Laboratoire d'Analyses et Modélisation de Systèmes pour l'Aide à la Décision
UMR 7243

CAHIER DU LAMSADE

341

Septembre 2013

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

E. Bonnet, V.Th. Paschos, F. Sikora

Parameterized Exact and Approximation Algorithms for Maximum k-Set Cover and Related Satisfiability Problems

Édouard Bonnet[1], Vangelis Th. Paschos[2], Florian Sikora[2]

[1] Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI)
bonnet.edouard@sztaki.mta.hu
[2] Université Paris-Dauphine, PSL Research University, CNRS, LAMSADE, Paris, France
{paschos,florian.sikora}@lamsade.dauphine.fr

Abstract. Given a family of subsets $\mathcal{S}$ over a set of elements X and two integers p and k, MAX k-SET COVER consists of finding a subfamily $\mathcal{T} \subseteq \mathcal{S}$ of cardinality at most k, covering at least p elements of X. This problem is W[2]-hard when parameterized by k, and FPT when parameterized by p. We investigate the parameterized approximability of the problem with respect to parameters k and p. Then, we show that MAX SAT-k, a satisfiability problem generalizing MAX k-SET COVER, is also FPT with respect to parameter p.

1 Introduction

In the MAX k-SET COVER problem, we are given a family of subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$, and two integers p and k. The goal is to find a subcollection $\mathcal{T}$ of at most k subsets that covers at least p elements. In what follows, we make the following two natural hypotheses for the instances of MAX k-SET COVER: (a) $S_i \neq S_j$, $i, j = 1, \ldots, m$ and (b) $S_i \not\subseteq S_j$, $i, j = 1, \ldots, m$.

MAX k-SET COVER is a well-known problem met in many real-world applications. To the best of our knowledge, it has been studied for the first time in the late seventies by Cornuejols et al. [13]. This combinatorial problem originated from a financial application, where one wishes to find an optimal location of bank accounts in order to maximize clearing time. Since then, it is used for modeling real problems met in several areas such as databases, social networks, sensor placement, information retrieval, etc. A non-exhaustive list of references to such applications can be found in Badanidiyuru and al. [2].

MAX k-VERTEX COVER, the graph version of MAX k-SET COVER is defined as follows: given a graph $G = (V, E)$ and two integers k and p, one wants to determine k vertices that cover at least p edges. MAX k-VERTEX COVER is a special case of MAX k-SET COVER where any element of X belongs to exactly two sets of $\mathcal{S}$.

Both MAX k-SET COVER and MAX k-VERTEX COVER are very important problems, since they are natural generalizations of MIN SET COVER and MIN VERTEX COVER, respectively. Both are NP-hard (setting $p = n$, MAX k-SET COVER becomes the seminal MIN SET COVER problem; setting $p = |E|$, MAX k-VERTEX COVER coincides with the MIN VERTEX COVER problem).

MAX k-SET COVER is known to be approximable within a factor $1 - 1/e$ (to our knowledge, it is the only polynomial approximation result known for MAX k-SET COVER on general instances) but, for any $\epsilon > 0$, no polynomial algorithm can approximate it within ratio $1 - 1/e + \epsilon$ unless P = NP [19], while MIN SET COVER is polynomially inapproximable within ratio $(1 - \epsilon) \ln n$, for arbitrarily small $\epsilon > 0$, unless P = NP [25]. On the other hand, MAX k-VERTEX COVER, is APX-hard and the best-known approximation ratio for this problem, in general graphs, is bounded below by $3/4$, obtained by a very smart linear programming method by Ageev et al. [1].

MAX SAT-k is a satisfiability problem closely related to MAX k-SET COVER. It is also a natural "fixed cardinality generalization" of MAX SAT. In MAX SAT-k, we are given a CNF on n variables and m clauses and we ask for setting to *true* at most k variables satisfying at least p clauses. One may observe that MAX SAT-k without negation is MAX k-SET COVER.

The goal of the paper is to establish several parameterized results for MAX k-SET COVER and for MAX SAT-k. We mainly study the parameterized approximability of the former and exact parameterization of the latter.

2 Preliminaries

We first give the basic definitions of the parameterized complexity theory. A parameterized problem (Π, k) is said *fixed-parameter tractable* (or in the class FPT) with respect to a parameter k if it can be solved by an algorithm with running time $f(k) \cdot |I|^{O(1)}$ time (in *fpt-time*), where f is some computable function and $|I|$ is the instance size. Such algorithms are called *fixed-parameter tractable algorithms*, or FPT algorithms. A parameterized reduction (or FPT reduction) from a problem Π_1 to a problem Π_2 is a mapping of an instance (I, k) of Π_1 to an instance (I', k') of Π_2, computable in time $f(k) \cdot |I|^{O(1)}$, such that $(I, k) \in \Pi_1 \Leftrightarrow (I', k') \in \Pi_2$, $k' \leqslant g(k)$, and $|I'| \leqslant h(k) \cdot |I|^{O(1)}$ for some computable functions f, g, and h. This seemingly technical definition is just tailored to ensure that if Π_2 is in FPT and there is an FPT reduction from Π_1 to Π_2, then Π_1 is also in FPT. Some problems such as CLIQUE parameterized by the solution size are not in FPT. In fact, there is a whole hierarchy of classes beyond FPT: FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq \cdots \subseteq$ W[P] $\subseteq$ XP. It is commonly believed that FPT $\neq$ W[1].

We need some additional definitions in order to give a precise meaning to those classes. A *boolean circuit* is a directed acyclic graph where every vertex of in-degree 0 is an *input vertex*, every vertex of in-degree 1 is a *negation vertex* and every vertex of in-degree greater than 2 is either an *and-vertex* or an *or-vertex*. Exactly one vertex with out-degree 0 is the *output vertex*. The *depth* of such a circuit is the maximum length of a path from an input vertex to the output vertex, and the *weft* of such circuit is the maximum number of *large* vertices on a path from an input vertex to the output vertex. A vertex is *large* if its in-degree exceeds some pre-agreed constant bound. Giving boolean values to the input vertices determines the value of every vertex in the classic way, and in particular, if the output vertex receive value true for a given assignment, we say that this assignment satisfies the circuit.

The WEIGHTED CIRCUIT SATISFIABILITY (WCS) problem takes as input a boolean circuit and an integer k and decides if there is a satisfying assignment for this circuit with exactly k input vertices set to true. A parameterized problem Π belongs to the class W[t], $t \geqslant 1$, if there is an FPT reduction from Π to WCS restricted to circuits of weft at most t. A parameterized problem Π is hard for the class W[t] (with $t \geqslant 1$) if, for any problem Π' in W[t], there is an FPT reduction from Π' to Π; or equivalently if there is an FPT reduction from WCS restricted to circuits of weft at most t to Π. A parameterized problem Π is W[t]-complete if it is W[t]-hard and in W[t]. For example, MAX INDEPENDENT SET parameterized by the size of the solution is W[1]-complete and MIN DOMINATING SET parameterized by the size of the solution is W[2]-complete. The class W[P] contains problems reducible to WCS without constraints on the weft of the circuits. The class XP contains problems solvable in time $|I|^{f(k)}$, where f is any computable function. See for example the monograph of Downey and Fellows for more details about fixed-parameter tractability [16].

