4
Parameterized Complexity of Conflict as Bounded Arboricity Graph

Now we consider the conflicts in graphs with more complex settings than a matching or a collection of vertex disjoint cliques. For this, we consider the notion of *arboricity*. Recall, the arboricity of an undirected graph is the minimum number of forests into which its edges can be partitioned. A graph G is said to have *arboricity* d if the edges of G can be partitioned into at most d forests. Let $\mathscr{G}_d$ denote the family of graphs of arboricity d. We restrict the conflict graphs to be belonging to $\mathscr{G}_d$ in this chapter.

We study the following problems in "geometric settings" in the realm of Parameterized Complexity.

GRAPHICAL CONFLICT FREE SET COVER (GRAPHICAL CF-SC) **Parameter:** k
Input: A universe U of size n, a family $\mathscr{F}$ of size m of subsets of U, a conflict graph $CG_{\mathscr{F}}$ and a positive integer k.
Question: Does there exist a set cover $\mathscr{F}' \subseteq \mathscr{F}$ of size at most k such that $CG_{\mathscr{F}}[\mathscr{F}']$ is an independent set?

Let $(\mathscr{A},\mathscr{B})$-SET COVER denote a restriction of SET COVER, where every instance $(U,\mathscr{F},k)$ of SET COVER satisfies the property that $U \subseteq \mathscr{A}$ and $\mathscr{F} \subseteq \mathscr{B}$. For example in this setting, COVERING POINTS BY INTERVALS corresponds to $(\mathscr{A},\mathscr{B})$-SET COVER where $\mathscr{A}$ is the set of points on x-axis and $\mathscr{B}$ is the set of intervals on x-axis. Given $(\mathscr{A},\mathscr{B})$-SET COVER, the corresponding GRAPHICAL CF-SC corresponds to $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC.

Results in this Chapter We start by introducing following theorem which we shall prove later in the Section 4.1.

Theorem 1. *Let $(\mathscr{A},\mathscr{B})$-SET COVER be tractable and let $\mathscr{G}_d$ be the family of graphs of arboricity d. Then, the corresponding $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC is also tractable if $CG_{\mathscr{F}}$ belongs to $\mathscr{G}_d$. In particular we obtain following results when $CG_{\mathscr{F}}$ belongs to $\mathscr{G}_d$:*

- *If $(\mathscr{A},\mathscr{B})$-SET COVER admits a* FPT *algorithm with running time $\tau(k) \cdot n^{\mathscr{O}(1)}$, then $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC admits a* FPT *algorithm with running time $2^{\mathscr{O}(dk)} \cdot \tau(k) \cdot n^{\mathscr{O}(1)}$.*
- *If $(\mathscr{A},\mathscr{B})$-SET COVER admits a factor α-approximation running in time $n^{\mathscr{O}(1)}$ then $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC admits a factor α-*FPT*-approximation algorithm running in time $2^{\mathscr{O}(dk)} \cdot n^{\mathscr{O}(1)}$.*

The proof of Theorem 1 is essentially a black-box reduction to the non-conflict version of the problem. Thus, Theorem 1 covers a number of conflict-free versions of many fundamental geometric coverage problems as illustrated in Table 4.1. In light of Theorem 1, it is natural to ask whether or not, these problems admit polynomial time *approximation* algorithms. Unfortunately, we cannot expect these problems to admit even a factor $o(n)$-approximation algorithm. This is because for most of these problems even deciding whether there exists a conflict free solution, *with no restriction on the size of the solution*, is NP-complete (for example RAINBOW COVERING is NP-complete [3]). Thus, having an $o(n)$-approximation

$(\mathbb{R}^2,\mathscr{A})$-SC	**Complexity of** $(\mathbb{R}^2,\mathscr{A})$-SC	**Complexity of** $(\mathbb{R}^2,\mathscr{A})$-GRAPHICAL CF-SC
Disks/pseudo-disks	PTAS [69]	α-FPT approx., $\forall \alpha > 1$
Fat triangles of same size	$\mathscr{O}(1)$ [16]	$\mathscr{O}(1)$-FPT approx.
Fat objects in $\mathbb{R}^2$	$\mathscr{O}(\log^* \text{OPT})$ [6]	$\mathscr{O}(\log^* \text{OPT})$-FPT approx.
$\mathscr{O}(1)$ density objects in $\mathbb{R}^2$	PTAS [43]	α-FPT approx., $\forall \alpha > 1$
Objects with polylog density	QPTAS [43]	$2^{\mathscr{O}(k)} n^{\mathscr{O}(\log^* n)}$ time approx., $\forall \alpha > 1$
Objects with density $\mathscr{O}(1)$ in $\mathbb{R}^d$	PTAS [43]	α-FPT approx., $\forall \alpha > 1$
$(\mathscr{A},\mathscr{B})$-SET COVER where every instance $(U,\mathscr{F})$ has VC dimension d	$\mathscr{O}(d\log(d\text{OPT}))$ [12]	$\mathscr{O}(d\log(d\text{OPT}))$-FPT approx.
POINT GUARD ART GALLERY	$\mathscr{O}(\log \text{OPT})$ [11]	$\mathscr{O}(\log \text{OPT})$-FPT approx.
TERRAIN GUARDING	PTAS [57]	α-FPT approx., $\forall \alpha > 1$
$(\mathscr{P},\mathscr{I})$-SET COVER	Polynomial Time [27]	$2^{\mathscr{O}(dk)} \cdot n^{\mathscr{O}(1)}$-FPT algorithm

Table 4.1 : Corollaries of Theorem 1. Here $(\mathbb{R}^2,\mathscr{A})$-SET COVER ($(\mathbb{R}^2,\mathscr{A})$-SC) is a geometric set cover problem where $\mathbb{R}^2$ is a set of points in the plane and the covering objects are specified in the first column. The conflict graph for all the problems is $\mathscr{G}_d$, family of graphs of arboricity d, for some constant d. The entries in the second column give the approximation ratio of the $(\mathbb{R}^2,\mathscr{A})$-SC problem based on Theorem 1.

algorithm would imply a polynomial time algorithm for the decision version of the problem, which we do not expect unless P=NP. Hence, the best we can expect for the $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC problems is an FPT-approximation algorithm, as for many of them we can neither have an FPT algorithm, nor a polynomial time approximation algorithm.

