserved for those categories. Therefore, we can conclude that $repC_{\gamma,t}(a)$ continues to contain a maximal independent set for $W_t(a)$.

At this point, we can state that $repC_{\gamma,t}(a)$ contains a maximal independent set for $W_t(a)$ for every $t \geq t(a)$. ■

One note worth mentioning about Lemma 4.2 is that, differently to what happens in Lemma 3.6, we only proved that $repC_{\gamma}(a)$ *contains* a maximal independent set for $W_t(a)$ instead of actually *being* a maximal independent set. Given this difference, in Theorem 3.7 we need to change the set B used by the extended augmentation property from $B = repC_{\gamma,t}(a')$, to B equal to the maximal independent set contained in $repC_{\gamma,t}(a')$. Apart from this detail, Theorem 3.7 and Corollary 3.8 can be re-used to prove the correctness and the approximation factor of the updated procedures.

After presenting the two updated lemmas, let us now move to the study of the time and space complexity for the two procedures.

Theorem 4.3 *At any time t during the processing of the stream S, the sets stored in the working memory (i.e AV_γ, RV_γ, OV_γ, A_γ, R_γ and O_γ for every guess γ) contain*

$$O\left(k^3 \frac{\log(\Delta)}{\log(1+\beta)} \left(\frac{32}{\delta}\right)^{D_{W_t}}\right) \tag{4.1}$$

points, overall, where D_{W_t} is the doubling dimension of the current window W_t and Δ is the aspect ratio of the stream S.

Proof The bounds on the size of the support structures remain the same for the validation sets and for A_γ. The only difference can be found in the bounds on R_γ and O_γ, because instead of keeping a set of maximum size $O(k)$ for each attraction c-point in A_γ, we keep an hash map of size at most $O(k^2)$ (the maximum number of duplicated points). Given this reason, we can state that, at any time t, the overall number of points stored in the working memory is $O\left(k^3 \frac{\log(\Delta)}{\log(1+\beta)} \left(\frac{32}{\delta}\right)^{D_{W_t}}\right)$. ■

Theorem 4.4 *Procedure UPDATE(p) runs in time*

$$O\left(k^3 \frac{\log(\Delta)}{\log(1+\beta)} \left(\frac{32}{\delta}\right)^{D_{W_t}}\right) \tag{4.2}$$

while procedure QUERY() runs in time

$$O\left(k^2 \frac{\log(\Delta)}{\log(1+\beta)} + k^3 \left(\frac{32}{\delta}\right)^{D_{W_t}} + T_{\mathcal{A}}(R_\gamma \cup O_\gamma)\right). \tag{4.3}$$

We avoid the proof of Theorem 4.4 since the complexity of UPDATE(p) and QUERY() only change because of updated bound on the working memory contained in Theorem 4.3.

4.2 Second proposed solution

After completing the first proposed solution, we move to our second idea on how to approximate efficiently the k-Center problem under transversal matroid constraints. Unfortunately, unlike the results in the previous chapter and in the previous section, which were rigorous in all their facets, we were not able to prove the formal correctness of this additional generalized version of the algorithm. In particular, we were not able to construct the proof of the updated Lemma 3.6 for this case. However, we still decided to present it given its attractive space and time requirements and given some empirical evidence which justifies a conjecture of correctness.

As we did in the previous section, we will present only the changes that must be applied to the procedure UPDATE(p) and the space and time complexities of the procedures as we did in Theorems 3.9 and 3.10.

In order to set ourselves in the same situation as in the previous cases, let us reasonably assume that each point in the window W_t and more generally in the stream S can be assigned to at most a constant number of categories. This implies that the set of (possibly non-disjoint) categories A for the window W_t in the transversal matroid $M_{W_t} = (W_t, I_{W_t})$ and the connected bipartite graph $(W_t, A; E)$ can be very large, but the set of edges E has size $|E| = O(|W_t|)$ given that each point in W_t, when considered as a vertex, has constant degree. Let us also remember that we assumed for the transversal matroid $M_{W_t} = (W_t, I_{W_t})$, as for the case of partition matroid, to have rank $rank(M_{W_t}) = k$. These properties will be useful to maintain the bound on the size of the support structures and to limit the time and space complexity of the two procedures for the algorithm.