We now go back to our problems. MAX k-SET COVER is W[2]-hard for the parameter k since, by setting $p = n$, we obtain an instance of MIN SET COVER which is W[2]-hard. An FPT algorithm with respect to the standard parameter p is given by Bläser [4]. Let us note that recently and independently certain aspects of parameterized complexity of MAX k-SET COVER have also been studied by Skowron and Faliszewski [29]. These results are summarized in Table 1.

Note that different parameterizations are possible for the same problem. In fact, one can also parameterize a problem by a combination of parameters (instead of just one parameter). A *multiparameterization* by parameters $k_1, k_2, \ldots, k_h$ consists of taking $k_1 + k_2 + \ldots + k_h$ as parameter. Multiparameterization poses novel and interesting open questions. For MAX k-SET COVER, for instance, several natural parameters as p (commonly called the "standard parameter"), k, $\Delta = \max_i\{|S_i|\}$ and $f = \max_i |\{j | x_i \in S_j\}|$ (commonly called the maximum frequency), can be jointly involved in a complexity study of the problem.

We first give multiparameterization of MAX k-SET COVER with respect parameters k and p (Section 3). The most important part of this section is Subsection 3.2 dedicated to the study

of the parameterized approximation of MAX k-SET COVER for these two parameters. Consider a problem Π, parameterized by some parameter π. Then, we say that Π is *parameterized r-approximable* if there exists an algorithm A that is FPT when parameterized by π such that:

- if Π is a minimization problem, then for any instance I of Π where $\pi \leqslant \beta$, A produces a solution with value at most $r\beta$; otherwise it returns any solution (which can be smaller or greater than $r\beta$);
- if Π is a maximization problem, then for any instance I of Π where $\pi \geqslant \beta$, A produces a solution with value at least $r\beta$; otherwise it returns any solution (which can be smaller or greater than $r\beta$).

This line of research was initiated by three independent works [17, 8, 11]. For an excellent overview, see the survey of Marx [24]. It aims at beating polynomial approximation barriers by offering more generous running time. The underlying question motivating Subsection 3.2 is *to what extent parameterized approximation is able to do it for* MAX k-SET COVER*?*.

Skowron and Faliszewski show, in [29], that it is possible when the parameter considered is $k + f$, where f is the frequency of the MAX k-SET COVER-instance, i.e., the maximum number of sets in $\mathcal{S}$, a ground element belongs to. But what happens when considering only k instead?

For parameter k we mainly show a conditional result, 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 $(1 - \varepsilon) \ln n$, for some fixed $\varepsilon > 0$. Even if this is a conditional result, we conjecture that the right answer is negative, i.e., that MAX k-SET COVER is inapproximable within ratio greater than $1 - 1/e + \epsilon$, for any $\epsilon > 0$, in FPT time parameterized by k. We also give in Subsection 3.2 a weaker negative result for MAX k-SET COVER, namely, that under the same parameterization, it is inapproximable within ratio $1 - (1/n)^{\sqrt[4]{\ln n}}$, unless ETH[3] fails.

Let us mention that MAX k-VERTEX COVER can be approximately solved within ratio $1 - \epsilon$, for any fixed $\epsilon > 0$, in FPT time parameterized by k [24].

For parameter p, we show that MAX k-SET COVER can be solved within ratio (strictly) greater than $1 - 1/e$ in time FPT parameterized by p, which is smaller than that needed for the exact solution of the problem.

In Section 4, we settle the parameterized complexity of MAX SAT-k, where, given a CNF on n variables and m clauses, one asks for setting to *true* at most k variables satisfying at least p clauses. The main result is that MAX SAT-k is FPT with respect to parameter p.

To prove that, we refine a technique for obtaining multiparameterized FPT algorithms developed by Bonnet et al. [5], 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 output. The basic idea of the technique is to branch on:

- a greedy extension of the partial solution;
- other extensions in the neighborhood 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. Although the techniques are not the same, greediness-for-parameterization shares some common points with the *greedy localization* technique (see [10, 15, 23] 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.

In Section 5, we suggest an enhanced weft hierarchy called *counting weft hierarchy* dedicated to those cardinality-constrained problems, such as MAX SAT-k and MAX k-SET COVER which are

[3] *Exponential Time Hypothesis*: there is a real number $\delta > 0$ such that 3-SAT is not solvable in $O(2^{\delta n})$ on instances with n variables.

W[i]-hard for some i, and in W[P], but not even known to be in W[j] for some integer j. Related issues have been discussed by Fellows et al. [20].

3 Parameterizations for MAX k-SET COVER

3.1 Exact Parameterization

As mentioned in Section 1, in the MAX k-SET COVER problem, we are given a family of subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$, and two integers p and k. The goal is to find a subcollection of $\mathcal{S}$ of size k that covers at least p elements of X.

Let us first note that, as $p \leqslant \Delta k$, the FPT result for MAX k-SET COVER with respect to parameter p presented by Bläser [4], immediately implies that this problem is also FPT when parameterized by $k + \Delta$. An alternative proof using greediness-for-parameterization is given by Bonnet et al. [6]. It might be worth reading it since it is a good introduction to the FPT algorithm for MAX SAT-k in Section 4.

We now explain why MAX k-SET COVER parameterized by $k + f$ is W[1]-hard. Each instance $(\mathcal{S}, X)$ of MAX k-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 MAX k-SET COVER with frequency 2 is equivalent to the MAX k-VERTEX COVER problem where, given a graph G and a number k, the goal is to cover at least p edges with k vertices. Thus, MAX k-VERTEX COVER, W[1]-hard with respect to k [7], is a restricted case of MAX k-SET COVER.

Note that in the reduction above the maximum set-cardinality Δ in an instance of MAX k-SET COVER with $f = 2$, coincides with the maximum degree of the derived graph. Hence, with the same argument, it can be shown that MAX k-SET COVER is not in XP when parameterized by $\Delta + f$, since MAX k-VERTEX COVER is NP-hard even in graphs with bounded degree (being, as mentioned above, a generalization of MIN VERTEX COVER that remains NP-hard even in these graphs).

Finally, in the following proposition, we prove that MAX k-SET COVER parameterized by k belongs to W[P].

Proposition 1. MAX k-SET COVER *parameterized by k belongs to W[P].*

Proof. The proof is in exactly the same spirit with the proof by Cesati [9]. We reduce MAX k-SET COVER to BOUNDED NON-DETERMINISTIC TURING MACHINE COMPUTATION which is a known W[P]-complete problem [16] and defined as follows. Given a non-deterministic Turing machine M, an input word w, an integer n encoded in unary and a positive integer k, does $M(w)$ non-deterministically accept in at most n steps and using at most k deterministic steps?