Theorem 1 captures those families of conflict graphs that are "everywhere sparse". However, the $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC problem is also tractable if the conflict graphs belong to the family of cliques. When the conflict graph belongs to a "dense family" of graphs, we design a general theorem using matroid machinery as follows.

Let $(U,\mathscr{F},k)$ be an instance of SET COVER. In the matroidal model of representing conflicts, we are given a matroid $M=(E,\mathscr{J})$, where the ground set $E=\mathscr{F}$, and $\mathscr{J}$ is a family of subsets of $\mathscr{F}$ satisfying all the three properties of a matroid. In this paper we assume that $M=(E,\mathscr{J})$ is a *linear or representable matroid*, and the corresponding linear representation is given as part of the input. See Section 4.1.2 for the definition of matroid and related concepts. In the RAINBOW COVERING problem, let $\mathscr{Q}$ denote the family of conflict free subsets of intervals in $\mathscr{I}$. One can define a *partition matroid* on $\mathscr{F}$ such that $\mathscr{J}=\mathscr{Q}$. Thus, the question of finding a conflict free subset of intervals covering all the points in P becomes a problem of finding an independent set in $\mathscr{J}$ that covers all the points in P. The MATROIDAL CONFLICT FREE SET COVER problem (MATROIDAL CF-SC, in short) is defined similar to GRAPHICAL CF-SC. In particular, the input consists of a linear matroid $M=(\mathscr{F},\mathscr{J})$ over the ground set $\mathscr{F}$ such that the set cover $\mathscr{F}'\in\mathscr{J}$.

Theorem 2. *$(\mathscr{P},\mathscr{I})$-MATROIDAL CF-SC is* FPT *for all representable matroids $M=(\mathscr{I},\mathscr{J})$ defined over $\mathscr{I}$. In fact, given a linear representation, the algorithm runs in time $2^{\omega k}\cdot(n+m)^{\mathscr{O}(1)}$. Here, ω is the exponent in the running time of matrix multiplication.*

A graph is called a *cluster graph*, if all its connected components are cliques. Since cluster graphs can be captured by partition matroids, Theorem 2 implies that $(\mathscr{P},\mathscr{I})$-MATROIDAL CF-SC is FPT if $CG_{\mathscr{F}}$ belongs to the family of cluster graphs. We prove the above Theorem in Section 4.1.

We complement our algorithmic findings by a hardness reduction. Let $\mathscr{G}$ denote a family of graphs. Let $\mathscr{G}$-INDEPENDENT SET be the problem where the input is a graph $G \in \mathscr{G}$ and a positive integer k, where the objective is to decide whether there exists a set S of size at least k such that $G[S]$ is an independent set. We first start with Theorem 1.

Theorem 3. *Let $\mathscr{G}$ denote a family of graphs such that $\mathscr{G}$-INDEPENDENT SET is* W[1]*-hard. If $CG_{\mathscr{I}}$ belongs to $\mathscr{G}$, then $(\mathscr{P},\mathscr{I})$-GRAPHICAL CF-SC does not admit an* FPT *algorithm, unless* FPT =W[1].

The proof of Theorem 3 is a Turing reduction based on (n,k)-*perfect hash families* [70] that takes time $2^{\mathscr{O}(k)} \cdot n^{\mathscr{O}(1)}$. In fact, for any fixed $\mathscr{A}$ and $\mathscr{B}$, one should be able to follow this proof and show W[1]-hardness for $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC, where $CG_{\mathscr{F}}$ belongs to a graph family $\mathscr{G}$ for which $\mathscr{G}$-INDEPENDENT SET is W[1]-hard. We describe the proof of the Theorem in Section 4.2.

4.1 FIXED PARAMETER ALGORITHMS: PROOFS OF THEOREMS 1 AND 2

In this section we prove Theorems 1 and 2. Proof of Theorem 1 is based on a randomization scheme while the proof of Theorem 2 uses the idea of efficient computation of representative families [33].

4.1.1 FPT Algorithms for GRAPHICAL CF-SC

Our algorithm for Theorem 1 is essentially a randomized reduction from $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC to $(\mathscr{A},\mathscr{B})$-SET COVER, when the conflict graph has bounded arboricity. Towards, this we start with a forest decomposition of graphs of bounded arboracity and then apply a randomized process to obtain an instance of $(\mathscr{A},\mathscr{B})$-SET COVER. However, to design a deterministic algorithm we use the construction of universal sets. Towards that we will exploit the following definition and theorem.

Definition 4.1.1 ([70]). *An (n,t)-universal set $\mathscr{F}$ is a set of functions from $\{1,\dots,n\}$ to $\{0,1\}$, such that for every subset $S \subseteq \{1,\dots,n\}$, $|S| = t$, the set $\mathscr{F}|_S = \{f|_S \mid f \in \mathscr{F}\}$ is equal to the set 2^S of all the functions from S to $\{0,1\}$.*

Theorem 4.1.1 ([70]). *There is a deterministic algorithm with running time $\mathscr{O}(2^t t^{\mathscr{O}(\log t)} n \log n)$ that constructs an (n,t)-universal set $\mathscr{F}$ such that $|\mathscr{F}| = 2^t t^{\mathscr{O}(\log t)} \log n$.*

Now we are ready to give the proof of Theorem 1. For an ease of presentation we restate Theorem 1.