With these ideas in mind, we first present the generalized version of procedure UPDATE(p):

Algorithm 4 UPDATE(p)

```
for each $\gamma \in \Gamma$ do
    Omitting pseudocode ...
    $E = \{a \in A_\gamma \mid dist(p, a) \leq \frac{\delta\gamma}{2}\}$
    Omitting pseudocode ...
    if $E == 0$ then
        Omitting pseudocode ...
    else
        Consider the following lines as numbered from 47 to 54 ...
        for each $a \in E$ do
            if $repC_\gamma(a) \cup \{p\} \in I_{W_t}$ then
                Set $repC_\gamma(a) = repC_\gamma(a) \cup \{p\}$ in $R_\gamma$
            else
                for each $r \in repC_\gamma(a)$ with increasing values of $TTL(r)$ do
                    if $(repC_\gamma(a) \setminus \{r\}) \cup \{p\} \in I_{W_t}$ then
                        Set $repC_\gamma(a) = (repC_\gamma(a) \setminus \{r\}) \cup \{p\}$ in $R_\gamma$
                        break
```

Before moving to the study of the empirical correctness for the solution, we are going to present the updated versions of Lemmas 3.5 and 3.6, giving the proof for the former but lacking one for the latter.

Lemma 4.5 *Let p be the point that enters the window W_t at time step t. We have that p always enters $R_{\gamma,t}$ at time $t = t(p)$, for all $\gamma \in \Gamma$.*

Proof Consider a generic $\gamma \in \Gamma$. When p enters the window W_t ($t = t(p)$) we have $E_t = \{a \in A_{\gamma,t-1} : dist(p,a) \leq \frac{\delta\gamma}{2}\}$. Let us distinguish two cases depending on the size of E_t:

- If $E_t == \emptyset$, then the point p is added to $A_{\gamma,t}$ and its set of representatives is initialized to $repC_{\gamma,t}(p) = \{p\} \subseteq R_{\gamma,t}$.
- If $E_t \neq \emptyset$, the point p is at distance $dist(p,a) \leq \frac{\delta\gamma}{2}$ from all $a \in E_t$. Let us consider this generic $a \in A_{\gamma,t}$ (it will be true for all $a \in E_t$) and its set of representatives $repC_{\gamma,t-1}(a)$ at time $t-1$.
  - If $repC_{\gamma,t-1}(a) \cup \{p\} \in I_{W_t}$, then the new $repC_{\gamma,t}(a)$ is set equal to $repC_{\gamma,t-1}(a) \cup \{p\}$, hence it contains p and the statement of the lemma holds;
  - If $repC_{\gamma,t-1}(a) \cup \{p\} \notin I_{W_t}$, then all categories associated with p are used in any matching between $repC_{\gamma,t-1}(a)$ and the categories A ($repC_{\gamma,t-1}(a) \cup \{p\}$ cannot be the left endpoints of a matching for the bipartite graph (W_t, A, E)). Hence, for sure, p can be substituted with some point of $repC_{\gamma,t-1}(a)$ in the for each loop of the "else" branch (lines 51-54) and allow us to obtain a matching of equal size containing p.

At this point, we can conclude that p is always added to $R_{\gamma,t}$ at time $t = t(p)$. ■

Lemma 4.6 *Let W_t be the window at time t. For each $\gamma \in \Gamma$, the following holds. Let $a \in S$ be a point that entered A_γ at some point in time, stated $a \in A_{\gamma,t(a)}$ (we denote a as a point in the stream S and not in the window W_t because we need to specify the behaviour of all sets $repC_{\gamma,t}(a)$, even those orphaned whose $a \notin W_t$). For $t \geq t(a)$ we have:*

- *If $a \in A_{\gamma,t}$, then $repC_{\gamma,t}(a)$ is a maximal independent set for $W_t(a) = \{x \in W_t : (dist(a,x) \leq \frac{\delta\gamma}{2}) \land (t(a) \leq t(x))\}$.*
- *If $a \notin A_{\gamma,t}$, then $repC_{\gamma,t}(a)$ is a maximal independent set for $W_t(a) = \{x \in W_t : (dist(a,x) \leq \frac{\delta\gamma}{2}) \land (t(a) \leq t(x) \leq \hat{t})\}$ where $\hat{t} \leq t$ is the time step a was removed from A_γ ($a \in A_{\gamma,\hat{t}-1}$ but $a \notin A_{\gamma,\hat{t}}$). Note that $repC_{\gamma,t}(a)$ is a subset of $repC_{\gamma,\hat{t}-1}(a)$ of points still not expired.*

As we said previously, unfortunately the second lemma lacks a formal proof and for that reason we cannot guarantee the correctness and the approximation ratio of the solution. To cope with this problem, we present now an empirical study of correctness for Lemma 4.6, which is the only missing link in the chain of results we need in order to guarantee the procedures are correct and return a $(3 + \epsilon)$-approximation for the k-Center problem under transversal matroid constraints.

The experiment we carried out consisted in checking, for a list of various bipartite graphs we can denote as $(W_t(a), A; E)$, which subsets of $W_t(a)$ constituted the left endpoints of a maximum matching, or equivalently a maximal independent set for the corresponding matroid $M_{W_t(a)}$. Moreover, we checked which of those subsets of points maintained maximality over time, that is, as time progressed and we removed points from the window, which maximal $X_{t-1} \in I_{W_{t-1}(a)}$ implied $X_t \in I_{W_t(a)}$ was still maximal. In the end we saw that, in general, the set $repC_{\gamma,t}(a)$ always maintain maximality over $W_t(a)$ as time progresses and often it is the only subset of $W_t(a)$ to do so.