Let $\mathcal{I} = (\mathcal{S} = \{S_1, \ldots, S_m\}, p)$ be an instance of 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 $|\mathcal{I}|$) number of deterministic steps (precisely, $O(|\mathcal{I}|^2)$). □

The results mentioned in this paragraph as well as literature results are summarized in Table 1.

Parameter:	$\Delta + f$	k	$k + f$	$k + \Delta$ or p	$n - p$	$(n - p) + k$
Status:	$\notin$ XP	W[2]-hard in W[P] (Prop. 1)	W[1]-hard in W[P]	FPT [4, 6]	$\notin$ XP [29]	W[2]-complete [29]

Table 1. Exact parameterized complexity of MAX k-SET COVER for different parameters.

3.2 Approximation Issues

Let us now handle parameterized approximation of MAX k-SET COVER. We first prove the following basic lemma that is an easy generalization of Proposition 5.2 given by Feige [19].

Lemma 2. *Any r-approximation algorithm, parameterized by k, for* MAX k-SET COVER *can be transformed into an FPT t-approximation algorithm, parameterized by the standard parameter (optimum value), for* MIN SET COVER *where:*

$$t < \left\lceil \frac{-\ln n}{\ln(1-r)} \right\rceil$$

Proof. The basic idea is similar to the idea of Feige [19] (Proposition 5.2). Its key ingredient is the following. Consider some algorithm `kSC-ALG` that solves MAX k-SET COVER. Then, it can iteratively be used to solve MIN SET COVER as follows. Consider an instance $I = (\mathcal{F}, U)$ of MIN SET COVER where $\mathcal{F}$ is a family of subsets of a ground set U. Iteratively run `kSC-ALG` for $k = 1, \ldots, m$ (where m is the size of $\mathcal{F}$). Eventually, one value of k will equal the value of the optimal solution for MIN SET COVER. Let us reason with respect to this value of k, denoted by k_0. Furthermore, assume that `kSC-ALG` achieves approximation ratio r for MAX k-SET COVER. Invoke it with value k_0, (note that now $p = n$, the size of the ground set U), remove the ground elements covered, store the k_0 elements used and relaunch it with value k_0, until all ground elements of U are removed. Since it is assumed to achieve approximation ratio r after its ℓ-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):

$$(1-r)^t n < 1 \Longrightarrow t < \left\lceil \frac{-\ln n}{\ln(1-r)} \right\rceil \tag{1}$$

Moreover, observe that the complexity of the algorithm derived for MIN SET COVER is at most m times the complexity of `kSC-ALG`; so, it remains FPT with respect to the optimal value for MIN SET COVER . □

Recall that, as mentioned in the beginning of Section 1, MAX k-SET COVER is inapproximable in polynomial time within ratio $1 - 1/e + \epsilon$, for any $\epsilon > 0$, unless P = NP [19]. We first prove in the sequel a conditional result, informally, getting such a ratio even in FPT time parameterized by k, is a rather difficult task. More precisely, we prove that if this were possible, then we could get, in parameterized time, an approximation ratio of $(1-\epsilon)\ln(n/\ln n)$ for MIN SET COVER, for some fixed $\epsilon > 0$. Even if this result is, properly speaking, a conditional result, it gives, in some sense, the measure of the difficulty of approximating MAX k-SET COVER within ratio strictly better than $1 - 1/e$ in FPT time parameterized by k. Next, we prove an FPT inapproximability result, namely that FPT approximation of MAX k-SET COVER within ratio greater than $1 - (1/n)^{\sqrt[4]{\ln k}}$ is impossible unless W[2] = FPT.

Proposition 3. MAX k-SET COVER *parameterized by k is inapproximable within ratio $(1-1/e+\epsilon)$, for any $\epsilon \in [0, 1/e)$, unless* MIN SET COVER *is approximable within ratio $(1-\eta)\ln(n/\ln n)$, for some fixed $\eta > 0$, in FPT time parameterized by the value of the optimum.*

Proof. Revisit Lemma 2, take $r = 1 - (1/e) + \epsilon$, for some $\epsilon \in [0, 1/e)$ and assume that the `kSC-ALG` of Lemma 2 (that is FPT in k) achieves approximation ratio r. Then, in order to prove the result claimed, follow the procedure described in Lemma 2 until there are at most $\ln n$, say $c \ln n$ for some $c \leqslant 1$, uncovered elements in U 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 [3]. Since the surviving ground set has size $c \ln n$, it is polynomial to optimally solve it. Reasoning exactly as in Lemma 2 we get:

$$n(1-r)^t = c\ln n \Longrightarrow t = \frac{\ln c + \ln\ln n - \ln n}{\ln(1-r)} \leqslant \frac{\ln\ln n - \ln n}{\ln(1-r)} \simeq \frac{\ln\ln n - \ln n}{-\epsilon e - 1}$$
$$= \frac{\ln n - \ln\ln n}{1+\epsilon e} = \frac{1}{1+\epsilon e}\ln\left(\frac{n}{\ln n}\right)$$

Setting $\eta = {}^{\epsilon e}/_{(1+\epsilon e)}$, the proof of the proposition is concluded. □

As one can see, the result of Proposition 3 is conditional and relates the parameterized approximability of MAX k-SET COVER within ratios better than the one achieved in polynomial time to the parameterized approximability of MIN SET COVER within ratios that are almost the same (in fact slightly smaller) as the one polynomially achieved for this problem. Furthermore this ratio is tight for the polynomial time (recall that it is NP-hard to approximate MIN SET COVER within ratio $(1-\varepsilon)\ln n$, for arbitrarily small $\varepsilon > 0$ [25]). We conjecture that the real parameterized (with respect to the optimum) inapproximability bound of MIN SET COVER is $O(\log n)$, so that the inapproximability bound (conditionally) conjectured by Proposition 3 is the correct one. But, unfortunately, we have not been able to prove it until now and the negative result by Moshkovitz in [25] does not seem to be usable as it is for the parameterized inapproximability of MAX k-SET COVER.

In what follows, in the spirit of Lemma 2 and of Proposition 3 and based upon a recent result by Chen and Lin [12], we show a weaker upper bound for the parameterized approximability of MAX k-SET COVER with respect to k.

Consider the MIN DOMINATING SET problem defined as follows: *given a graph $G = (V, E)$, determine a minimum size vertex subset $D \subseteq V$ such that every vertex of V is either in D or has a neighbor in D*. There exists a well-known approximability preserving reduction from MIN DOMINATING SET to MIN SET COVER that works as follows: given a graph $G = (V, E)$ of order n, instance of MIN DOMINATING SET, we transform it into an instance $I = (\mathcal{F}, U)$ of MIN SET COVER as follows:

- $\mathcal{F} = \{F_1, \ldots, F_n\}$;
- $U = \{u_1, \ldots, u_n\}$;
- $\forall i \in \{1, \ldots, n\}$, $F_i = \{u_j : v_j \in \Gamma[v_i]\}$, where $\Gamma[v_i]$ denotes the closed neighborhood of vertex $v_i \in V$.