▶ **Theorem 1.1**[1] *Let $(\mathscr{A},\mathscr{B})$-SET COVER be tractable and let $\mathscr{G}_d$ be the family of graphs of arboricity d. Then, the corresponding $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC is also tractable if $CG_{\mathscr{F}}$ belongs to $\mathscr{G}_d$. In particular we obtain following results when $CG_{\mathscr{F}}$ belongs to $\mathscr{G}_d$:*

- *If $(\mathscr{A},\mathscr{B})$-SET COVER admits a* FPT *algorithm with running time $\tau(k) \cdot n^{\mathscr{O}(1)}$, then $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC admits a* FPT *algorithm with running time $2^{\mathscr{O}(dk)} \cdot \tau(k) \cdot n^{\mathscr{O}(1)}$.*
- *If $(\mathscr{A},\mathscr{B})$-SET COVER admits a factor α-approximation (or PTAS) running in time $n^{\mathscr{O}(1)}$ (or $n^{f(\varepsilon)}$) then $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC admits a factor α* FPT *approximation algorithm (or* FPT *PTAS) running in time $2^{\mathscr{O}(dk)} \cdot n^{\mathscr{O}(1)}$ (or $2^{\mathscr{O}(dk)} \cdot n^{f(\varepsilon)}$).*

Proof. Let $(U,\mathscr{F},CG_{\mathscr{F}},k)$ be an instance of $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC, where $CG_{\mathscr{F}}$, belongs to $\mathscr{G}_d$. Our algorithm has the following phases.

[1]The idea used in the proof of Theorem 1 is inspired by a proof used in [61].

Decomposing $CG_{\mathscr{F}}$ into Forests. We apply the known polynomial time algorithm [36] to decompose the graph $CG_{\mathscr{F}}$ into $T_1,\ldots,T_d$ where T_i is a forest in $CG_{\mathscr{F}}$ and $\bigcup_{i=1}^{d} E(T_i) = E(CG_{\mathscr{F}})$. Let v_{root} be a special vertex such that v_{root} does not belong to $V(CG_{\mathscr{F}}) = \mathscr{F}$. Now for every T_i, and for every connected component of T_i, we pick an arbitrary vertex and connect it to v_{root}. Now if we look at the tree induced on $V(T_i) \cup \{v_{\text{root}}\}$ then it is connected. We denote this tree by T_i'. Furthermore, we will treat each T_i' as a tree rooted at v_{root}. This automatically defines *parent-child* relationship among the vertices of T_i'. This completes the partitioning of the edge set of $CG_{\mathscr{F}}$ into forests.

Step 1: Randomized event and probability of success. Independently color the vertices of $CG_{\mathscr{F}}$ into blue and green uniformly at random. That is, we color the vertices of $CG_{\mathscr{F}}$ blue and green with probability $\frac{1}{2}$. Furthermore, we color $\{v_{\text{root}}\}$ to blue. Let $\mathscr{F}'$ be a solution to the instance $(U,\mathscr{F},CG_{\mathscr{F}},k)$. That is, $\mathscr{F}'$ is a conflict free set cover of size at most k. We say the following event to be *good*.

> Every vertex in $\mathscr{F}'$ is colored green and every *parent* of every vertex in $\mathscr{F}'$ in every tree T_i' is colored blue.

Let S_{parent} denote the set of parents of every vertex in $\mathscr{F}'$ in every tree T_i'. Since, we have at most d trees and the size of $\mathscr{F}'$ is upper bounded by k we have that $|S_{\text{parent}}| \le kd$. We say that $\mathscr{F}'$ (S_{parent}) is green (blue) to mean that every vertex in $\mathscr{F}'$ (S_{parent}) is colored green (blue). Thus,

$$\begin{aligned}
\Pr[\text{good event happens}] &= \Pr[\mathscr{F}' \text{ is green } \wedge S_{\text{parent}} \text{ is blue}] \\
&= \Pr[\mathscr{F}' \text{ is green}] \times \Pr[S_{\text{parent}} \text{ is blue}] \\
&\ge \frac{1}{2^{k(d+1)}}.
\end{aligned}$$

The second equality follows from the following fact. The set $\mathscr{F}'$ is an independent set in $CG_{\mathscr{F}}$ and $S_{\text{parent}} \subseteq N_{CG_{\mathscr{F}}}(\mathscr{F}') \cup \{v_{\text{root}}\}$. Thus, these sets are pairwise disjoint and hence the events in $\mathscr{F}'$ are colored green and S_{parent} is colored blue are independent.

Step 2: A cleaning process. Let $p = \frac{1}{2^{kd}}$. Now we apply a cleaning procedure so that we get a set Z such that $CG_{\mathscr{F}}[Z]$ is an independent set in $CG_{\mathscr{F}}$ and it contains $\mathscr{F}'$. Let $\mathscr{B}$ denote the set of vertices that have been colored blue. We start by deleting every vertex in $\mathscr{B}$. Now for every edge (f_1,f_2) in $CG_{\mathscr{F}}[V(CG_{\mathscr{F}}) \setminus \mathscr{B}]$, we do as follows. We know that (f_1,f_2) belongs to some tree T_i' and thus either f_1 is a *child* of f_2 or vice-versa. If f_1 is a child of f_2 then we delete f_1, otherwise we delete f_2. Let the resulting set of vertices be Z. By construction Z is an independent set in $CG_{\mathscr{F}}$. Next we show that $\mathscr{F}' \subseteq Z$ with probability $p/2^k$. Clearly, with probability $\frac{1}{2^k}$ we know that no vertex of $\mathscr{F}'$ is colored blue and thus with probability $\frac{1}{2^k}$ we know that $\mathscr{F}' \subseteq V(CG_{\mathscr{F}}) \setminus \mathscr{B}$. Observe that with probability p, we have that all the parents of $\mathscr{F}'$ in any tree T_i' have been colored blue. Thus, a vertex $x \in V(CG_{\mathscr{F}}) \setminus \mathscr{B}$, colored green, can not belong to $\mathscr{F}'$, if it is a child of some vertex in some tree T_i' after deleting the vertices of $\mathscr{B}$. This is the reason that when we delete a vertex from an edge (f_1,f_2), we delete the one which is a child in some tree T_i'. Thus, by deleting a vertex that is a child in an edge (f_1,f_2), we do not delete any vertex from $\mathscr{F}'$. This implies that with probability $\frac{1}{2^{k(d+1)}}$, we have that $\mathscr{F}' \subseteq Z$. This completes the proof.

