Parameterized algorithms for the max $k$-set cover and related satisfiability problems

Edouard Bonnet$^{1,2}$, Vangelis Th. Paschos$^{1,2,3}$, Florian Sikora$^{1,2}$

$^1$ PSL Research University, Université Paris-Dauphine, LAMSAD\{edouard.bonnet,paschos,florian.sikora\}@lamsade.dauphine.fr
$^2$ CNRS UMR 7243
$^3$ Institut Universitaire de France

Abstract. We study the complexity of several parameterizations for max $k$-set cover. Given a family of subsets $S = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$ and an integer $p$, max $k$-set cover consists of finding a set $T$ of at most $k$ subsets covering at least $p$ elements. This problem, when parameterized by $k$, is W[2]-hard. Here, we settle the multiparameterized complexity of max $k$-set cover under pairs of parameters as $(k, \Delta)$ and $(k, f)$, where $\Delta = \max_i |S_i|$ and $f = \max_i \{|j |x_i \in S_j\}$. We also study parameterized approximability of the problem with respect to parameters $k$ and $p$. Then, we investigate some similar parameterizations of a satisfiability problem max sat-$k$ which is close to max $k$-set cover. Finally, we sketch an enhancement of the classes of the W[•] hierarchy that seems more appropriate for showing completeness of cardinality constrained W[•]-hard problems.

1 Introduction

In the max $k$-set cover problem, we are given a family of subsets $S = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$, and an integer $p$. The goal is to find a subcollection $T$ of at most $k$ subsets that covers at least $p$ elements. Max $k$-set cover as well as its graph-version max $k$-vertex cover are well known and important problems, since they are natural generalizations of min set cover and min vertex cover, respectively. Furthermore, max $k$-set cover is also significant from a practical point of view, since it arises quite frequently in several areas, for instance, in several location problems, particular when resources location is needed to perform a maximum coverage but the number of resources is restricted.

Max $k$-set cover is known to be NP-hard (setting $p = n$, max $k$-set cover becomes the seminal min set cover problem). It is also known to be approximable within a factor $1 - 1/e$, but, for any $\epsilon > 0$, no polynomial algorithm can approximate it within ratio $1 - 1/e + \epsilon$ unless P = NP [16].
Concerning the parameterized complexity of the problem, MAX k-SET COVER is W[2]-hard for the parameter k, by setting p = n since MIN SET COVER is W[2]-hard too (for more about the definition of the W[·] hierarchy, as well as for everything about the foundations of parameterized complexity, the interested reader is referred to [13]). An FPT algorithm with respect to the standard parameter p is given by Bläser in [3]. The main goal of the present paper is to establish multiparameterization results for MAX k-SET COVER and for MAX SAT-k which is a resembling satisfiability problem defined in Section 3.

For MAX k-SET COVER, several natural parameters as the standard parameter p, k, \( \Delta = \max_i |S_i| \) and \( f = \max_i |\{j|x_i \in S_j\}| \), where the quantity \( |\{j|x_i \in S_j\}| \) is commonly called the frequency of the element \( x_i \), can be involved in a complexity study of the problem. While the parameterized complexity regarding a single parameter is settled, the combinations of parameters constitute some novel open questions. Note that in several papers (see, for example, [4,6]), the parameterized complexity of several problems has been studied via a multiparameterized analysis involving sets of two parameters or more.

We first study multiparameterization of MAX k-SET COVER with respect to pairs of the parameters mentioned above (Section 2.1). We refine a technique for obtaining multiparameterized FPT algorithms developed in [4], called greediness-for-parameterization, which is based on branching algorithms.

Roughly, a branching algorithm extends a partial solution at each recursion step. The execution of such an algorithm can be seen as a branching tree. The best among the complete solutions at the leaves of the branching tree, is outputted. The basic idea of the technique is to branch on:

- a greedy extension of the partial solution.
- other extensions in the neighbourhood of the greedy extension.

The soundness of the algorithm lies on the fact that if none of the above extensions of the partial solution is done by a supposed optimal solution, then the greedy choice stays optimal at the end. All these terms and ideas are defined more precisely in Section 2.1 and the technique greediness-for-parameterization is applied with some refinements for MAX k-SET COVER in Section 2.1 and for MAX SAT-k in Section 3. Table 1 summarizes the main complexity results of the paper for both MAX k-SET COVER. Although the techniques are not the same, greediness-for-parameterization shares some common points with the greedy localization.
technique (see [10][12][20] for some applications). Here, one uses a local search approach: one starts from a computed approximate solution and turns it to an optimal solution. However, greedy localization technique is less general than greediness-for-parameterization, since it suits maximization problems only.

Then, in Section 2.2 we handle parameterized approximation of MAX $k$-SET COVER for parameters $k$ and $p$. We say that a minimization (maximization, respectively) problem $\Pi$, together with a parameter $k$, is parameterized $r$-approximable if there exists an FPT-time algorithm which computes a solution of size at most (at least, respectively) $rk$ whenever the input instance has a solution of size at most (at least, respectively) $k$; otherwise it returns any solution (smaller or greater than $rk$). This line of research was initiated by three independent works [14][7][11]. For an excellent overview, see [21]. For parameter $k$ we show a conditional result that is considered very likely, informally, a parameterized (with respect to $k$) approximation of MAX $k$-SET COVER within ratio greater than $1 - 1/e + \epsilon$, for some $\epsilon > 0$, would lead to a parameterized approximation of MIN SET COVER (with respect to the standard parameter) within ratio $(1 - \epsilon)\ln(n/\ln n)$, for some fixed $\epsilon > 0$. For parameter $p$, we show that MAX $k$-SET COVER can be solved within ratio greater than $1 - 1/e$ in parameterized time smaller than that needed for the exact solution of the problem.