Then, it is easy to see that any dominating set D of G corresponds to a set cover $\mathcal{F}'$ of I of the same cardinality by simply considering in $\mathcal{F}'$ the subsets of $\mathcal{F}$ having the same indices with the vertices of D and vice-versa. An immediate consequence of this reduction is that both problems share the same approximation ratios and inapproximability bounds.

Recently, Chen and Lin [12] have proved that, *under ETH, no FPT algorithm for* MIN DOMINATING SET *can achieve approximation ratio smaller than, or equal to, $\sqrt[4+\epsilon]{\ln \gamma(G)}$, for any positive constant ϵ, where $\gamma(G)$ denotes the cardinality of a minimum dominating set in G*. The reduction just described, immediately transfers this lower bound to MIN SET COVER; so the following holds for this latter problem: *under ETH, no FPT algorithm for* MIN SET COVER *can achieve approximation ratio smaller than, or equal to, $\sqrt[4+\epsilon]{\ln k_0}$, for any positive constant ϵ, where k_0 denotes the cardinality of a minimum set cover instance $(\mathcal{F}, U)$*.

Based upon the result by Chen and Lin [12], Lemma 2, and the approximation-preserving reduction from MIN DOMINATING SET to MAX k-SET COVER given just above, the following can be proved.

Proposition 4. MAX k-SET COVER *is inapproximable within ratio $1 - ({}^1/_n)^{\sqrt[4]{\ln k}}$ in FPT time parameterized by k, unless ETH fails.*

Proof. The proof follows the one of Lemma 2. From (1), taking for t (the ratio of MIN SET COVER) the $\sqrt[4]{\ln k_0}$-inapproximability bound derived by the the reduction above and by Chen and Lin in [12] and omitting (in order to simplify calculations) the ceiling in (1), some elementary algebra leads to:

$$\ln(1-r) \leqslant \frac{-\ln n}{t} \leqslant \frac{-\ln n}{\sqrt[4]{\ln k_0}} \Longrightarrow 1 - r \geqslant \left(\frac{1}{n}\right)^{\sqrt[4]{\ln K_0}} \Longrightarrow r \leqslant 1 - \left(\frac{1}{n}\right)^{\sqrt[4]{\ln k_0}}$$

as claimed. □

Let us note that a parameterized inapproximability bound weaker than that of Proposition 4 can be obtained as follows. Consider an instance G of MIN DOMINATING SET and transform it to an instance $I = (\mathcal{F}, U)$ 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 (the value of the optimal solution for MIN SET COVER in instance I), 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 error $c + 1$ (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 error is W[2]-hard [18]. So, the following corollary holds.

Corollary 5. MAX k-SET COVER *is inapproximable within ratio* $1 - c/n$ *in time parameterized by* k*, for any constant* $c > 1$*, unless W[2] = FPT.*

For the rest of the section, we relax the optimality requirement for the MAX k-SET COVER-solution and we show that we can devise an approximation algorithm with ratio strictly better than $1 - 1/e$ (beating so the polynomial inapproximability bound of Feige [19]), that runs in FPT time parameterized by k and Δ but whose complexity is lower (although depending on the accuracy) than the best exact parameterized complexity for MAX k-SET COVER.

In fact, we are going to prove a stronger result, claiming that, for any polynomial approximation ratio r achieved by some polynomial time algorithm `APPROX` and any parameterized algorithm `PARAM`(π) for MAX k-SET COVER, where π can be a single parameter or a vector of parameters, r can be improved in FPT time parameterized by π by an algorithm whose running time is smaller than the one of `PARAM`(π).

The way of doing it is to built a kind of hybrid algorithm that, informally, in the case of MAX k-SET COVER we deal with, where $\pi = (k, \Delta)$, it works as follows:

- it solves MAX k-SET COVER by invoking `PARAM`(π'), with $\pi' = (k', \Delta)$ for some $k' < k$, stores the solution computed, and removes it from the initial instance;
- it invokes `APPROX` on the surviving instance and stores the solution obtained;
- it takes the union of the two solutions.

Here, our objective is to establish a trade-off between the solution quality and the running time. Therefore, the running time of the presented algorithms depend on the approximation ratio.

Assume an FPT (exact) algorithm running in time $O^*(F(k, \Delta))$ for some function F (that is Algorithm `PARAM`(k, Δ)) together with an approximation algorithm `APPROX` achieving approximation ratio r for MAX k-SET COVER and consider Algorithm 1, called `pSC-IMPROVED` in what follows, running on an instance $(\mathcal{S}, X)$ of MAX k-SET COVER, where X is the ground set and $\mathcal{S}$ a family of subsets of X.

Algorithm 1: A description of the algorithm `pSC-IMPROVED`.

Input: An instance $(\mathcal{S}, X, k)$ of MAX k-SET COVER.
Output: A subfamily $\hat{\mathcal{T}} \subset \mathcal{S}$ containing k subsets.
fix some $\varepsilon > 0$;
take $\mu \in (0, 1)$ such that $\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu$;
set $k' = \mu k$;
run Algorithm `PARAM`(k', Δ) and store the solution computed (denoted by $\mathcal{T}_1$); let X_1 be the subset of X covered by $\mathcal{T}_1$;
set $\mathcal{S}' = \mathcal{S} \setminus \mathcal{T}_1$, $X' = X \setminus X_1$ and $k'' = k - k' = (1 - \mu)k$;
run the r-approximation algorithm `APPROX` on the MAX k-SET COVER-instance $(\mathcal{S}', X', k'')$ and store the solution $\mathcal{T}_2$ computed;
return $\hat{\mathcal{T}} = \mathcal{T}_1 \cup \mathcal{T}_2$;

Solution $\hat{\mathcal{T}}$ computed by pSC-IMPROVED has cardinality k, i.e., it is feasible for MAX k-SET COVER. Let us now analyze it in the following proposition.