Solving the problem. Let $\mathscr{Q}$ be a parameterized algorithm for $(\mathscr{A},\mathscr{B})$-Set Cover running in time $\tau(k) \cdot n^{\mathscr{O}(1)}$. Recall that $(U,\mathscr{F},CG_{\mathscr{F}},k)$ is an instance of $(\mathscr{A},\mathscr{B})$-Graphical CF-SC. Now to test whether there exists a conflict free set cover $\mathscr{F}'$ of size at most k, we run $\mathscr{Q}$ on (U,Z,k). If the algorithm returns Yes, we return the same for $(\mathscr{A},\mathscr{B})$-Graphical CF-SC. Else, we repeat the process by randomly finding another Z^* by following Steps 1 and 2 and then running the algorithm $\mathscr{Q}$ on the instance (U,Z^*,k) and returning the answer accordingly. We repeat the process $2^{k(d+1)}$ time. If we fail to detect whether $(U,\mathscr{F},k,CG_{\mathscr{F}})$ is a Yes instance of $(\mathscr{A},\mathscr{B})$-Graphical CF-SC in $2^{k(d+1)}$ rounds, then we return that the given instance is a No instance. Thus, if $(U,\mathscr{F},k,CG_{\mathscr{F}})$ is No instance of $(\mathscr{A},\mathscr{B})$-Graphical CF-SC, then we always return No.

However, if $(U,\mathscr{F},k,CG_{\mathscr{F}})$ is a Yes instance of $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC then there exists a set $\mathscr{F}'$, that is a conflict free set cover of size at most k. The probability that we will not find a set Z containing $\mathscr{F}'$ in $q=2^{k(d+1)}$ rounds is upper bounded by

$$\left(1-\frac{1}{q}\right)^q \leq \frac{1}{e}.$$

Thus, the probability that we will find a set Z containing $\mathscr{F}'$ in q rounds is at least $1-\frac{1}{e}\geq\frac{1}{2}$. Thus, if the given instance is a Yes instance then the algorithm succeeds with probability at least $\frac{1}{2}$. The running time of the algorithm is upper bounded by $\tau(k)\cdot 2^{k(d+1)}\cdot n^{\mathscr{O}(1)}$.

Derandomizing the algorithm. Now to design our deterministic algorithm all we will need to do is to replace the randomized coloring function with a deterministic coloring function that colors the vertices in $\mathscr{F}'$ to green and all the vertices in S_{parent} to green. To design such a coloring function we set $t=k(d+1)$, and use Theorem 4.1.1 to construct an (n,t)-universal set $\mathscr{F}$ such that $|\mathscr{F}|=2^t t^{\mathscr{O}(\log(t))}\log n$. The algorithm to construct $\mathscr{F}$ takes $\mathscr{O}(2^t t^{\mathscr{O}(\log(t))} n\log n)$. Finally, to derandomize our algorithm, rather than randomly coloring vertices with $\{\text{blue},\text{green}\}$, we go through each function f in the family $\mathscr{F}$ and view the vertices that have been assigned 0 as blue and others as green. By the properties of (n,t)-universal set we know that there exists a function f that correctly colors the vertices in $\mathscr{F}'$ with 1 and every vertex in S_{parent} with 0. Thus, the set Z_f that we will obtain by applying Step 2, will contain the set $\mathscr{F}'$. After this, the correctness of the algorithm follows from the correctness of the algorithm $\mathscr{Q}$. Thus, the running time of the algorithm is upper bounded by $\tau(k)\cdot|\mathscr{F}|\cdot n^{\mathscr{O}(1)}=\tau(k)\cdot 2^{k(d+1)+o(kd)}\cdot n^{\mathscr{O}(1)}$. This completes the proof of the first part.

Let $\mathscr{S}$ be a factor α-approximation algorithm for $(\mathscr{A},\mathscr{B})$-SET COVER running in time $n^{\mathscr{O}(1)}$. To obtain the desired FPT approximation algorithm with factor α, we do as follows. We only give the deterministic version of the algorithm based on the uses of universal sets. As before, let $(U,\mathscr{F},CG_{\mathscr{F}},k)$ be an instance of $(\mathscr{A},\mathscr{B})$-GRAPHICAL CF-SC, where $CG_{\mathscr{F}}$, belongs to $\mathscr{G}_d$. We again set $t=k(d+1)$, and use Theorem 4.1.1 to construct an (n,t)-universal set $\mathscr{F}$ such that $|\mathscr{F}|=2^t t^{\mathscr{O}(\log(t))}\log n$. The algorithm to construct $\mathscr{F}$ takes $\mathscr{O}(2^t t^{\mathscr{O}(\log(t))} n\log n)$. We go through each function f in the family $\mathscr{F}$ and view the vertices that have been assigned 0 as blue and others as green. If there exists a conflict set cover $\mathscr{F}'$ of size at most k, then by the properties of (n,t)-universal set we know that there exists a function f that correctly colors the vertices in $\mathscr{F}'$ with 1 and every vertex in S_{parent} with 0. Thus, the set Z_f we will obtain by applying the Step 2, will contain the set $\mathscr{F}'$. Thus, to design the approximation algorithm, for every $f\in\mathscr{F}$, we first construct Z_f. And for each such Z_f we run $\mathscr{S}$ on (U,Z_f,k). This could either return that there is No solution, or returns a solution $\mathscr{F}'$ which is a factor α-approximation to the instance (U,Z_f,k). If for some $f\in\mathscr{F}$, $\mathscr{S}$ returns $\mathscr{F}'$ of size at most αk when run on (U,Z_f,k) then the algorithm returns $\mathscr{F}'$. In all other cases the algorithm returns that the given instance is a No instance. In other words, our algorithm returns No if the following happens: For every f, $\mathscr{S}$ either returns that (U,Z_f,k) is a No instance or the size of the solution, $\mathscr{F}'$, returned by $\mathscr{S}$ when run on (U,Z_f,k), is more than αk. The correctness of the algorithm follows from the properties of universal sets and the correctness of the algorithm $\mathscr{S}$. The running time of the algorithm is upper bounded by: $|\mathscr{F}|\times$ Running time of $\mathscr{S}=2^{k(d+1)+o(kd)}\cdot n^{\mathscr{O}(1)}$. This completes the proof of the theorem. $\square$

4.1.2 FPT Algorithm for $(\mathscr{P},\mathscr{I})$-MATROIDAL CF-SC

In this section we will design an FPT algorithm proving Theorem 2. We start with restating the statement.