While there are many hardness results and many membership results, in the parameterized complexity literature, the number of completeness results is not so high. Sometimes, a problem is shown to be hard for a class $\mathcal{K}$ of the parameterized hierarchy, and to belong to $\mathcal{K}'$, but the gap between $\mathcal{K}$ and $\mathcal{K}'$ can be important. The reader can find examples of that phenomenon in [8]. For instance, although as mentioned above, MAX $k$-SET COVER is W[2]-hard, the only inclusion that we are able to show for it, is in W[P] (see [8][13] for its definition). In fact (see also [8]) there exists about a dozen of problems that behave in the same way, i.e., that are W[2]-hard, in W[P]. We give in Section 3 one more such problem, that can be seen as a kind of canonical problem, called MAX SAT-$k$. Here, given a CNF on $n$ variables and $m$ clauses, one asks for setting to true at most $k$ variables satisfying at least $p$ clauses. Then, we further settle the parameterized complexity of MAX SAT-$k$. The above observation about the dichotomy between hardness and completeness in the W[·] world, naturally gives rise to the following questions: (i) “can one extend the definition of the classes of the W[·] hierarchy, in such a way that in the so obtained hierarchy (let us denote it by W'[·]), problems that are, say, W[1]-, or W[2]-hard (this is the
| Parameter | Complexity |
|-----------|------------|
| $\Delta + f$ | $k$ | $k + f$ | $k + \Delta$ & $p$ |
| $\notin \mathcal{XP}$ | $\mathcal{W}[2]$-hard, in $\mathcal{W}[P]$ | $\mathcal{W}[1]$-hard, in $\mathcal{W}[P]$ | FPT |

Table 1. Our main parameterized complexity results

In the case of most of natural problems that are not FPT, correspondingly belong to classes $\mathcal{W}[1]$ and $\mathcal{W}[2]$?

(i) “can one, using standard FPT-reducibility, or refining it, prove the existence of complete problems for this new hierarchy?”. In Section 4 we try to give a preliminary answer to the first of the questions above. We sketch a hierarchy of circuits, called counting weft hierarchy the classes of which are larger than the corresponding in the weft hierarchy and show that several problems that only belong to $\mathcal{W}[P]$, belong to the first two classes of the counting weft hierarchy. Anyway, it remains to exhibit complete problems for the classes of this new hierarchy.

In [17], the authors deal with such an issue. They introduce the classes $\mathcal{W}(\mathcal{F})$ which is defined similarly to the standard $\mathcal{W}$-hierarchy, except the gates are no more and, or, and not gates but any gates specified in $\mathcal{F}$. More precisely, they consider symmetric (i.e., the value of any output depends only on the number of inputs to 1) and bounded (i.e., the value of any output depends only on a constant number of its inputs) gates, and they show that if $\mathcal{F}$ contains only symmetric and bounded gates, the $\mathcal{W}(\mathcal{F})$-hierarchy coincide with the $\mathcal{W}$-hierarchy. As counting gates are not bounded, we can not use those results, at least directly.

2 Multiparameterizations for MAX $k$-SET COVER

We say that a set covers an element if the element is contained in the set.

**MAX $k$-SET COVER**

- **Input**: A set $\mathcal{S}$ of subsets of $X$, two integers $k$ and $p$.
- **Output**: $k$ subsets of $\mathcal{S}$ which covers at least $p$ elements of $X$.

2.1 Exact parameterization

First, we explain why $\text{MAX } k\text{-SET COVER}$ parameterized by $k + f$ is $\mathcal{W}[1]$-hard. Each instance $(\mathcal{S}, X)$ of $\text{MAX } k\text{-SET COVER}$ such that $f = 2$ (that is, each element appears in at most two sets) can be seen as a graph whose vertices are the sets in $\mathcal{S}$, and where there is an edge between two vertices
if the corresponding sets share at least one element. Therefore \( \text{max } k\text{-set cover} \) with frequency 2 is equivalent to the \( \text{max } k\text{-vertex cover} \) problem where the goal is to cover at least \( p \) edges with \( k \) vertices. Thus, \( \text{max } k\text{-vertex cover} \), \( W[1]\)-hard with respect to \( k \) [5], is a restricted case of \( \text{max } k\text{-set cover} \).

Note that in the reduction above the maximum set-cardinality \( \Delta \) in an instance of \( \text{max } k\text{-set cover} \) with \( f = 2 \), coincides with the maximum degree of the derived graph. Hence, with the same argument, it can be shown that \( \text{max } k\text{-set cover} \) is not in XP when parameterized by \( \Delta + f \) if \( P \neq NP \), since \( \text{max } k\text{-vertex cover} \) is NP-hard even in graphs with bounded degree.

Before we prove that \( \text{max } k\text{-set cover} \) is FPT with respect to \( k + \Delta \), let us introduce some vocabulary around branching algorithms. A partial solution is a subset of a (complete) solution. A branching algorithm is a recursive algorithm. Its execution on an instance \( I \) can be seen as a tree, called branching tree. In this tree, each node is labelled with a subinstance of \( I \) together with a partial solution, or more generally with some data maintained by the algorithm. The root is labelled with \( I \) and a leaf is a subinstance that causes the branching algorithm to stop. At a leaf, a complete solution is computed and returned. When identifying a node \( v \) to its label (a subinstance), the children of a subinstance are the subinstances which label the children of \( v \) in the branching tree. A node \( v \) of the branching tree is in accordance with a solution \( S \) if its corresponding partial solution is included in \( S \). A node \( v \) of the branching tree deviates from a solution \( S \) if it is in accordance with \( S \) but none of its children are in accordance with \( S \).