Proposition 6. *Given a polynomial time approximation algorithm with ratio r for* MAX k-SET COVER*, for any $\mu \in (2-(1/r), 1]$,* MAX k-SET COVER *can be approximated within ratio $r(1-\mu)^2+\mu > r$, in $O^*(F(\mu k, \Delta))$-time, where Δ is the maximum cardinality of a set.*

Proof. Fix an optimal solution $\mathcal{T}^*$ and denote by X^* the subset of X covered by $\mathcal{T}^*$. Recall that in Algorithm 1, we denote by X_1 the subset of X covered by the set $\mathcal{T}_1$ computed by PARAM(k', Δ). Obviously:

$$|X_1| \geqslant \mu\,|X^*| \tag{2}$$

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

1. $X^* \setminus \tilde{X}$ is covered with at least k'' sets in $\mathcal{T}^*$ (denote by $\mathcal{T}''^*$ this system); otherwise, the sets of $\mathcal{T}^*$ covering $X^* \setminus \tilde{X}$ together with $\mathcal{T}_1$ would be a solution better than $\mathcal{T}^*$; indeed, if the elements of $X^* \setminus \tilde{X}$ were covered with less than k'' sets, i.e., if $|\mathcal{T}''^*| < k''$, then, since $\tilde{X} \subseteq X_1$, $\mathcal{T}_1 \cup \mathcal{T}''^*$ would be a set system of size $|\mathcal{T}_1 \cup \mathcal{T}''^*| < k$ covering $|X^*|$ elements; in this case, completing (even greedily) the family $\mathcal{T}_1 \cup \mathcal{T}''^*$ with $k - |\mathcal{T}_1 \cup \mathcal{T}''^*|$ sets would lead to a k-sets subfamily of $\mathcal{S}$ covering more than $|X^*|$ ground elements, absurd since X^* is the value of an optimal solution for MAX k-SET COVER;
2. the elements of $X^* \setminus \tilde{X}$ are still present in the instance $(\mathcal{S}', X')$ where the r-approximation algorithm APPROX is invoked, as well as the subsets of $\mathcal{S}$ covering them, i.e., the sets of $\mathcal{T}''^*$;
3. hence, the k'' "best"[4] sets of $\mathcal{T}''^*$ form a feasible solution for MAX k''-SET COVER in $(\mathcal{S}', X')$ and cover more than $(k''/k)|X^* \setminus \tilde{X}|$ elements of $X^* \setminus \tilde{X}$.

Combining Facts 1, 2 and 3 and taking into account that $\mathcal{T}_2$ is an r-approximation for MAX k-SET COVER, the following holds denoting by X_2 the subset of X covered by $\mathcal{T}_2$:

$$\begin{aligned} |X_2| &\geqslant r \cdot \left|\bar{X}^*\right| \geqslant r \cdot \frac{k''}{k} \cdot \left(\left|X^* \setminus \tilde{X}\right|\right) = r \cdot \frac{k-k'}{k} \cdot \left(\left|X^* \setminus \tilde{X}\right|\right) \\ &= r \cdot (1-\mu) \left(\left|X^* \setminus \tilde{X}\right|\right) \end{aligned} \tag{3}$$

$$\left|X^* \setminus \tilde{X}\right| = |X^*| - \left|\tilde{X}\right| \geqslant |X^*| - |X_1| \tag{4}$$

Putting together (2), (3) and (4), we get the following for the approximation ratio of Algorithm pSC-IMPROVED:

$$\begin{aligned} \frac{|X_1| + |X_2|}{|X^*|} &\geqslant \frac{|X_1| + r \cdot (1-\mu) \cdot (|X^*| - |X_1|)}{|X^*|} \\ &\geqslant \frac{r \cdot (1-\mu) \cdot |X^*| + [1 - r \cdot (1-\mu)] \cdot |X_1|}{|X^*|} \\ &\geqslant \frac{|X^*| \cdot [r \cdot (1-\mu) + \mu \cdot [1 - r \cdot (1-\mu)]]}{|X^*|} \\ &= r \cdot (1-\mu) + \mu \cdot [1 - r \cdot (1-\mu)] = r(1-\mu)^2 + \mu \end{aligned} \tag{5}$$

Ratio in (5) is at least r, for any $\mu \in ((2 - (1/r)), 1]$.

For the overall running time, it suffices to observe that, since APPROX runs in polynomial time, the running time of pSC-IMPROVED is dominated by that of PARAM invoked within Algorithm 1. Thus, the whole complexity of Algorithm pSC-IMPROVED becomes $O^*(F(\mu k, \Delta))$, as claimed. □

[4] In the sense that they cover the most of the elements covered by any other union of k'' sets of $\mathcal{T}''^*$.

Revisit now the proof of Proposition 6 and set $r = 1 - e^{-1}$. Then, (5) becomes:

$$\begin{aligned}
\frac{|X_1| + |X_2|}{|X^*|} &\geqslant \frac{|X_1| + \left(1 - e^{-1}\right) \cdot (1 - \mu) \cdot (|X^*| - |X_1|)}{|X^*|} \\
&\geqslant \frac{\left(1 - e^{-1}\right) \cdot (1 - \mu) \cdot |X^*| + \left[1 - \left(1 - e^{-1}\right) \cdot (1 - \mu)\right] \cdot |X_1|}{|X^*|} \\
&\geqslant \frac{|X^*| \cdot \left[\left(1 - e^{-1}\right) \cdot (1 - \mu) + \mu \cdot \left[1 - \left(1 - e^{-1}\right) \cdot (1 - \mu)\right]\right]}{|X^*|} \\
&= \left(1 - e^{-1}\right) \cdot (1 - \mu) + \mu \cdot \left[1 - \left(1 - e^{-1}\right) \cdot (1 - \mu)\right] \\
&= \left(1 - e^{-1}\right) - \left(1 - 2e^{-1}\right) \cdot \mu + \left(1 - e^{-1}\right) \cdot \mu^2
\end{aligned}$$

This ratio is at least $1 - 1/e + \varepsilon$, for any $\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu$, and the following corollary holds.

Corollary 7. *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, let us take $\mu = 0.5 \geqslant (e-2)/(e-1)$. Then application of Corollary 7 leads, after some easy algebra to an approximation ratio at least $0.75 + (0.25/e) \geqslant 0.841$ achieved with complexity $F(k/2, \Delta)$. Even if, in the case of MAX k-SET COVER, the potential of the running time seems to be quite small, we think that looking for this type of trade-offs between approximation ratios and running times in order to try to beat polynomial approximation barriers is an interesting research program.

In [14] an analogous result is given. There, the authors try to beat the APX-hardness of MAX k-VERTEX COVER by a moderately exponential approximation algorithm, i.e., by an approximation algorithm achieving ratio $1 - \epsilon$, for any $\epsilon > 0$, and running in time that, although exponential, remains smaller than the running time of the best known exact algorithm for this problem. More precisely, the following is proved by Della Croce and Paschos [14]. *If $T(n, k)$ is the running time of an exact algorithm and ρ the approximation ratio of some polynomial approximation algorithm for* MAX k-VERTEX COVER*, then, for any $\epsilon > 0$,* MAX k-VERTEX COVER *can be approximated within ratio $1 - \epsilon$ with worst-case running time $T(n, [(2\rho - 1) + \sqrt{1 - 4\epsilon\rho}]k/2\rho)$.*

4 MAX SAT-k

We now study a generalization of MAX k-SET COVER, the MAX SAT-k problem. Recall that in MAX SAT-k, given a CNF formula on n variables and m clauses, the objective is to satisfy at least p clauses, by setting at most k variables to *true*.