Not being satisfied just by the description of the experiments, let us now first present three graphical examples and subsequently give a reference to an external notebook that programmatically checks the same ideas on additional instances of bipartite graphs. Each graphical example will take an attraction c-point x_1 and the corresponding set $W_t(x_1) = \{x_1, x_2, x_3, x_4, x_5\}$ of points at distance at most $\frac{\delta\gamma}{2}$ from x_1, where t is the time step exactly before x_1 is removed from A_γ ($t = \hat{t} - 1$, the time step in which $W_t(x_1)$ contains the most amount of points). In each figure, we will consider the squares at the bottom as the points in $W_t(x_1)$ and the black circle at the top as the corresponding categories A.

For all three cases, we present two different set of figures:

- The first set shows how the heuristic solution $repC_{\gamma,t}(x_1)$ for $x_1 \in A_{\gamma,t}$ is computed one step at the time, following lines 48-54 of procedure UPDATE(p) as the window slides from $t - 4$ to t.

- The second set instead exhibits all the maximal matchings for the bipartite graph $(W_t(x_1), A; E)$, highlighting their left endpoints in grey and the arcs of the matching with dashed lines. Furthermore, for each figure, we show at the bottom the time step for which the matching is not going to be maximal anymore.

Now that we have a clear idea on what the graphical example contains, let us present them one after the other accompanied by short comments on what we can extract from watching these figures.

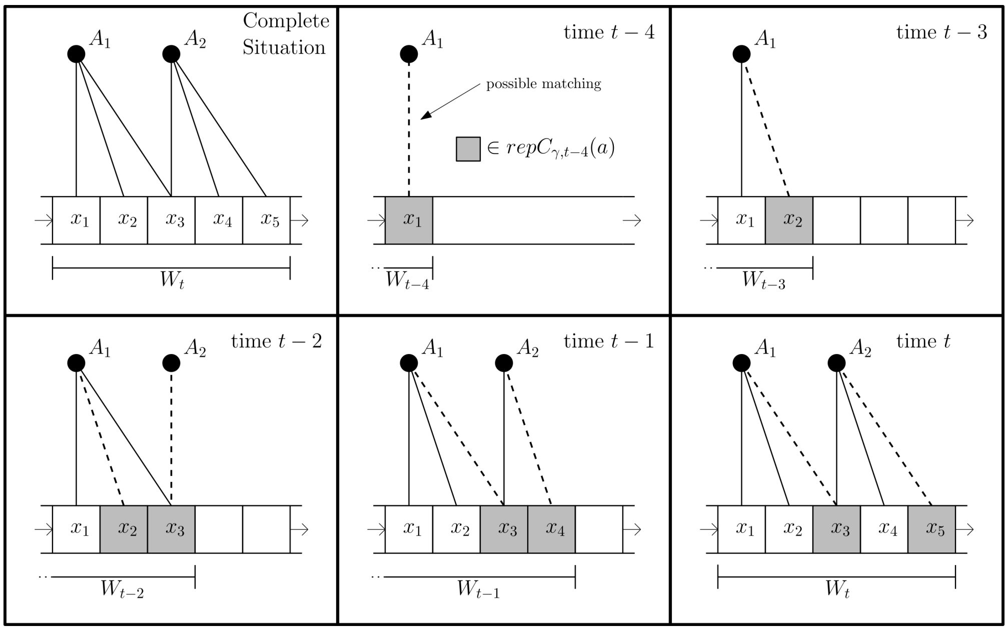
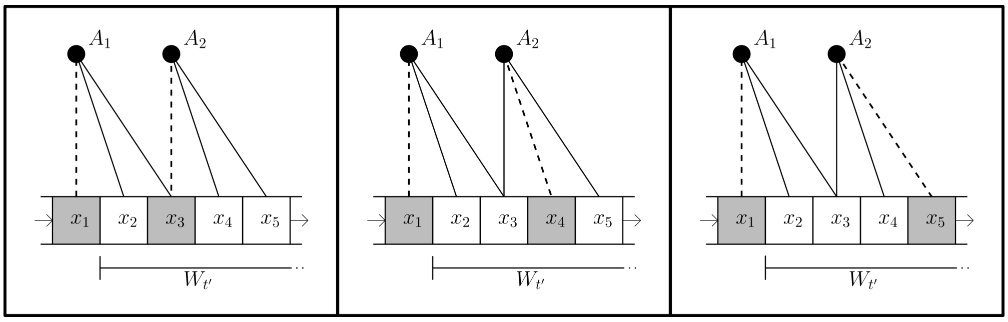