Theorem 2. *$(\mathscr{P},\mathscr{I})$-MATROIDAL CF-SC is* FPT *for all representable matroids $M=(\mathscr{I},\mathscr{J})$ defined over $\mathscr{I}$. In fact, given a linear representation, the algorithm runs in time $2^{\omega k}\cdot(n+m)^{\mathscr{O}(1)}$. Here, ω is the exponent in the running time of matrix multiplication.*

Towards that we need to define some basic notions related to representative families and results regarding their fast and efficient computation.

Matroids and Representative Families

In this subsection we give definitions related to matroids and representative families. For a broader overview on matroids we refer to[72], see also [21, Chapter 12].

Definition 4.1.2. *A pair $M = (E, \mathscr{I})$, where E is a ground set and $\mathscr{I}$ is a family of subsets (called independent sets) of E, is a* matroid *if it satisfies the following conditions:*

(I1) $\emptyset \in \mathscr{I}$.

(I2) *If $A' \subseteq A$ and $A \in \mathscr{I}$ then $A' \in \mathscr{I}$.*

(I3) *If $A, B \in \mathscr{I}$ and $|A| < |B|$, then there is $e \in (B \setminus A)$ such that $A \cup \{e\} \in \mathscr{I}$.*

An inclusion wise maximal set of $\mathscr{I}$ is called a *basis* of the matroid. Using axiom (I3) it is easy to show that all the bases of a matroid have the same size. This size is called the *rank* of the matroid M, and is denoted by rank(M).

Linear Matroids. Let A be a matrix over an arbitrary field $\mathbb{F}$ and let E be the set of columns of A. For A, we define matroid $M = (E, \mathscr{I})$ as follows. A set $X \subseteq E$ is independent (that is $X \in \mathscr{I}$) if the corresponding columns are linearly independent over $\mathbb{F}$. The matroids that can be defined by such a construction are called *linear matroids*, and if a matroid can be defined by a matrix A over a field $\mathbb{F}$, then we say that the matroid is representable over $\mathbb{F}$. That is, a matroid $M = (E, \mathscr{I})$ of rank d is representable over a field $\mathbb{F}$ if there exist vectors in $\mathbb{F}^d$ corresponding to the elements such that linearly independent sets of vectors correspond to independent sets of the matroid. A matroid $M = (E, \mathscr{I})$ is called *representable* or *linear* if it is representable over some field $\mathbb{F}$.

Partition matroids. A partition matroid $M = (E, \mathscr{I})$ is defined by a ground set E being partitioned into (disjoint) sets $E_1, \dots, E_\ell$ and by ℓ non-negative integers $k_1, \dots, k_\ell$. A set $X \subseteq E$ is independent if and only if $|X \cap E_i| \leq k_i$ for all $i \in \{1, \dots, \ell\}$.

Proposition 4.1.1 ([65])**.** *A representation over a field of size $\mathscr{O}(|E|)$ of a partition matroid can be constructed in polynomial time.*

Representative Families. Now we define the notion of q-representative family and state the results about its efficient computation.

Definition 4.1.3 (**q-Representative Family** [65])**.** *Given a matroid $M = (E, \mathscr{I})$ and a family $\mathscr{S}$ of subsets of E, we say that a subfamily $\widehat{\mathscr{S}} \subseteq \mathscr{S}$ is q-*representative *for $\mathscr{S}$ if the following holds: for every set $Y \subseteq E$ of size at most q, if there is a set $X \in \mathscr{S}$ disjoint from Y with $X \cup Y \in \mathscr{I}$, then there is a set $\widehat{X} \in \widehat{\mathscr{S}}$ disjoint from Y with $\widehat{X} \cup Y \in \mathscr{I}$. If $\hat{\mathscr{S}} \subseteq \mathscr{S}$ is q-representative for $\mathscr{S}$ we write $\widehat{\mathscr{S}} \subseteq^q_{rep} \mathscr{S}$.*

In other words if some independent set in $\mathscr{S}$ can be extended to a larger independent set by adding q new elements, then there is a set in $\widehat{\mathscr{S}}$ that can be extended by the same q elements.

Lemma 4.1.1 ([33])**.** *Let $M = (E, \mathscr{I})$ be a matroid and $\mathscr{S}$ be a family of subsets of E. If $\mathscr{S}' \subseteq^q_{rep} \mathscr{S}$ and $\widehat{\mathscr{S}} \subseteq^q_{rep} \mathscr{S}'$, then $\widehat{\mathscr{S}} \subseteq^q_{rep} \mathscr{S}$.*

Theorem 4.1.2 ([33])**.** *Let $M = (E, \mathscr{I})$ be a linear matroid of rank $p + q = k$, $\mathscr{S} = \{S_1, \dots, S_t\}$ be a p-family of independent sets. Given a representation A_M of M over a field $\mathbb{F}$, we can find $\widehat{\mathscr{S}} \subseteq^q_{rep} \mathscr{S}$ of size at most $\binom{p+q}{p}$ in $\mathscr{O}\left(\binom{p+q}{p} t p^\omega + t \binom{p+q}{q}^{\omega-1}\right)$ operations over $\mathbb{F}$.*

Theorem 4.1.3 ([59]). *Let $M = (E, \mathscr{J})$ be a linear matroid of rank n and let $\mathscr{S} = \{S_1, \ldots, S_t\}$ be a p-family of independent sets. Let A be a $n \times |E|$ matrix representing M over a field $\mathbb{F}$, where $\mathbb{F} = \mathbb{F}_{p^\ell}$ or $\mathbb{F}$ is $\mathbb{Q}$. Then there is a deterministic algorithm computing $\widehat{\mathscr{S}} \subseteq^q_{rep} \mathscr{S}$ of size $np\binom{p+q}{p}$ in $\mathscr{O}\left(\binom{p+q}{p} t p^3 n^2 + t\binom{p+q}{q}^{\omega-1}(pn)^{\omega-1}\right) + (n+|E|)^{\mathscr{O}(1)}$ operations over $\mathbb{F}$.*

Algorithm for $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC

Now we have gathered all the tools required to prove Theorem 2. Let $(P, \mathscr{I}, k, M = (\mathscr{I}, \mathscr{J}))$ be an instance of $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC, where P is a set of points on the x-axis, $\mathscr{I} = \{I_1, \ldots, I_m\}$ is a set of intervals on the x-axis and $M = (\mathscr{I}, \mathscr{J})$ is a matroid over the ground set $\mathscr{I}$. The objective is to find a set cover $\mathscr{S} \subseteq \mathscr{I}$ of size at most k such that $\mathscr{S} \in \mathscr{J}$.