Proposition 8. MAX SAT-k *parameterized by k is W[2]-hard and in W[P].*

Proof. Setting $p = m$, MAX SAT-k becomes SAT-k that is W[2]-hard [26] (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. □

Consider an instance ϕ of MAX SAT-k on a set of C clauses. Given any subset C' of C, we denote by $\mathrm{occ}^+(x_i, C')$ the number of positive occurrences of the variable x_i in C', and by $\mathrm{occ}^-(x_i, C')$ the number of its negative occurrences. We set $f(x_i) := \mathrm{occ}^+(x_i, C) + \mathrm{occ}^-(x_i, C)$; so, the frequency of the formula is $f := \max_i f(x_i)$.

Before proving that MAX SAT-k is FPT with respect to parameter p, let us introduce some vocabulary on 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 labeled with a sub-instance of I together

with a partial solution, or more generally with some data maintained by the algorithm. The *root* is labeled with I and a *leaf* is a sub-instance 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 sub-instance), the *children* of a sub-instance are the subinstances which label the children of v in the branching tree.

We now prove this simple lemma.

Lemma 9. MAX SAT-k *is solvable in time* $O^*(2^m)$.

Proof. We take any variable x that appears positively in at least one clause *and* negatively in at least one clause. We do the standard branching: either set x to *true* (and decrease k by 1), or set x to *false*. This branching satisfies at least one more clause in each branch. Therefore, the branching tree is a subtree of the full binary tree with 2^m leaves. At a leaf ℓ of the branching tree, the remaining number of clauses is at most $m - \lambda$ where λ is the depth of ℓ. The branching stops when each variable appears only negatively, or only positively. At this 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. Therefore, at the leaves, the instances can now be seen as instances of MAX k-HITTING SET with at most $m - \lambda$ sets; or equivalently, MAX k-SET COVER with at most $m - \lambda$ elements which can be solved by standard dynamic programming in time $O^*(2^{m-\lambda})$. So, the overall running time is $O^*(2^m)$. □

We identify a solution of an instance of MAX SAT-k to the set S of variables put to *true*. It induces a set C' of satisfied clauses. The algorithm solving MAX SAT-k we will present, performs two kinds of choices: (1) setting a variable to *true* and (2) putting a clause which is not satisfied yet into a set C_s of clauses that should eventually be satisfied. Putting a clause in C_s means that we commit to satisfy it later. At any node of the branching tree, a child corresponds to either performing choice (1) for some given variable, or choice (2) for some given clause. A choice (1) is *in accordance* with S if it sets to *true* a variable in S. A choice (2) is *in accordance* with S if it puts in C_S a clause satisfied by solution S (i.e., this clause is in C'). A node v of the branching tree is *in accordance* with S if all the choices made from the root to this node are in accordance with 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.

Let us give a toy example to clarify those notions. Assume a solution which sets x_2 and x_3 to *true* and all the other variables to *false*. So, $S = \{x_2, x_3\}$. And, this assignment satisfies the following set of clauses: $\{c_2, c_4, c_5, c_6, c_8, c_9\}$. Say, the root of the branching tree has 4 children: setting x_1 to *true*, committing to satisfy c_1, committing to satisfy c_2, and committing to satisfy c_6. The two first children are not in accordance with S, but the two last are. Indeed, c_2 and c_6 are satisfied by S. Let us move to the child where we commit to satisfy c_2. Suppose this node has three children: setting x_2 to *true*, committing to satisfy c_1, committing to satisfy c_3. We now move to the child where we set x_2 to *true*. So far, we have only done choices in accordance with S, so our current node is in accordance with S. Now, say, this new node v has three children: setting x_1 to *true*, committing to satisfy c_3, committing to satisfy c_7. None of those choices is in accordance with S, so v deviates from S.

Proposition 10. MAX SAT-k *parameterized by* p *is FPT.*

Proof. Let (C, k, p) be an instance of MAX SAT-k where C is a set of clauses over a set of variables $X = \{x_1, x_2, \ldots, x_n\}$, k is the maximum number of variables that can be set to true, and p the minimum number of clauses to satisfy. We can assume that $p < m/2$. Indeed, if $p \geqslant m/2$, the algorithm of Lemma 9 is an FPT algorithm. We also assume that the number of clauses containing only negative literals is bounded above by $m/2$. 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 $p < f/2$ are always YES-instances, since one can set one variable x_i with frequency f to *true* if $\mathrm{occ}^+(x_i, C) \geqslant \mathrm{occ}^-(x_i, C)$, and to *false*

otherwise, and set all the other variables to false. This assignment does indeed satisfy at least $\max(\text{occ}^+(x_i, C), \text{occ}^-(x_i, C)) \geqslant f/2 < p$ clauses.

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 $p \geqslant f/2$ and $p \geqslant k$, so our parameter might as well be $p + f + k$.

We construct a branching algorithm which operates accordingly to a greedy criterion. A solution, or *complete assignment*, is given by a set S of size up to k which contains all the variables set to *true*. Additionally, we maintain a list C_s of clauses that we satisfy or commit to satisfy. For notational convenience we define $C_u := C \setminus C_s$, $d_i(C') := \text{occ}^+(x_i, C')$, 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. Finally, $C(x_i, C') := C^+(x_i, C') \cup C^-(x_i, C')$. Algorithm `pSAT-k` (see Algorithm 2) is fairly simple. We find the variable x that, if set to *true*, would satisfy the maximum number of clauses among the still unsatisfied clauses. We branch on setting x to *true* (choice (1)) or for each still unsatisfied clause c that x would satisfy, on putting c in C_s (choice (2)).

Algorithm 2: A description of the algorithm `pSAT-k`.

Input: A set C of clauses on a set X of variables.
Output: A subset $S \subseteq X$ of size at most k such that setting all the variables in S to *true* and all the variables in $X \setminus S$ to *false*, satisfies the greatest number of clauses in C.

```
set S = ∅, C_s = ∅;
ALG1(S, C_s):
if |S| < k and |C_s| < p then
    pick the variable x_i maximizing d_i(C \ C_s);
    run ALG1(S ∪ {x_i}, C_s ∪ C^+(x_i, C \ C_s));
    for each clause c ∈ C(x_i, C \ C_s) do
        run ALG1(S, C_s ∪ {c});
else
    if |S| = k then
        store S;
    else
        (|C_s| ⩾ p) store a complete assignment satisfying C_s, if possible;
return the best among the solutions stored;
```

The branching tree has depth at most $k + p$ and arity at most $f + 1$, so the running time of `pSAT-k` is $O^*(2^p(f+1)^{k+p}) = O^*(p^{O(p)})$, that is FPT with respect to parameter p, because completing a solution to satisfy all the clauses of C_s can be done in time $O^*(2^{|C_s|})$ since MAX SAT-k can be solved in $O^*(2^m)$ time by Lemma 9.