**Proposition 1.** \( \text{max } k\text{-set cover} \) parameterized by \( k + \Delta \) is FPT.

**Proof.** We present a branching algorithm (ALG1) whose particularity is to maintain, in addition to a partial solution \( T \) of subsets of \( S \), a subset \( C \subseteq X \) of elements which we commit to cover. The elements of \( C \) are therefore called the imposed elements. If \( R \) is a set of sets, \( \bigcup R \) denotes the union of the elements of \( R \). Now, a node \( v \) of the branching tree is in accordance with a solution \( T_0 \) if \( T \subseteq T_0 \) and \( C \subseteq \bigcup T_0 \). Let \( l \) be the labelling function of the nodes of the branching tree, such that for each node \( v \), \( l(v) = (T, C) \) where \( T \) and \( C \) are the sets described above. We can infer the subinstance \( I' \) at the node \( v \) from this labelling since \( I' = S \setminus T \).

To understand the bound over the number of imposed elements used in ALG1, note that if there exists a solution \( T \) covering more than \( p \) elements, then there exists a solution \( T' \subseteq T \) that covers more than \( p \) elements and
less than \( p + \Delta - 1 \) ones. Indeed, when no subset can be removed from a solution \( \mathcal{T}_0 \) (without going under \( p \) elements covered) then the number of covered elements can not exceed \( p + \Delta - 1 \), since removing a subset can uncover at most \( \Delta \) elements. For a set of subsets \( R \), we denote by \( S \downarrow R \) the set of subsets \( \{ S_i \setminus \bigcup R : S_i \in S\} \). We set \( T = \emptyset \) and \( C = \emptyset \). The overall specification of \( \text{ALG1} \) is the following:

- if \( |T| < k \) and \( |C| < p \) then: pick a set \( S_i \in S \setminus T \) that covers the largest number of elements in \( X \setminus C \); branch on \( \text{ALG1}(T \cup \{ S_i \}, C \cup S_i) \) and, for each element \( x \in S_i \setminus C \) branch on \( \text{ALG1}(T, C \cup \{ x \}) \);
- else: if \( |T| = k \), then store \( T \); else \((p \leq |C| \leq p + \Delta - 1)\) store a solution covering \( C \) (if possible);
- output the best among the solutions stored.

Let us first establish the time complexity of \( \text{ALG1} \). The number of children of a node of the branching tree is at most \( \Delta + 1 \). At each step, we add either a subset or an element, so the depth of the branching tree is, at most, \( k + p + \Delta - 1 \). Note that \( p \leq k\Delta \) on non-trivial \textsc{max} \( k \)-set cover-instances. So, the branching tree has size \( O((\Delta + 1)^{k+p}) \). On an internal node of the branching tree, \( \text{ALG1} \) only does polynomial computations. In a leaf of this tree, we find in time \( O^*(2^{p+\Delta-1}) \) if at most \( k - |T| \) subsets of \( (S \downarrow T)[C] \) can cover all the ground elements in \( C \) [18], where \( S[C] \) denotes the subsets in \( S \) entirely contained in \( C \). So, \( \text{ALG1} \) works in time \( O^*(2^{p+\Delta-1}(\Delta + 1)^{k+p}) \), i.e., it is fixed parameter with respect to \( k + \Delta \).

We now show that \( \text{ALG1} \) is sound. Let \( \mathcal{T}_0 \) be a solution which covers between \( p \) and \( p + \Delta - 1 \) elements. Recall that each node of the branching tree has one child adding a set to \( T \) and up to \( \Delta \) children each adding one imposed element to \( C \). Let \( \mathcal{B} \) be a maximal branch in the branching tree from the root to a node \( v \) such that all the nodes of \( \mathcal{B} \) are in accordance with \( \mathcal{T}_0 \). By the maximality of the branch, \( v \) deviates from \( \mathcal{T}_0 \). Let \( (T, C) = l(v) \) and \( S_i \) the set chosen by our greedy criterion at the node \( v \). We know that \( S_i \notin \mathcal{T}_0 \). Substituting in \( T \), any subset of \( \mathcal{T}_0 \setminus T \) by \( S_i \), we cover at least as many elements as \( \mathcal{T}_0 \), since \( \forall x \in S_i \setminus C, x \) is not covered by the solution \( \mathcal{T}_0 \). From \( v \), we consider again a maximal branch \( \mathcal{B}' \) in accordance with \( \mathcal{T}_0 \), and we iterate the same hybridization trick at most \( k + p \) times until we reach a leaf. At this leaf, \( \text{ALG1} \) computes an exact solution \( T_i \) containing at most \( k - |T| \) subsets of \( (S \downarrow T)[C] \). So, \( T \cup T_i \) is as good as \( \mathcal{T}_0 \), hence, an optimal solution. \( \square \)

Given that on non-trivial instances of \textsc{max} \( k \)-set cover, \( p > \Delta \) and \( p > k \), Proposition 1 derives an FPT algorithm with respect to the standard
parameter $p$ with running time $O^*(4^p p^{2p})$. However, this complexity is dominated by that of \cite{3}.

Another problem, very similar to \textsc{max k-set cover}, is the \textsc{max k-dominating set} problem. It consists, given some graph, of determining a set $V'$ of $k$ vertices that dominate at least $p$ vertices from $V \setminus V'$. Since \textsc{min dominating set} is known to be $W[2]$-hard, one can immediately derive that so is \textsc{max k-dominating set} with respect to $k$.