Figure 4.1: First graphical example of the heuristic used in the second solution for transversal matroid

As we can see above from Figure 4.1, for the single $W_t(x_1)$ with $x_1 \in A_{\gamma,t}$, the heuristic builds the maximal independent set $repC_{\gamma,t}(a)$ incrementally, substituting points only if the independent set cannot be enlarged further, namely, at times $t-3$, $t-1$ and t. In order to show that, in this example, our solution is actually the best we can hope for, we presented below Figure 4.2, containing all the maximal matchings for $W_t(x_1)$. Notably, we can see that all the maximal matchings for $W_t(x_1)$ decrease in size at most when our choice does. Furthermore, we see that the heuristic choice is the only one capable of maintaining a maximal matching at all time steps $t' \geq t$, instead all the other possibilities at some point are not capable anymore of maintaining maximality.



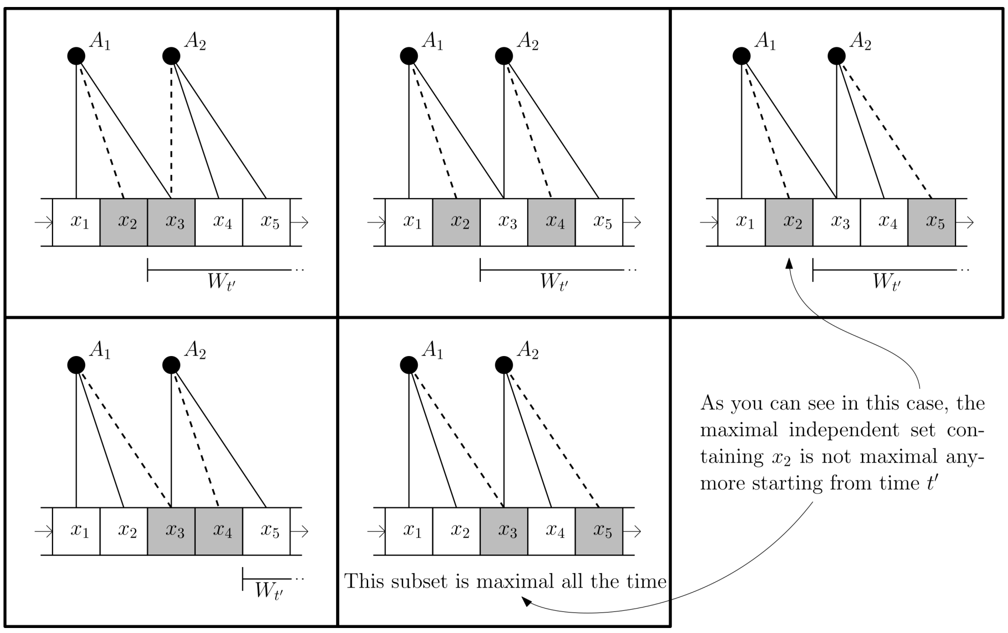

A_1
A_2
x_1
x_2
x_3
x_4
x_5
$W_{t'}$
As you can see in this case, the maximal independent set containing x_2 is not maximal anymore starting from time t'
This subset is maximal all the time

Figure 4.2: First graphical example of all the maximal matchings and their left endpoints for $W_t(a)$

Next, let us present another example. In this case, differently to what happened in the previous one, we can see that the subsets of categories $\{A_1, A_2\}$ may never be covered entirely by a matching. This happens because there are more categories for a subset of points than the number of points we can use to cover them. Formally this is a result of the Hall's Marriage theorem [16], whose modified statement is presented here for completeness.

Theorem 4.7 (Hall's Marriage Theorem) *Let $G = (W_t(x_1), A; E)$ be a finite bipartite graph. An A-perfect matching (also called an A-saturating matching) is a matching which covers every category in A. For a subset of categories $X \subseteq A$, if we consider $N_G(X)$ as the neighborhood of X in G (the set of neighbor points in $W_t(x_1)$ for X), then a X-perfect matching exists (a matching which cover every category in X) if and only if for every subset $W \subseteq X$ we have $|W| \leq |N_G(W)|$, which means having at least $|W|$ neighbor points in $W_t(x_1)$ for each W.*

In our case in order for $\{A_1, A_2\}$ to be covered entirely we must have at least two neighbor points, but as we can see we only have one. That being said, even if a maximal matching of size four does not exist, we can still assert that $repC_{\gamma,t}(a)$ is the only subset of points capable of keeping a maximal matching as time advances and the set $W_t(a)$ shrinks starting from $t' = t$ forward.