To design our algorithm for $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC, we will use efficient computation of representative families applied on a dynamic programming algorithm. Let $P = \{p_1, \ldots, p_n\}$ denote the set of points sorted from left to right. Next we introduce the notion of family of partial solutions. Let

$$\mathscr{P}^i = \left\{X \;\middle|\; X \subseteq \mathscr{I}, X \in \mathscr{J},\ |X| \le k,\ X \text{ covers } p_1, \ldots, p_i\right\}$$

denote the family of subset of intervals of size at most k that covers first i points and are independent in the matroid $M = (\mathscr{I}, \mathscr{J})$. Furthermore, for every $1 \le j \le k$, by $\mathscr{P}^{ij}$, we denote the subset of $\mathscr{P}^i$ containing sets of size *exactly* j. Thus,

$$\mathscr{P}^i = \biguplus_{j=1}^{k} \mathscr{P}^{ij}.$$

In this subsection whenever we talk about independent sets, these are independent sets of the matroid $M = (\mathscr{I}, \mathscr{J})$. Furthermore, we *assume that we are given, A_M, the linear representation of M.* Without loss of generality we can assume that A_M is a $n' \times |\mathscr{I}|$ matrix, where $n' \le |\mathscr{I}|$.

Observe that $(P, \mathscr{I}, k, M = (\mathscr{I}, \mathscr{J}))$ is a Yes instance of $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC if and only if $\mathscr{P}^n$ is non-empty. This implies that $\mathscr{P}^n$ is non-empty if and only if $\widehat{\mathscr{P}^n} \subseteq^0_{rep} \mathscr{P}^n$ is non-empty. We capture this into the following lemma.

Lemma 4.1.2. *Let $(P, \mathscr{I}, k, M = (\mathscr{I}, \mathscr{J}))$ be an instance of $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC. Then, $(P, \mathscr{I}, k, M = (\mathscr{I}, \mathscr{J}))$ is a* Yes *instance of $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC if and only if $\mathscr{P}^n$ is non-empty if and only if $\widehat{\mathscr{P}^n} \subseteq^0_{rep} \mathscr{P}^n$ is non-empty.*

For an ease of presentation by $\mathscr{P}^0$, we denote the set $\{\emptyset\}$. The next lemma provides an efficient computation of the family $\widehat{\mathscr{P}^i} \subseteq^{1\cdots k}_{rep} \mathscr{P}^i$. In particular, for every $1 \le i \le n$, we compute

$$\widehat{\mathscr{P}^i} = \bigcup_{j=1}^{k} \left(\widehat{\mathscr{P}^{ij}} \subseteq^{k-j}_{rep} \mathscr{P}^{ij}\right).$$

Lemma 4.1.3. *Let $(P, \mathscr{I}, k, M = (\mathscr{I}, \mathscr{J}))$ be an instance of $(\mathscr{P}, \mathscr{I})$-Matroidal CF-SC. Then for every $1 \le i \le n$, a collection of families $\widehat{\mathscr{P}^i} \subseteq^{1\cdots k}_{rep} \mathscr{P}^i$, of size at most $2^k \cdot |\mathscr{I}| \cdot k$ can be found in time $2^{\omega k} \cdot (n + |\mathscr{I}|)^{\mathscr{O}(1)}$.*

Proof. We describe a dynamic programming based algorithm. Let $P = \{p_1, \ldots, p_n\}$ denote the set of points sorted from left to right and $\mathscr{D}$ be a $n+1$-sized array indexed with $\{0, \ldots, n\}$. The entry $\mathscr{D}[i]$ will store a family $\widehat{\mathscr{P}^i} \subseteq^{1\cdots k}_{rep} \mathscr{P}^i$. We fill the entries in the matrix $\mathscr{D}$ in the increasing order of index. For $i = 0$, $\mathscr{D}[i] = \{\emptyset\}$. Let $i \in \{0, 1, \ldots, n\}$ and assume that we have filled all the entries until the row i (i.e, $\mathscr{D}[i]$ will contain a family $\widehat{\mathscr{P}^i} \subseteq^{1\cdots k}_{rep} \mathscr{P}^i$). For any interval $I \in \mathscr{I}$, let ℓ_I be the lowest index in $[n]$ such that p_{ℓ_I} is covered by I. Let $\mathscr{Z}_{i+1}$

denote the set of intervals $I \in \mathscr{I}$ that covers the point p_{i+1}. Now we compute

$$\mathscr{N}^{i+1} = \bigcup_{I \in \mathscr{Z}_{i+1}} (\mathscr{D}[\ell_I - 1] \bullet \{I\}) \cap \mathscr{J} \tag{4.1}$$

Notice that in the Equation 4.1, the union is taken over $I \in \mathscr{Z}_{i+1}$. Since for any $I \in \mathscr{Z}_{i+1}$, I covers p_{i+1}, the value $\ell_I - 1$ is strictly less than $i+1$ and hence Equation 4.1 is well defined. Let $\mathscr{N}^{(i+1)j}$ denote the subset of $\mathscr{N}^{i+1}$ containing subsets of size exactly j.

Claim 4.1.1. $\mathscr{N}^{i+1} \subseteq_{rep}^{1\cdots k} \mathscr{P}^{i+1}$.