Let us recall the approximation preserving reduction from \textsc{min dominating set} to \textsc{min set cover} \cite{1}. This reduction works as follows: for each vertex $v$, build a set $S_v = N[v]$, where $N[v]$ is the closed neighbourhood of $v$. Then, covering $p$ elements in the so-built instance of \textsc{max k-set cover} is equivalent to dominating $p$ vertices in the instance of \textsc{max k-dominating set}. Finally, observe that this reduction preserves the values of $k$, $p$ and $\Delta$ (indeed, it transforms $\Delta$ to $\Delta + 1$). It is easy to see that this reduction is simultaneously an FPT reduction too and that identically works when dealing with \textsc{max k-dominating set} and \textsc{max k-set cover}. Then, one can conclude that \textsc{max k-dominating set} parameterized either by $k + \Delta$, or by $p$ is FPT, where for the \textsc{max k-dominating set}-instance, $\Delta$ is the maximum graph-degree. Note that these results about \textsc{max k-dominating set} have already been obtained by using the random separation technique \cite{6}. But the use of our technique (greediness-for-parameterization) is less costly in time.

As we have seen above \textsc{max k-set cover} is hard when parameterized by $k$, $k + f$ and $\Delta + f$. But, in which class of the $W[\cdot]$ hierarchy does this problem belong?

**Proposition 2.** \textsc{max k-set cover} parameterized by $k$ belongs to $W[P]$.

**Proof.** The proof is in exactly the same spirit with the proofs in \cite{9}. We reduce \textsc{max k-set cover} to \textsc{bounded non-deterministic Turing machine computation} which is a known $W[P]$-hard problem \cite{13} and defined as follows.

| **BOUNDED NON-DETERMINISTIC TURING MACHINE COMPUTATION** |
|--------------------------------------------------------|
| • **Input:** A nondeterministic Turing machine $M$, an input word $w$, an integer $n$ encoded in unary, a positive integer $k$ |
| • **Output:** Decide if $M(w)$ nondeterministically accept in at most $n$ steps and using at most $k$ nondeterministic steps. |

Let $I = (S = \{S_1, \ldots, S_m\}, p)$ be an instance of \textsc{max k-set cover}. Build a Turing Machine $M$ with three tapes $T_1$, $T_2$ and $T_3$. Tape $T_1$ is dedicated to non-deterministic guess. Write there the $k$ sets $S_{a_1}, \ldots, S_{a_k}$. Then, the
head of $T_1$ runs through all the elements and when a new element is found it is written down on the second tape. The third tape counts the number of already covered elements. If this number reaches $p$, then $M$ accepts. Thus, there exist $k$ non-deterministic steps, and a polynomial (in $|I|$) number of deterministic steps (precisely, $O(|I|^2)$).

By similar arguments, one can derive that $\text{MAX } k\text{-DOMINATING SET}$, parameterized by $k$ is also in $W[P]$.

### 2.2 Approximation issues

Let us now say some words about parameterized approximation of $\text{MAX } k\text{-SET COVER}$. Recall that as mentioned in the beginning of Section 1, $\text{MAX } k\text{-SET COVER}$ is inapproximable in polynomial time within ratio $1 - 1/e + \epsilon$, for any $\epsilon > 0$, unless $P = \text{NP}$ [16]. We mainly prove in the sequel that getting such a ratio even in time parameterized by $k$, is quite unlikely. More precisely, we prove that if this was possible, then we could get, in parameterized time, an approximation ratio of $(1 - \epsilon) \ln(n/\ln n)$ for $\text{MIN SET COVER}$, for some fixed $\epsilon > 0$, fact considered as highly unlikely. We also prove that approximating $\text{MAX } k\text{-SET COVER}$ within ratios better than $1 - \epsilon/n$ for any fixed constant $c > 1$, in time parameterized by $k$, is $W[2]$-hard. Note, finally, that $\text{MIN SET COVER}$ is inapproximable in polynomial time within ratio $(1 - \epsilon) \ln n$, for any $\epsilon > 0$, unless $P = \text{NP}$ [25].

**Proposition 3.** The following hold for $\text{MAX } k\text{-SET COVER}$:

1. unless $\text{MIN SET COVER}$ is approximable within ratio $(1 - \epsilon) \ln(n/\ln n)$, for some fixed $\epsilon > 0$, in time parameterized by the value of the optimum, $\text{MAX } k\text{-SET COVER}$ parameterized by $k$ is inapproximable within ratio $(1 - 1/e + \epsilon)$, for any $\epsilon > 0$;
2. unless $W[2] = \text{FPT}$, $\text{MAX } k\text{-SET COVER}$ is inapproximable within ratio $1 - \epsilon/n$, for any constant $c > 1$, in time parameterized by $k$.

**Proof.** The basic idea is similar to the idea in [16] (Proposition 5.2). Its key ingredient is the following. Consider some algorithm $k\text{SC-ALG}$ that solves $\text{MAX } k\text{-SET COVER}$. Then, it can iteratively be used to solve $\text{MIN SET COVER}$ as follows. Iteratively run $k\text{SC-ALG}$ for $k = 1, \ldots, m$ (where $m$ is the size of the set system in the instance of $\text{MIN SET COVER}$). One of these $k$’s will correspond to the value of the optimal solution for $\text{MIN SET COVER}$. Let us reason with respect to this value of $k$, denoted by $k_0$. Furthermore, assume $k\text{SC-ALG}$ achieves approximation ratio $r$ for $\text{MAX } k\text{-SET COVER}$. Call it with value $k_0$, (note that now $p = n$, the size of the
ground set), remove the ground elements covered, store the $k_0$ elements used and relaunch it with value $k_0$, until all ground elements are removed. Since it is assumed achieving approximation ratio $r$ after its $\ell$-th execution at most $(1 - r)^\ell n$ ground elements remain uncovered. Finally, suppose that after $t$ executions, all ground elements are removed (covered). Then, the $tk_0$ subsets stored form a $t$-approximate solution for the MIN SET COVER-instance, where $t$ satisfies (after some very simple algebra):

$$
(1 - r)^t n \leq 1 \implies t = \left\lceil \frac{-\ln n}{\ln(1 - r)} \right\rceil
$$

Taking $r = 1 - 1/e + \epsilon$, (1) leads to $t \leq \ln n / (1 - \ln(1 - \epsilon e)) \leq c' \ln n$, for some constant $c' < 1$, contradicting the inapproximability bound of [25], mentioned above proposition’s statement.