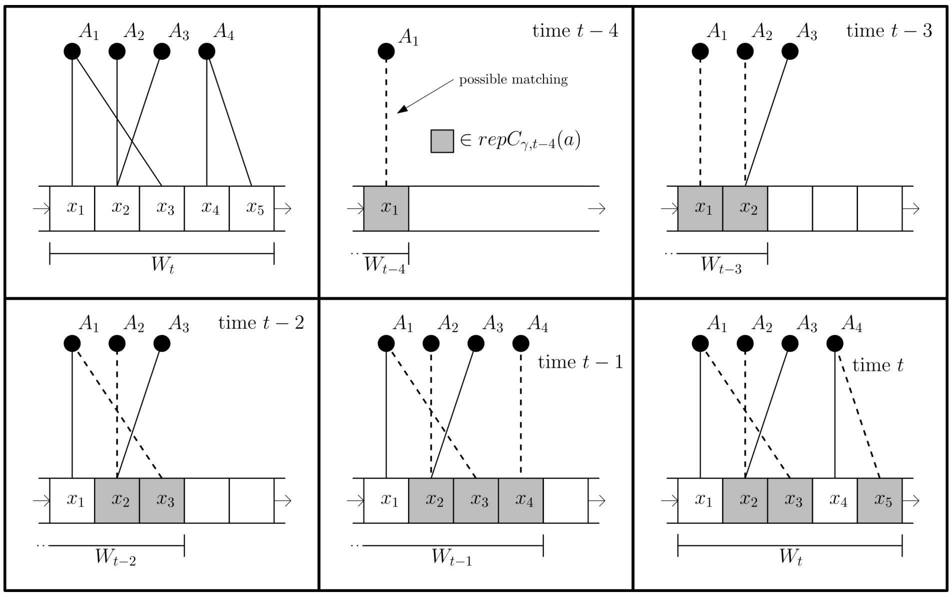


Figure 4.3: Second graphical example of the heuristic used in the second solution for transversal matroid

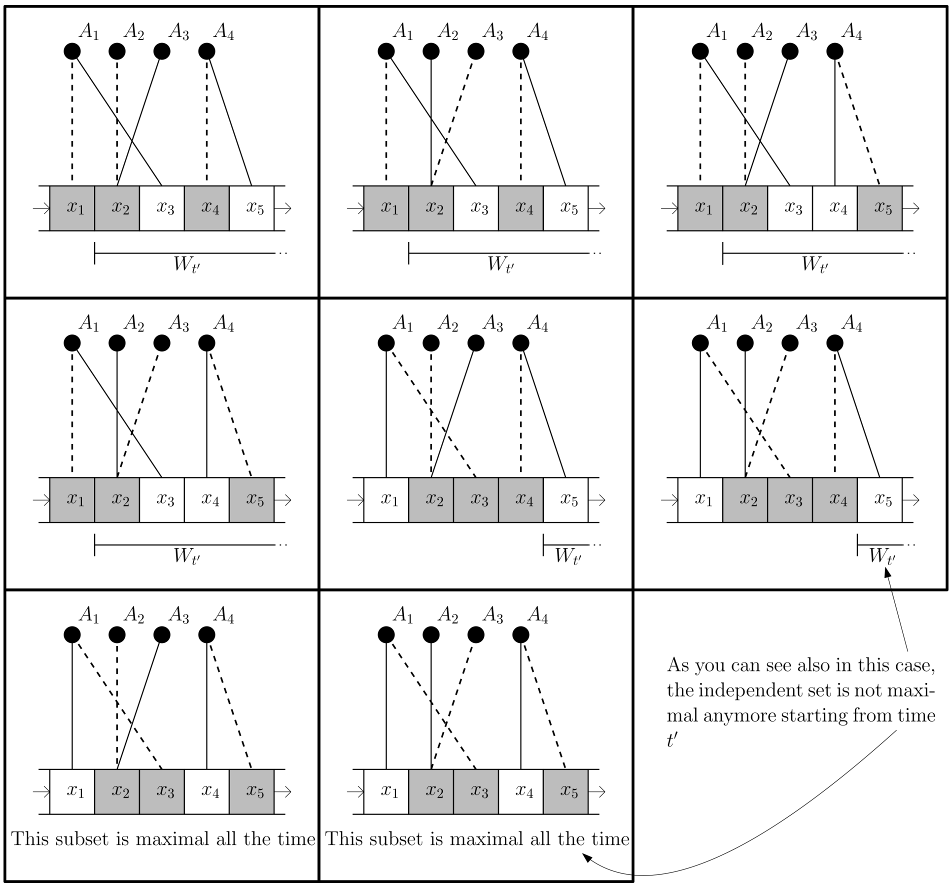

A_1 A_2 A_3 A_4
x_1 x_2 x_3 x_4 x_5
$W_{t'}$
As you can see also in this case, the independent set is not maximal anymore starting from time t'
This subset is maximal all the time
This subset is maximal all the time

Figure 4.4: Second graphical example of all the maximal matchings for $W_t(a)$

Below we add one last example without additional comments, since the same conclusion obtained above can be drawn by looking at this case.