Proof. Let $S \in \mathscr{P}^{(i+1)j}$ and Y be a set of size $k-j$ (which is essentially an independent set of M) such that $S \cap Y = \emptyset$ and $S \cup Y \in \mathscr{J}$. We will show that there exists a set $\widehat{S} \in \mathscr{N}^{(i+1)}$ such that $\widehat{S} \cap Y = \emptyset$ and $\widehat{S} \cup Y \in \mathscr{J}$. This will imply the desired result.

Since S covers $\{p_1, \dots, p_{i+1}\}$, there is an interval J in S which covers p_{i+1}. Since S covers $\{p_1, \dots, p_{i+1}\}$ and J covers p_{i+1}, the set of intervals $S' = S \setminus \{J\}$ covers $\{p_1, \dots, p_{i+1}\} \setminus \{p_{\ell_J}, \dots, p_{i+1}\}$ and J covers $\{p_{\ell_J}, \dots p_{i+1}\}$. Let $Y' = Y \cup \{J\}$. Notice that $S' \cup Y' = S \cup Y \in \mathscr{J}$, $|S'| = j-1$, $|Y'| = k-j+1$ and S' covers $\{p_1, \dots, p_{i+1}\} \setminus \{p_{\ell_J}, \dots, p_{i+1}\}$. This implies that $S' \in \mathscr{P}^{(\ell_J - 1)(j-1)}$ and by our assumption that $\mathscr{D}[\ell_J - 1]$ contain $\widehat{\mathscr{P}}^{(\ell_J-1)(j-1)} \subseteq_{rep}^{k-j+1} \mathscr{P}^{(\ell_J-1)(j-1)}$, we have that there exists $S^* \in \mathscr{D}[\ell_J - 1]$ such that $S^* \cap Y' = \emptyset$ and $S^* \cup Y' \in \mathscr{J}$. By Equation 4.1, $S^* \cup \{J\}$ in $\mathscr{N}^{i+1}$, because $S^* \cup \{J\} \in \mathscr{J}$. Now we set $\widehat{S} = S^* \cup \{J\}$. Observe that $\widehat{S} \cap Y = \emptyset$ and $\widehat{S} \cup Y \in \mathscr{J}$. This completes the proof of the claim. □

We fill the entry for $\mathscr{D}[i+1]$ as follows.

$$\mathscr{D}[i+1] = \bigcup_{j=1}^{k} \left(\widehat{\mathscr{N}}^{(i+1)j} \subseteq_{rep}^{k-j} \mathscr{N}^{(i+1)j} \right) \tag{4.2}$$

In Equation 4.2, for every $1 \leq j \leq k$, $\mathscr{N}^{(i+1)j}$ denotes the subset of $\mathscr{N}^{(i+1)}$ containing sets of size *exactly* j and $\widehat{\mathscr{N}}^{(i+1)j}$ can be computed using either Theorem 4.1.2 or Theorem 4.1.3 based on the dimension of A_M. Theorem 4.1.2 can be applied when the given matrix A_M has at most k rows and Theorem 4.1.3 can be applied even when the number of rows in A_M is not upper bounded by k. So, in Equation 4.2, we use Theorem 4.1.3 to compute the representative families. Lemma 4.1.1 and Claim 4.1.1 implies that $\mathscr{D}[i+1] \subseteq_{rep}^{1\cdots k} \mathscr{P}^{i+1}$.

Now we analyse the running time of the algorithm. Consider the time to compute $\mathscr{D}[i+1]$. We already have computed the family corresponding to $\mathscr{D}[r]$ for all $r \in [i]$. By Theorem 4.1.3, for any $r \in [i]$ and $j \in [k]$, the subset of $\mathscr{D}[r]$ containing sets of size exactly j is upper bounded by $|\mathscr{I}| \cdot k \cdot \binom{k}{j}$. Hence, the cardinality of $\mathscr{N}^{(i+1)j}$ is upper bounded by $|\mathscr{I}|^2 \cdot n \cdot k \cdot \binom{k}{j}$. Thus, by Theorem 4.1.3, the time to compute $\widehat{\mathscr{N}}^{(i+1)j} \subseteq_{rep}^{k-j} \mathscr{N}^{(i+1)j}$ is bounded by $\left(\binom{k}{j}^2 + \binom{k}{j}^{\omega}\right)(n + |\mathscr{I}|)^{\mathscr{O}(1)} = \binom{k}{j}^{\omega} \cdot (n + |\mathscr{I}|)^{\mathscr{O}(1)}$ number of operation over the field in which A_M is given and $|\widehat{\mathscr{N}}^{(i+1)j}| \leq |\mathscr{I}| \cdot k \cdot \binom{k}{j}$. Hence the total running time to compute $\mathscr{D}[i+1]$ for any $i+1 \in [n]$ is

$$\sum_{j=1}^{k} \binom{k}{j}^{\omega} \cdot (n + |\mathscr{I}|)^{\mathscr{O}(1)}) = 2^{\omega k} \cdot (n + |\mathscr{I}|)^{\mathscr{O}(1)}.$$

By Theorem 4.1.3, the cardinality of $\mathscr{D}[i+1]$ is bounded by,

$$|\mathscr{D}[i+1]| = \sum_{j=1}^{k} |\widehat{\mathscr{N}^{(i+1)j}}| \leq \sum_{j=1}^{k} |\mathscr{I}| \cdot k \cdot \binom{k}{j} = 2^k |\mathscr{I}| \cdot k.$$

This completes the proof.

□

Theorem 2 follows from Lemma 4.1.2 and 4.1.3. Now we explain an application of Theorem 2. Consider the problem $(\mathscr{P},\mathscr{I})$-Graphical CF-SC, where $CG_{\mathscr{I}}$ is a cluster graph. Let $(P,\mathscr{I},CG_{\mathscr{I}},k)$ be an instance of $(\mathscr{P},\mathscr{I})$-Graphical CF-SC. Let $C_1,\ldots C_t$ be the connected components of $CG_{\mathscr{I}}$, where each C_i is a clique for all $i \in [t]$. In any solution we are allowed to pick at most one vertex (an interval) from C_i for any $i \in [t]$. This information can be encoded using a partition matroid $M = (\mathscr{I} = V(C_1) \uplus \ldots \uplus V(C_t), \mathscr{J})$ where any subset $\mathscr{I}' \subseteq \mathscr{I}$ is independent in M if and only if $|\mathscr{I}' \cap V(C_i)| \leq 1$ for any $i \in [t]$. As a result, by applying Theorem 2 along with Proposition 4.1.1, we get the following corollary.