Moreover, observe that the complexity of the algorithm derived for MIN SET COVER is exactly the complexity of kSC-ALG, since any other operation as well as the number of its executions are polynomial in the size of the MIN SET COVER-instance.

Assume now that kSC-ALG is FPT in $k$ and that it achieves approximation ratio $1 - 1/e + \epsilon$ for some small $\epsilon$. Then, in order to prove item [1] follow the procedure described above until there are at most $\ln n$ uncovered elements, stop it and solve the remaining instance by, say, the best known exact algorithm which works within $O^*(2^n)$ in instances with ground set-size $n$ [2]. Since the surviving ground set has size $\ln n$, it is polynomial to optimally solve it. After some easy algebra, one gets:

$$
t \leq \frac{1}{1 + \epsilon e} \ln \left( \frac{n}{c' \ln n} \right)
$$

that concludes the proof of item [1].

For item [2] we will show how parameterized (with respect to $k$) approximability of MAX $k$-SET COVER within ratio $1 - c/n$, for some constant $c > 1$, would lead to parameterized approximability of MIN DOMINATING SET within additive approximation ratio $c + 1$.

Consider an instance $G$ of MIN DOMINATING SET and transform it to an instance $(S, X)$ of MIN SET COVER by the transformation seen above. Assume now that kSC-ALG achieves ratio $1 - c/n$ for some fixed $c > 1$. Then, just run kSC-ALG only once for every $k$. Assuming that kSC-ALG runs in time $O(p(n)F(k))$ for some polynomial $p$, the whole of runs will take $m \cdot O(p(n)F(k))$-time that remains FPT in $k$. For $k_0$, it holds that, after this run, at most $n - n(1 - (c/n)) = c$ elements will remain uncovered. Any (non-trivial) cover for them uses at most $c$ sets
to cover them. In this case, the procedure above achieves an additive approximation ratio $c + 1$ (recall that $c$ is fixed) for MIN SET COVER, and this ratio is identically transferred to MIN DOMINATING SET via the reduction. But for MIN DOMINATING SET, achievement of any constant additive approximation ratio is W[2]-hard [15]. □

For the rest of the section, we will relax the optimality requirement for the MAX k-SET COVER-solution and we will show that we can devise an approximation algorithm with ratio strictly better than $1 - 1/e$ in time parameterized by $k$ and $\Delta$, whose (parameterized) complexity is lower (although depending on the accuracy) than the best exact parameterized complexity for this problem.

Assume an FPT (exact) algorithm for MAX k-SET COVER with running time $O^*(F(k, \Delta))$ for some function $F$, denote it by fpt-ALG and consider the following algorithm, called pSC-IMPROVE in what follows:

1. fix some $\varepsilon > 0$, take $\mu$ such that $\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu$ and set $k' = \mu k$; run Algorithm fpt-ALG with parameter $k'$ and $\Delta$ and store the solution computed (denoted by $T_1$); let $X_1$ be the subset of $X$ covered by $T_1$;
2. set $S' = S \setminus T_1$, $X' = X \setminus X_1$ and $k'' = k - k' = (1 - \mu)k$; run the polynomial approximation MAX k-SET COVER-algorithm of [16] on the MAX k-SET COVER-instance $(S', X')$ (with parameter $k''$) and store the solution $T_2$ computed;
3. output $\hat{T} = T_1 \cup T_2$.

It is easy to see that solution $\hat{T}$ computed by pSC-IMPROVE has cardinality $k$, i.e., it is feasible for MAX k-SET COVER and the following holds (the proof is deferred in the appendix).

**Proposition 4.** For any $\mu > (e-2)/(e-1)$ and any $\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu$, MAX k-SET COVER can be approximated within ratio $1 - 1/e + \varepsilon$, in $O^*(F(\mu k, \Delta))$-time.

For instance, using Algorithm ALG1 for fpt-ALG, the complexity claimed by Proposition 4 becomes $O^*(2^{p+\Delta-1}(\Delta + 1)\mu k + p) \leq O^*(4^{p+p(1+\mu)})$.

### 3 Satisfiability problems

We introduce another quite natural satisfiability problem, called MAX SAT-k, and prove that it behaves like MAX k-SET COVER, from a parameterized point of view.
**Input:** A CNF-formula \( \phi \) on \( X \) a set of variables, two integers \( k \) and \( p \).

**Output:** A subset \( S \) of \( X \) of size at most \( k \), such that setting the variables of \( S \) to true and the variables of \( X \setminus S \) to false, satisfies at least \( p \) clauses.

In other words, given a CNF on \( n \) variables and \( m \) clauses, **MAX SAT-\( k \)** consists of setting at most \( k \) variables to true, satisfying at least \( p \) clauses.

**Proposition 5.** **MAX SAT-\( k \)** is \( W[2] \)-hard and in \( W[P] \) for parameter \( k \).

**Proof.** Setting \( p = m \), **MAX SAT-\( k \)** becomes **SAT-\( k \)** that is \( W[2] \)-hard \([22]\) (under the name **Weighted CNF-Satisfiability**). Proof of membership of **MAX SAT-\( k \)** in \( W[P] \) can be done by an easy reduction of this problem to **bounded non-deterministic Turing machine computation**. One can guess within \( k \) non-deterministic steps the variables to put to true and then one can check in polynomial time whether, or not, at least \( p \) clauses are satisfied. \( \Box \)

In what follows, \( C \) denotes the set of clauses of an instance and \( C' \) any subset of this set. We denote by \( \text{occ}^+(X_i, C') \) the number of positive occurrences of the variable \( X_i \) in the instance, and by \( \text{occ}^-(X_i, C') \) the number of its negative occurrences. We set \( f(X_i) = \text{occ}^+(X_i, C) + \text{occ}^-(X_i, C) \); so, the frequency of the formula is \( f = \max_i \{ \text{occ}^+(X_i, C) + \text{occ}^-(X_i, C) \} \).