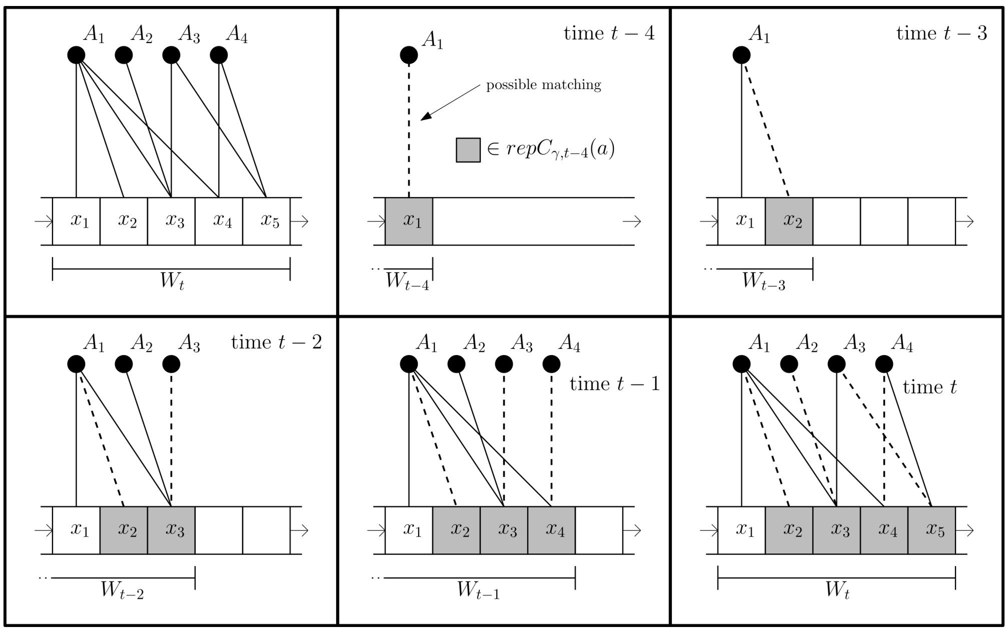
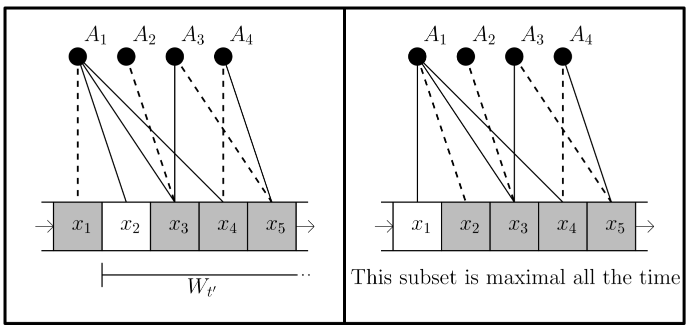


Figure 4.5: Third graphical example of the heuristic used in the second solution for transversal matroid


A_1 A_2 A_3 A_4
x_1 x_2 x_3 x_4 x_5
$W_{t'}$
This subset is maximal all the time

As you can see also in this case, the first independent set is not maximal anymore starting from time t'

Figure 4.6: Third graphical example of all the maximal matchings and their left endpoints for $W_t(a)$

To conclude the empirical study of Lemma 4.6, we add here an external resource where more instances like the ones above are considered and the same conclusions are obtained. These additional experiments were done using Python as programming language. The resulting source code can be found in the form of a Python Notebook on GitHub: https://github.com/lorenzocappe/matroid-center-sliding-window.

Even though we cannot formally prove that our solution is correct, the above examples allow us to conjecture that the technique satisfies Lemma 4.6, hence we regard it as a promising heuristic

Finally, let us move to the study of the time and space complexity for the second version of updated procedures. Since we assumed the rank of the transversal matroid to be equal to k, the bounds on the size of the support structures remain the same as the ones in Theorem 3.9. For this reason, we can avoid re-presenting that theorem and focus on providing just an updated version of Theorem 3.10.

Theorem 4.8 *Procedure UPDATE(p) runs in time*

$$O\left(k^2 \frac{\log(\Delta)}{\log(1+\beta)}\left(\frac{32}{\delta}\right)^{D_{W_t}}\right) \tag{4.4}$$

while procedure QUERY() runs in time

$$O\left(k^2 \frac{\log(\Delta)}{\log(1+\beta)} + k^2\left(\frac{32}{\delta}\right)^{D_{W_t}} + T_{\mathcal{A}}(R_\gamma \cup O_\gamma)\right) \tag{4.5}$$

where the most expensive operation in UPDATE(p) is the call to an oracle capable of verifying if a subset $X \subseteq W_t$ is an independent set of M_{W_t}.