Corollary 4.1.1. *$(\mathscr{P},\mathscr{I})$-Graphical CF-SC, when $CG_{\mathscr{I}}$ is a cluster graph, can be solved in time $2^{\omega k} \cdot (n+|\mathscr{I}|)^{\mathscr{O}(1)}$.*

4.2 HARDNESS

In this section we prove the following theorem.

Theorem 3. *Let $\mathscr{G}$ denote a family of graphs such that $\mathscr{G}$-Independent Set is* W[1]*-hard. If $CG_{\mathscr{I}}$ belongs to $\mathscr{G}$, then $(\mathscr{P},\mathscr{I})$-Graphical CF-SC does not admit an* FPT *algorithm, unless* FPT =W[1].

Towards proving Theorem 3, we give a Turing reduction from $\mathscr{G}$-Independent Set to $(\mathscr{P},\mathscr{I})$-Graphical CF-SC where $CG_{\mathscr{I}}$ belongs to $\mathscr{G}$. To give such a reduction we need the notion of (n,k)*-perfect hash families*. Towards that, we first define (n,k)-perfect hash family and state a theorem about efficient computation of these objects.

Definition 4.2.1 ([70])**.** *An (n,k)-perfect hash family $\mathscr{F}$ is a set of functions from $\{1,\ldots,n\}$ to $\{1,\ldots,k\}$, such that for every subset $S \subseteq \{1,\ldots,n\}$, $|S| = k$, there is a function $f \in \mathscr{F}$ such that $f|_S$ is injective.*

Theorem 4.2.1 ([70])**.** *There is a deterministic algorithm with running time $e^k k^{\mathscr{O}(\log k)} n \log n$ that constructs an (n,k)-perfect hash family $\mathscr{F}$ of cardinality at most $e^k k^{\mathscr{O}(\log k)} \log n$.*

Proof of Theorem 3. We give a Turing reduction from $\mathscr{G}$-Independent Set. Let (G,k) be an instance of $\mathscr{G}$-Independent Set and $n = |V(G)|$. We first apply Theorem 4.2.1 and construct a (n,k)-perfect hash family $\mathscr{F}$. Then, we construct $|\mathscr{F}|$ many instances of $(\mathscr{P},\mathscr{I})$-Graphical CF-SC such that the parameter value in each instance is k and (G,k) is a Yes instance of $\mathscr{G}$-Independent Set if and only if at least one instance constructed is a Yes instance of $(\mathscr{P},\mathscr{I})$-Graphical CF-SC. More over the running time of our reduction will be $e^k k^{\mathscr{O}(\log k)} n \log n$.

Now we give details about the reduction. As mentioned earlier let (G,k) be the given instance of $\mathscr{G}$-Independent Set and $n = |V(G)|$. For each $f \in \mathscr{F}$ we create an instance $(P,\mathscr{I},G,k)$. Notice that in our reduction the conflict graph is G itself. Now we create a set of points $P = \{p_1 = (1,0), p_2 = (2,0), \ldots, p_k = (k,0)\}$. We can think of f as a coloring function on $V(G)$, where each vertex in v gets a color from $[k]$. For each vertex $v \in V(G)$ we create an interval $I_v = [f(v)-0.5, f(v)+0.5]$. Notice that any interval I_v covers only the point $p_{f(v)}$. See Fig. 4.1 for an illustration.

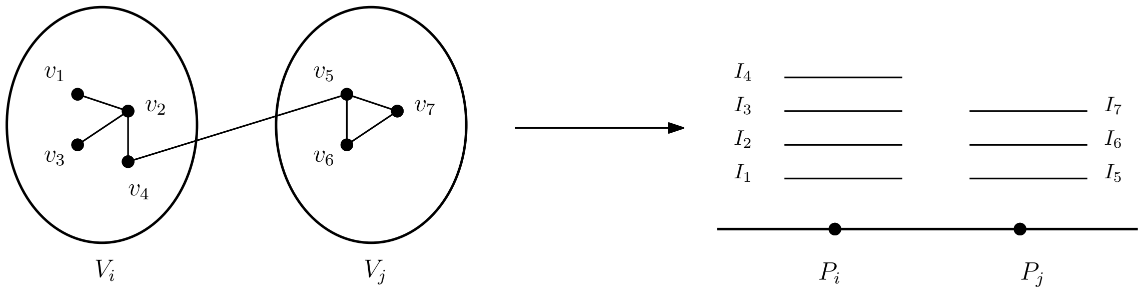

Figure 4.1 : Here V_i and V_j are the set of vertices in G which are colored by i and j, respectively.

Now we prove that (G,k) is a Yes instance of $\mathscr{G}$-Independent Set if and only if at least one instance constructed is a Yes instance of $(\mathscr{P},\mathscr{I})$-Graphical CF-SC. Suppose (G,k) is a Yes instance of $\mathscr{G}$-Independent Set and S is an independent set of size k in G. By the property of (n,k)-perfect hash family, there is a function f such that $f|_S$ is injective. Now consider the instance $(P,\mathscr{I},G,k)$ created for f. The set of intervals $\{I_v \ : \ v \in S\}$ covering P and S is independent in G. Hence $(P,\mathscr{I},G,k)$ is a Yes instance. Now suppose there is an instance $(P,\mathscr{I},G,k)$ created for a function $f \in \mathscr{F}$, such that $(P,\mathscr{I},G,k)$ is a Yes instance. This implies that there is a k sized independent set in G and hence (G,k) is a Yes instance of $\mathscr{G}$-Independent Set.

Because of Theorem 4.2.1, the running time of our reduction is $e^k k^{\mathscr{O}(\log k)} n \log n$. This completes the proof of the theorem. $\square$