**Lemma 1.** **MAX SAT-\( k \)** is solvable in \( O(2^m) \).

**Proof.** We take any variable \( X \) that appears positively and negatively. We do the standard branching: either set \( X \) to true (and decrease \( k \) by 1), either set \( X \) to false. This branching satisfies at least one more clause in each branch. Thus, it takes time bounded by \( O(2^m) \). The branching stops when each variable appears only negatively, or only positively. At that point, the variables appearing only negatively can be set to false. This step is safe since we are constrained to put at most (not exactly) \( k \) variables to true. We end up with an instance containing only positive literals. This instance can be seen as an instance of \( k \)-Hitting Set which can be solved in time \( O(2^m) \) \([18]\). Overall, it takes time \( O(2^m) \). \( \Box \)

**Proposition 6.** **MAX SAT-\( k \)** parameterized by \( p \) is FPT.
\textit{Proof.} We can assume that \( p < \frac{m}{2} \). Indeed, if \( p \geq \frac{m}{2} \), the algorithm of Lemma 1 is an FPT algorithm. We also assume that the number of clauses containing only negative literals is bounded above by \( \frac{m}{2} \). Otherwise, setting all the variables to false, satisfies more than \( p \) clauses. We recall that we are not forced to set exactly \( k \) variables to true, but at most \( k \). We observe that instances such that \( p < \frac{k}{2} \) are always YES-instances, since one can set one variable \( X_i \) with frequency \( f \) to true if \( \text{occ}^+(X_i, C) \geq \text{occ}^-(X_i, C) \), and to false otherwise. Note also that instances such that \( p < k \) are all YES-instances, too. Indeed, one can iteratively set to true \( k \) variables such that at each step one satisfies at least one more clause. If, at some point this is no longer possible, then setting all the remaining variables to false will satisfy all the clauses which do not initially contain only negative literals, that is at least half of the clauses, so more than \( p \) clauses. We may now assume that \( p \geq \frac{k}{2} \) and \( p \geq k \), so our parameter might as well be \( p + f + k \).

Once again, we construct a branching algorithm which operates accordingly to a greedy criterion. A solution is given as a set \( S \), of size up to \( k \), containing all the variables set to true. Additionally, we maintain a list \( C_s \) of clauses that we satisfy or commit to satisfy. Set \( d_i(C') = \text{occ}^+(X_i, C') \) and let \( C^+(X_i, C') \) be the set of clauses in \( C' \) where \( X_i \) appears positively and \( C^-(X_i, C') \) the set of clauses where \( X_i \) appears negatively. Set, finally, \( C(X_i, C') = C^+(X_i, C') \cup C^-(X_i, C') \) and consider the following algorithm (\textsc{Alg2}):

- set \( S = \emptyset \) and \( C_s = \emptyset \);
- if \( |S| < k \) and \( |C_s| < p \) then: set \( C_u = C \setminus C_s \) and let \( X_i \) be a variable maximizing \( d_i(C_u) \); branch on \textsc{Alg2}(\( S \cup \{X_i\}, C_s \cup C^+(X_i, C_u) \)), and for each clause \( c \in C(X_i, C_u) \), branch on \textsc{Alg2}(\( S, C_s \cup \{c\} \));
- else, if \( |S| = k \), then store \( S \) as solution; if \( |C_s| \geq p \), then check if the solution can be extended to satisfy each clause of \( C_s \);
- output the best among the solutions stored in the two previous steps.

The branching tree has depth at most \( k + p \) and width at most \( f + 1 \), so the running time of \textsc{Alg2} is \( O^*(2^p(f + 1)^{k+p}) \) that is FPT with respect to \( p \), because completing a solution to satisfy all the clauses of \( C_s \) can be done in time \( O^*(2^{|C_s|}) \) since \textsc{sat}-\( k \) can be solved in \( O(2^m) \) by Lemma 1.

Let now \( S_0 \) be an optimal solution. From the root of the branching tree, follow a maximal branch where the variables set to true are all in \( S_0 \), and the clauses in \( C_s \) are satisfied by \( S_0 \). Let \( S_c \) be the set of variables set to true along this branch (by definition, \( S_c \subseteq S_0 \)), and set \( S_n = S_0 \setminus S_c \). By maximality of the branch, at its extremity \( v \), \textsc{Alg2} deviates from \( S_0 \), i.e.,
no child of $v$ is in accordance with $S_0$. Let $X_i$ be the variable chosen at this point by ALG2 and consider $C_d = C(X_i, C_u)$ that is the set of clauses not yet in $C_s$ and where $X_i$ appears positively or negatively. We know that no clause in $C_d$ is satisfied by $S_0$. Let $X_j$ be any variable in $S_n$. We claim that $S_h = (S_0 \setminus \{X_j\}) \cup \{X_i\}$ is also optimal and, by a straightforward induction, one solution at the leaves of the branching tree is as good as $S_0$. Indeed, setting $X_j$ to false can lose at most $\text{occ}^+(X_j, C_u) - \text{occ}^-(X_j, C_u) \leq d_j(C_u)$ clauses and setting $X_i$ to true gains $d_i(C_u)$ clauses and, by construction, $d_i(C_u) \geq d_j(C_u)$.