We now show the soundness of the algorithm. Let S_0 be an optimal solution. From the root of the branching tree, while it is possible, we follow a branch where all the nodes are in accordance with 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 construction, this branch terminates at v, which is either a leaf or a node that deviates from S_0. The leaf case being a special case of v being a deviating node, we assume that v 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 `pSAT-k` and consider $C_d := C(x_i, C_u)$ that is the set of clauses not yet in C_s in which 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, a solution at the leaves of the branching tree is as good as S_0. Setting x_j to *false*, one loses at

most $d_j(C_u)$ clauses and setting x_i to *true*, one gains exactly $d_i(C_u)$ clauses. Indeed, we recall that no clause of C_d can be satisfied by S_0, and *a fortiori* by S_n, since otherwise, v would not deviate from S_0 (if a clause $c \in C_d$ is satisfied by S_0, then the child of v that commits to satisfy c remains in accordance with S_0). And, by our greedy choice, $d_i(C_u) \geqslant d_j(C_u)$. □

We may observe that the previous algorithm has a worse time complexity than the already known FPT algorithm for MAX k-SET COVER [4]. This is rather not surprising since, as we recall, MAX k-SET COVER is a particular case of MAX SAT-k corresponding to CNFs without negative literals.

We close this section by recalling that if the length of the clauses is also part of the parameter, the decision version SAT-k is FPT [26]. In other words, denoting by l-SAT-k, the version SAT-k where each clause contains at most l literals, the following holds.

Proposition 11. *[26] l-SAT-k parameterized by $k + l$ is FPT.*

The proof of Proposition 11, as it is given by Niedermeier [26] works only because one has to satisfy all the clauses. The parameterized complexity of MAX SAT-k with respect to $k + l$ still remains unclear and, to our opinion, deserves further investigation.

5 Some Preliminary Thoughts About an Enhanced Weft Hierarchy: the Counting Weft Hierarchy

A natural way to generalize any problem Π where one has to find a solution which universally satisfies a property is to define PARTIAL Π, 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. Cai studied the parameterized complexity of such partial problems (and others) in [7].

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 by Guo et al. [22]). This is a quite important asymmetry between classical complexity theory as we know it from the literature (see, for example, [21, 27, 28]) 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 depth and weft 1 or 2. By definition of the W-hierarchy, the fact that k *input*-vertices of the boolean circuit can be set to true permits to deal with cardinality constraint problems, but it is not suitable to problems, such as MAX k-SET COVER, where both 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 the corresponding ones in the weft hierarchy (W-hierarchy). Basically, in the boolean circuit, we generalize the *and*-vertex to a *counting*-vertex.

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

Based upon the sketchy definition just above, the following can be proved by just taking the usual circuits for MIN SET COVER and SAT (recall for completeness that for MIN SET COVER, the

input-vertices are the sets, elements are large *or*-vertices taking as input the sets where they appear, and the *output*-vertex is a large *and*-vertex taking all or-vertices as input) and replacing the corresponding large *and*-vertices by vertices C_p.

Proposition 12. *The following inclusions hold for the counting weft hierarchy: both* MAX k-SET COVER *and* MAX SAT-k *are in CW[2].*

As mentioned in the introduction, the results by Fellows et al. [20] which also focus on parallel W-hierarchy with other types of vertices, cannot be used here since the counting vertices are symmetric but not bounded.

Acknowledgement. The pertinent suggestions and comments of two anonymous referees have greatly improved the quality of this paper.

References

1. A. A. Ageev and M. Sviridenko. Approximation algorithms for maximum coverage and max cut with given sizes of parts. In G. Cornuéjols, R. E. Burkard, and G. J. Woeginger, editors, *Proc. Conference on Integer Programming and Combinatorial Optimization, IPCO'99*, volume 1610 of *Lecture Notes in Computer Science*, pages 17–30. Springer-Verlag, 1999.
2. A. Badanidiyuru, R. Kleinberg, and H. Lee. Approximating low-dimensional coverage problems. In T. K. Dey and S. Whitesides, editors, *Proc. Symposuim on Computational Geometry, SoCG'12, Chapel Hill, NC*, pages 161–170. ACM, 2012.
3. A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.
4. M. Bläser. Computing small partial coverings. *Inform. Process. Lett.*, 85(6):327–331, 2003.
5. E. Bonnet, B. Escoffier, V. Th. Paschos, and E. Tourniaire. Multi-parameter complexity analysis for constrained size graph problems: using greediness for parameterization. *Algorithmica*, 71(3):566–580, 2015.
6. E. Bonnet, V. Th. Paschos, and F. Sikora. Multiparameterizations for MAX k-SET COVER and related satisfiability problems. *CoRR*, abs/1309.4718, 2013.
7. L. Cai. Parameter complexity of cardinality constrained optimization problems. *The Computer Journal*, 51:102–121, 2008.
8. L. Cai and X. Huang. Fixed-parameter approximation: conceptual framework and approximability results. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 96–108. Springer-Verlag, 2006.
9. M. Cesati. The turing way to parameterized complexity. *J. Comput. System Sci.*, 67(4):654–685, 2003.
10. J. Chen, D. K. Friesen, W. Jia, and I. A. Kanj. Using nondeterminism to design deterministic algorithms. In R. Hariharan, M. Mukund, and V. Vinay, editors, *Proc. Foundations of Software Technology and Theoretical Computer Science, FSTTCS'01*, volume 2245 of *Lecture Notes in Computer Science*, pages 120–131. Springer-Verlag, 2001.
11. Y. Chen, M. Grohe, and M. Grüber. On parameterized approximability. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 109–120. Springer-Verlag, 2006.
12. Y. Chen and B. Lin. The constant inapproximability of the parameterized dominating set problem. *CoRR*, abs/1511.00075, 2015.
13. G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms. *Management Sci.*, 23(8):789–810, 1977.
14. F. Della Croce and V. Th. Paschos. Efficient algorithms for the MAX k-VERTEX COVER problem. *J. Comb. Optim.*, 28(3):674–691, 2014.
15. 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. In R. G. Downey, M. R. Fellows, and F. Dehne, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'04*, volume 3162 of *Lecture Notes in Computer Science*, pages 271–280. Springer-Verlag, 2004.
16. R. G. Downey and M. R. Fellows. *Parameterized complexity.* Monographs in Computer Science. Springer, New York, 1999.