Proof The complexity of UPDATE(p) is dominated by the computation at lines 47-54 where, for each guess $\gamma \in \Gamma$, we try to update the sets of representatives for all the $a \in E$. In particular, given a generic guess γ, in the worst case we have to update all sets of representatives there are (at most $|A_\gamma| = O(k(\frac{32}{\delta})^{D_{W_t}})$). For each set of representatives we try to augment the corresponding independent set and, if not successful, we try to remove a single representative and augment again the remaining subset. The claimed bound follows from the multiplications of this single factors ($O(k \cdot k(\frac{32}{\delta})^{D_{W_t}} \cdot \frac{\log(\Delta)}{\log(1+\beta)})$).

For what concerns QUERY(), we observe that the complexity is dominated by the size of $T = R_\gamma \cup O_\gamma$, the time required by $\mathcal{A}(R_\gamma \cup O_\gamma)$ to run and the calculation of C where, for each point $q \in AV_\gamma \cup OV_\gamma \cup RV_\gamma$ we perform the calculation $dist(q, C)$, which can be accomplished in time $O(k^2)$ (for at most $O(k)$ points we compute $dist(\cdot,\cdot)$ in time $O(k)$) for a given guess $\gamma \in \Gamma$. ■

As we can see the bounds on time and space complexity for this second version of the generalized algorithm are a factor $O(k)$ better than the previous one and for this reason we believe them to be more attractive.

5 Conclusions

In this thesis, we made significant contributions to the field of approximation algorithms for constrained clustering problems. We started by successfully addressed the Fair k-Center problem under the sliding window model, obtaining an efficient $(3 + \epsilon)$-approximation. We then moved to the study of the k-Center problem under a transversal matroid constraint, a specific version of the Matroid Center problem, and provided two different ideas on how to tackle the task based on the ideas underlying our solution to the Fair k-Center problem. The first proposed generalization provides a solution with formal guarantees of correctness and approximation factor, but at the cost of more computationally expensive procedures and slightly degraded working memory requirements. The second approach, instead, obtains better performances in both working memory and running time of the procedures, but lacks the formal guarantees for correctness and approximation factor. In this case, we bridged the missing gap with an empirical study of correctness.

We want to remark that this thesis does not tackle the more general Matroid Center problem under the sliding window model, for which no efficient solution is available in the open literature, but our approach provides a valuable contribution to the development of constrained versions of k-Center clustering under the sliding window model, where fairness is taken into account starting from the problem formulation and algorithms provide accurate equitable solutions. We believe this thesis could provide a good starting point for further generalizations.

In the spirit of continuation, we can highlight two other more directions on which research can move forward, as described below.

The first direction regards the search for robust solutions, which means designing algorithms capable of tolerating the presence of a certain number of outliers in the set of points. Often real-world data is much messier than we expect. While synthetic datasets may be very well structured and contain very clean data points, outliers are a common occurrence when acquiring data from a noisy natural source, where environmental factors and errors in the data acquisition stage cannot generally be ignored. In this light we ought to consider solutions where a certain number of points can be excluded from the computation. For an input parameter z, robust solutions for k-Center clustering require to minimize the $z + 1$-th largest distance between the points in the universe and the set of centers, instead of the classical maximum distance of *any* point from the set of centers. This allows us to ignore the distance of at most z outliers from the set of centers and consider them as not representative for the entire dataset. Robust solutions in the sliding window model are more difficult to compute than in the classical streaming model since we do not only have to take into account the incremental addition of points to the stream, but also the falling of points outside the window.

The literature already contains results working on robust solutions for the k-Center problem under the sliding window model, such as the work by Pellizzoni et al. [15]. Adapting this solution is viable, at the expense of a small deterioration in performance for the algorithm, and represents an interesting follow-up of the research presented in this thesis.

The second regards the conduction of an exhaustive array of experiments to thoroughly evaluate the actual performances of the proposed solutions in a practical setting, and to compare them to other already known solutions. Two very good examples of suites of experiments where evaluation and comparison of the proposed solution are accomplished effectively are the two papers from which this work originated [14, 2]. One of the possible experiments that can be carried out is a study on the effect of the doubling dimension D_{W_t} and aspect ratio Δ on the performances for the two procedures. This could broaden our perspective on which of the parameters influence the performances the most and help us come up in the future with new designs that work better in practice.

References

[1] Matteo Ceccarello, Andrea Pietracaprina, and Geppino Pucci. “Solving k-center Clustering (with Outliers) in MapReduce and Streaming, almost as Accurately as Sequentially”. In: *CoRR* abs/1802.09205 (2018). arXiv: 1802.09205. URL: http://arxiv.org/abs/1802.09205.