The complexity of MAX SAT-$k$ parameterized by $k + f$ remains open.

4 Some preliminary thoughts about an enhanced weft hierarchy: the counting weft hierarchy

A natural way to generalize any problem $\Pi$ where one has to find a solution which universally satisfies a property is to define PARTIAL $\Pi$, where the solution only satisfies the property a “sufficient number of times”. In this sense, as mentioned MAX $k$-SET COVER where one has to cover at least $p$ elements, generalizes MIN SET COVER, where all the elements must be covered. Similarly MAX $k$-VERTEX COVER where one has to find a minimum subset of vertices which covers at least $p$ edges, generalizes MIN VERTEX COVER, where one has to cover all the edges; yet, MAX SAT generalizes SAT.

These partial problems come along with two parameters: the size of the solution, frequently denoted by $k$ and the “sufficient number of times” quantified by $p$. Many of these problems when parameterized by $k$ are shown to be either W[1]- or W[2]-hard, but we do not know how to prove a better membership result than the membership to W[P] (note that this is not the case of MAX $k$-VERTEX COVER, already proved to be W[1]-complete). This is a quite important asymmetry between classical complexity theory as we know it from the literature (see, for example, [19,23,24]) and parameterized complexity theory.

Showing the completeness of a W[1]- or a W[2]-hard problem, would imply that we can count up to $p$ with a circuit of constant height and weft 1 or 2. The “trick” of the input-vector weight permits to deal with cardinality constraint problems, but it is not suitable to problems, such as MAX $k$-SET COVER, where the value and the cardinality of the solution are constrained. We sketch, in what follows, a hierarchy of circuits named counting weft hierarchy whose classes are larger than their homologous
in the weft hierarchy. Basically, we generalize the \textit{and} gate to a \textit{counting} gate.

A counting gate $C_j$ with fan-in $i$ where $j \in \{0, \ldots, i\}$ has fan-out 1 and outputs 1 iff at least $j$ of its $i$ inputs are 1’s. Note that $C_i$ corresponds to an \textit{and} gate and $C_1$ is an \textit{or} gate. A \textit{counting} circuit is a circuit with some input gates, counting gates, negation gates, and exactly one output gate. Correspondingly, $\text{CW}[k]$ is the class of problems $\Pi$ parameterized by $p$ such that there is a constant $h$ and an FPT algorithm (in $p$) $A$, such that $A$ builds a counting circuit $C$ of constant height $h$ and weft $k$, and $I \in \Pi$ iff $C(I) = 1$. It can be immediately seen that the cweft hierarchy has exactly the same definition as the weft hierarchy up to replacing a circuit by a counting circuit.

Based upon the sketchy definition just above, the following can be proved by just taking the usual circuits for $\text{MIN SET COVER}$ and $\text{SAT}$ and replacing the corresponding large \textit{and} gates by gates $C_p$.

**Proposition 7.** The following inclusions hold for the counting weft hierarchy: $\text{MAX } k\text{-SET COVER and MAX SAT-}k$ are in $\text{CW}[2]$.

As mentioned in the introduction, the results of [17] which also focus on parallel W-hierarchy with other types of gates, can not be used here since the counting gates are symmetric but not bounded.

5 Conclusion

We have studied the parameterized complexity of $\text{MAX } k\text{-SET COVER}$ and $\text{MAX SAT-}k$ with respect to pairs of natural instance parameters as $k$, $\Delta$ (the maximum set-cardinality) and $f$, the maximum frequency of the ground elements, for the former, or the number of variables set to true in a feasible assignment and the maximum number of occurrences of the variables in the input-formula, for the latter. For this we have used the greediness-for-parameterization technique which is efficient for this kind of multiparameteraztion studies. We have introduced an enhancement of the classical weft hierarchy to try and prove completeness for those cardinality constrained $\text{W}[\cdot]$-hard problems. The existence of such complete problems for the new hierarchy is, in our opinion, the major open problem described in this paper.

References

1. R. Bar-Yehuda and S. Moran. On approximation problems related to the independent set and vertex cover problems. \textit{Discrete Appl. Math.}, 9:1–10, 1984.
2. A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.
3. M. Bläser. Computing small partial coverings. *Inform. Process. Lett.*, 85(6):327–331, 2003.
4. E. Bonnet, B. Escoffier, V. Th. Paschos, and E. Tourniaire. Multi-parameter complexity analysis for constrained size graph problems: using greediness for parameterization. *IPEC’13, LNCS* 8246, 66–77, 2013.
5. L. Cai. Parameter complexity of cardinality constrained optimization problems. *The Computer Journal*, 51:102–121, 2008.
6. L. Cai, S. M. Chan, and S. O. Chan. Random separation: a new method for solving fixed-cardinality optimization problems. *IWPEC’06, LNCS* 4169, 239–250, 2006.
7. L. Cai and X. Huang. Fixed-parameter approximation: conceptual framework and approximability results. *IWPEC’06, LNCS* 4169, 96–108, 2006.
8. M. Cesati. Compendium of parameterized problems. [http://www.sprg.uniroma2.it/home/cesati/](http://www.sprg.uniroma2.it/home/cesati/).
9. M. Cesati. The Turing way to parameterized complexity. *J. Comput. Syst. Sci.*, 67(4):654-685, 2003.
10. J. Chen, D. K. Friesen, W. Jia, and I. A. Kanj. Using nondeterminism to design deterministic algorithms. *FSTTCS’01, LNCS* 2245, 120–131, 2001.
11. Y. Chen, M. Grohe, and M. Grüber. On parameterized approximability. *IWPEC’06, LNCS* 4169, 109–120, 2006.
12. F. Dehne, M. R. Fellows, F. A. Rosamond, and P. Shaw. Greedy localization, iterative compression, modeled crown reductions: new FPT techniques, an improved algorithm for set splitting, and a novel 2k kernelization for vertex cover. *IWPEC’04, LNCS* 3162, 271–280, 2004.
13. R. G. Downey and M. R. Fellows. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer, 2013.
14. R. G. Downey, M. R. Fellows, and C. McCartin. Parameterized approximation problems. *IWPEC’06, LNCS* 4169, 121–129, 2006.
15. R. G. Downey, M. R. Fellows, C. McCartin, and F. A. Rosamond. Parameterized approximation of dominating set problems. *Inform. Process. Lett.*, 109(1):68–70, 2008.
16. U. Feige. A threshold of $\ln n$ for approximating set cover. *J. Assoc. Comput. Mach.*, 45:634–652, 1998.
17. M. R. Fellows, J. Flum, D. Hermelin, M. Müller and F. A. Rosamond. W-hierarchies Defined by Symmetric Gates. *Theory Comput. Syst.*, 46(2):311–339, 2010.
18. F. V. Fomin, D. Kratsch and G. J. Woeginger. Exact (exponential) algorithms for the dominating set problem. *WG’04, LNCS* 3353, 245–256, 2004.
19. M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.
20. Y. Liu, S. Lu, J. Chen, and S.-H. Sze. Greedy localization and color-coding: improved matching and packing algorithms. *IWPEC’06, LNCS* 4169, 84–95, 2006.
21. D. Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008.
22. R. Niedermeier. *Invitation to fixed-parameter algorithms*. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford, 2006.
23. C. H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
24. C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice Hall, New Jersey, 1981.

25. R. Raz and S. Safra. A sub-constant error probability low-degree test and a sub-constant error probability PCP characterization of NP. *STOC ’97*, pages 475–484, ACM, 1997.
Proof of Proposition 4

Fix an optimal solution \( T^* \) and denote by \( X^* \) the subset of \( X \) covered by \( T^* \). Obviously:

\[ |X_1| \geq \mu |X^*| \]  \hspace{1cm} (2)

Fix an optimal \textsc{max \( k'' \)-set cover} solution \( \bar{T} \) of \((S', X')\), denote by \( \bar{X}^* \) the set of elements of \( X \) covered by \( \bar{T} \) and set \( \bar{X} = X^* \cap X_1 \). Remark now the following facts:

1. \( X^* \setminus \bar{X} \) is covered by more than \( k'' \) sets in \( T^* \) (denote by \( T''' \) this system); otherwise, the sets of \( T^* \) covering \( X^* \setminus \bar{X} \) together with \( T_1 \) would be a solution better than \( T^* \);
2. the elements of \( X^* \setminus \bar{X} \) are still present in the instance \((S', X')\) where the approximation algorithm of [16] is called, as well as the subsets of \( S \) covering them, i.e., the sets of \( T''' \);
3. hence, the \( k'' \) “best” sets of \( T''' \) form a feasible solution for \textsc{max \( k'' \)-set cover} in \((S', X')\) covering more than \((k''/k)|X^* \setminus \bar{X}|\) elements of \( X^* \setminus \bar{X} \).

Combining Facts 1 and 3 and taking into account that \( T_2 \) is a \((1 - e^{-1})\)-approximation for \textsc{max \( k \)-set cover}, the following holds denoting by \( X_2 \) the subset of \( X \) covered by \( T_2 \):

\[
|X_2| \geq (1 - e^{-1}) \cdot |\bar{X}^*| \geq (1 - e^{-1}) \cdot \frac{k''}{k} \cdot \left( |X^* \setminus \bar{X}| \right) \\
= (1 - e^{-1}) \cdot \frac{k - k'}{k} \cdot \left( |X^* \setminus \bar{X}| \right) \\
= (1 - e^{-1}) (1 - \mu) \left( |X^* \setminus \bar{X}| \right) \\
\left| X^* \setminus \bar{X} \right| = |X^*| - \left| \bar{X} \right| \geq |X^*| - |X_1| \]  \hspace{1cm} (3)

\[
|X^* \setminus \bar{X}| = |X^*| - \left| \bar{X} \right| \geq |X^*| - |X_1| \]  \hspace{1cm} (4)

\footnote{In the sense that they cover the most of the elements covered by any other union of \( k'' \) sets of \( T''' \).}
Putting together (2), (3) and (4), we get the following for the approximation ratio of Algorithm \text{pSC-IMPROVE}:

\[
\frac{|X_1| + |X_2|}{|X^*|} \geq \frac{|X_1| + (1 - e^{-1})(1 - \mu)(|X^*| - |X_1|)}{|X^*|} \\
\geq \frac{(1 - e^{-1})(1 - \mu)|X^*| + [1 - (1 - e^{-1})(1 - \mu)]|X_1|}{|X^*|} \\
= \frac{|X^*|[(1 - e^{-1})(1 - \mu) + \mu[1 - (1 - e^{-1})(1 - \mu)]]}{|X^*|} \\
= (1 - e^{-1})(1 - \mu) + \mu[1 - (1 - e^{-1})(1 - \mu)] \\
= (1 - e^{-1}) - (1 - 2e^{-1})\mu + (1 - e^{-1})\mu^2
\]  

(5)

This ratio in (5) is at least \(1 - \frac{1}{e} + \varepsilon\), for any \(\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu\).

For the overall running time, it suffices to observe that, since the algorithm of [16] runs in polynomial time, the running time of \text{pSC-IMPROVE} is dominated by that of \text{fpt-ALG} called in step[1]. Thus, the whole complexity of Algorithm \text{pSC-IMPROVE} becomes \(O^*(F(\mu k, \Delta))\), as claimed.