[2] Matteo Ceccarello et al. “Scalable and space-efficient Robust Matroid Center algorithms”. In: *J. Big Data* 10.1 (2023), p. 49. DOI: 10.1186/s40537-023-00717-4. URL: https://doi.org/10.1186/s40537-023-00717-4.

[3] Deeparnab Chakrabarty and Maryam Negahbani. “Generalized Center Problems with Outliers”. In: *CoRR* abs/1805.02217 (2018). arXiv: 1805.02217. URL: http://arxiv.org/abs/1805.02217.

[4] Danny Z. Chen et al. “Matroid and Knapsack Center Problems”. In: *CoRR* abs/1301.0745 (2013). arXiv: 1301.0745. URL: http://arxiv.org/abs/1301.0745.

[5] Flavio Chierichetti et al. “Fair Clustering Through Fairlets”. In: *CoRR* abs/1802.05733 (2018). arXiv: 1802.05733. URL: http://arxiv.org/abs/1802.05733.

[6] Flavio Chierichetti et al. “Matroids, Matchings, and Fairness”. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 2019, pp. 2212–2220. URL: https://proceedings.mlr.press/v89/chierichetti19a.html.

[7] Ashish Chiplunkar, Sagar Sudhir Kale, and Sivaramakrishnan Natarajan Ramamoorthy. “How to Solve Fair k-Center in Massive Data Models”. In: *CoRR* abs/2002.07682 (2020). arXiv: 2002.07682. URL: https://arxiv.org/abs/2002.07682.

[8] Vincent Cohen-Addad, Chris Schwiegelshohn, and Christian Sohler. "Diameter and k-Center in Sliding Windows". In: *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*. Ed. by Ioannis Chatzigiannakis et al. Vol. 55. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, 19:1–19:12. ISBN: 978-3-95977-013-2. DOI: `10.4230/LIPIcs.ICALP.2016.19`. URL: `http://drops.dagstuhl.de/opus/volltexte/2016/6340`.

[9] Mayur Datar and Rajeev Motwani. "The Sliding-Window Computation Model and Results". In: *Data Stream Management - Processing High-Speed Data Streams*. Ed. by Minos N. Garofalakis, Johannes Gehrke, and Rajeev Rastogi. Data-Centric Systems and Applications. Springer, 2016, pp. 149–165. DOI: `10.1007/978-3-540-28608-0\_7`. URL: `https://doi.org/10.1007/978-3-540-28608-0%5C_7`.

[10] Jack Edmonds and Delbert Ray Fulkerson. "Transversals and Matroid Partition". In: *Journal of Research of the National Bureau of Standards Section B Mathematics and Mathematical Physics* (1965), p. 147.

[11] Teofilo F. Gonzalez. "Clustering to minimize the maximum intercluster distance". In: *Theoretical Computer Science* 38 (1985), pp. 293–306. ISSN: 0304-3975. DOI: `https://doi.org/10.1016/0304-3975(85)90224-5`. URL: `https://www.sciencedirect.com/science/article/pii/0304397585902245`.

[12] Monika Rauch Henzinger, Prabhakar Raghavan, and Sridhar Rajagopalan. "Computing on data streams." In: *External memory algorithms* 50 (1998), pp. 107–118.

[13] Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. *Fair k-Center Clustering for Data Summarization*. 2019. arXiv: `1901.08628 [stat.ML]`.

[14] Paolo Pellizzoni, Andrea Pietracaprina, and Geppino Pucci. "Adaptive k-center and diameter estimation in sliding windows". In: *Int. J. Data Sci. Anal.* 14.2 (2022), pp. 155–173. DOI: `10.1007/s41060-022-00318-z`. URL: `https://doi.org/10.1007/s41060-022-00318-z`.

[15] Paolo Pellizzoni, Andrea Pietracaprina, and Geppino Pucci. "k-Center Clustering with Outliers in Sliding Windows". In: *Algorithms* 15.2 (2022).

ISSN: 1999-4893. DOI: 10.3390/a15020052. URL: https://www.mdpi.com/1999-4893/15/2/52.

[16] Wikipedia contributors. *Hall's marriage theorem — Wikipedia, The Free Encyclopedia*. [Online; accessed 6-July-2023]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Hall%27s_marriage_theorem&oldid=1159037091.

Acknowledgments

I want to thank my supervisors Professor Pietracaprina and Professor Pucci for their guidance and for allowing me to reach such an important step in my life. I also want to thank them for constantly supporting me with kindness and passion throughout the research training and thesis periods. I thank my mother and sister, who have always stood by me, especially in times of need, and who have never shied away from helping me. Finally, I want to thank my friends and colleagues with whom I have shared so much time.