17. R. G. Downey, M. R. Fellows, and C. McCartin. Parameterized approximation problems. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 121–129. Springer-Verlag, 2006.
18. 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.
19. U. Feige. A threshold of $\ln n$ for approximating set cover. *J. Assoc. Comput. Mach.*, 45:634–652, 1998.
20. M. R. Fellows, J. Flum, D. Hermelin, M. Müller, and F. A. Rosamond. W-hierarchies defined by symmetric gates. *Theory Comput. Sys.*, 46(2):311–339, 2010.
21. M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.
22. J. Guo, R. Niedermeier, and S. Wernicke. Parameterized complexity of vertex cover variants. *Theory Comput. Syst.*, 41(3):501–520, 2007.
23. Y. Liu, S. Lu, J. Chen, and S.-H. Sze. Greedy localization and color-coding: improved matching and packing algorithms. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 84–95. Springer-Verlag, 2006.
24. D. Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008.
25. D. Moshkovitz. The projection games conjecture and the NP-hardness of $\ln n$-approximating set-cover. In A. Gupta, K. Jansen, J. D. P. Rolim, and R. A. Servedio, editors, *Proc. Workshop on Approximation Algorithms for Combinatorial Optimization Problems and Workshop on Randomization and Computation, APPROX-RANDOM'12*, volume 7408 of *Lecture Notes in Computer Science*, pages 276–287. Springer-Verlag, 2012.
26. R. Niedermeier. *Invitation to fixed-parameter algorithms*. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford, 2006.
27. C. H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
28. C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice Hall, New Jersey, 1981.
29. P. Skowron and P. Faliszewski. Fully proportional representation with approval ballots: approximating the maxcover problem with bounded frequencies in FPT time. In *AAAI Conference on Artificial Intelligence*, 2015.

Parameterized Exact and Approximation Algorithms for Maximum k-Set Cover and Related Satisfiability Problems

Édouard Bonnet[1], Vangelis Th. Paschos[2], Florian Sikora[2]

[1] Institute for Computer Science and Control, Hungarian Academy of Sciences (MTA SZTAKI)
bonnet.edouard@sztaki.mta.hu
[2] Université Paris-Dauphine, PSL Research University, CNRS, LAMSADE, Paris, France
{paschos,florian.sikora}@lamsade.dauphine.fr

Abstract. Given a family of subsets $\mathcal{S}$ over a set of elements X and two integers p and k, MAX k-SET COVER consists of finding a subfamily $\mathcal{T} \subseteq \mathcal{S}$ of cardinality at most k, covering at least p elements of X. This problem is W[2]-hard when parameterized by k, and FPT when parameterized by p. We investigate the parameterized approximability of the problem with respect to parameters k and p. Then, we show that MAX SAT-k, a satisfiability problem generalizing MAX k-SET COVER, is also FPT with respect to parameter p.

1 Introduction

In the MAX k-SET COVER problem, we are given a family of subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$, and two integers p and k. The goal is to find a subcollection $\mathcal{T}$ of at most k subsets that covers at least p elements. In what follows, we make the following two natural hypotheses for the instances of MAX k-SET COVER: (a) $S_i \neq S_j$, $i, j = 1, \ldots, m$ and (b) $S_i \not\subseteq S_j$, $i, j = 1, \ldots, m$.

MAX k-SET COVER is a well-known problem met in many real-world applications. To the best of our knowledge, it has been studied for the first time in the late seventies by Cornuejols et al. [13]. This combinatorial problem originated from a financial application, where one wishes to find an optimal location of bank accounts in order to maximize clearing time. Since then, it is used for modeling real problems met in several areas such as databases, social networks, sensor placement, information retrieval, etc. A non-exhaustive list of references to such applications can be found in Badanidiyuru and al. [2].

MAX k-VERTEX COVER, the graph version of MAX k-SET COVER is defined as follows: given a graph $G = (V, E)$ and two integers k and p, one wants to determine k vertices that cover at least p edges. MAX k-VERTEX COVER is a special case of MAX k-SET COVER where any element of X belongs to exactly two sets of $\mathcal{S}$.

Both MAX k-SET COVER and MAX k-VERTEX COVER are very important problems, since they are natural generalizations of MIN SET COVER and MIN VERTEX COVER, respectively. Both are NP-hard (setting $p = n$, MAX k-SET COVER becomes the seminal MIN SET COVER problem; setting $p = |E|$, MAX k-VERTEX COVER coincides with the MIN VERTEX COVER problem).

MAX k-SET COVER is known to be approximable within a factor $1 - 1/e$ (to our knowledge, it is the only polynomial approximation result known for MAX k-SET COVER on general instances) but, for any $\epsilon > 0$, no polynomial algorithm can approximate it within ratio $1 - 1/e + \epsilon$ unless P = NP [19], while MIN SET COVER is polynomially inapproximable within ratio $(1 - \epsilon) \ln n$, for arbitrarily small $\epsilon > 0$, unless P = NP [25]. On the other hand, MAX k-VERTEX COVER, is APX-hard and the best-known approximation ratio for this problem, in general graphs, is bounded below by $3/4$, obtained by a very smart linear programming method by Ageev et al. [1].

MAX SAT-k is a satisfiability problem closely related to MAX k-SET COVER. It is also a natural "fixed cardinality generalization" of MAX SAT. In MAX SAT-k, we are given a CNF on n variables and m clauses and we ask for setting to *true* at most k variables satisfying at least p clauses. One may observe that MAX SAT-k without negation is MAX k-SET COVER.

The goal of the paper is to establish several parameterized results for MAX k-SET COVER and for MAX SAT-k. We mainly study the parameterized approximability of the former and exact parameterization of the latter.

2 Preliminaries

We first give the basic definitions of the parameterized complexity theory. A parameterized problem (Π, k) is said *fixed-parameter tractable* (or in the class FPT) with respect to a parameter k if it can be solved by an algorithm with running time $f(k) \cdot |I|^{O(1)}$ time (in *fpt-time*), where f is some computable function and $|I|$ is the instance size. Such algorithms are called *fixed-parameter tractable algorithms*, or FPT algorithms. A parameterized reduction (or FPT reduction) from a problem Π_1 to a problem Π_2 is a mapping of an instance (I, k) of Π_1 to an instance (I', k') of Π_2, computable in time $f(k) \cdot |I|^{O(1)}$, such that $(I, k) \in \Pi_1 \Leftrightarrow (I', k') \in \Pi_2$, $k' \leqslant g(k)$, and $|I'| \leqslant h(k) \cdot |I|^{O(1)}$ for some computable functions f, g, and h. This seemingly technical definition is just tailored to ensure that if Π_2 is in FPT and there is an FPT reduction from Π_1 to Π_2, then Π_1 is also in FPT. Some problems such as CLIQUE parameterized by the solution size are not in FPT. In fact, there is a whole hierarchy of classes beyond FPT: FPT $\subseteq$ W[1] $\subseteq$ W[2] $\subseteq \cdots \subseteq$ W[P] $\subseteq$ XP. It is commonly believed that FPT $\neq$ W[1].

We need some additional definitions in order to give a precise meaning to those classes. A *boolean circuit* is a directed acyclic graph where every vertex of in-degree 0 is an *input vertex*, every vertex of in-degree 1 is a *negation vertex* and every vertex of in-degree greater than 2 is either an *and-vertex* or an *or-vertex*. Exactly one vertex with out-degree 0 is the *output vertex*. The *depth* of such a circuit is the maximum length of a path from an input vertex to the output vertex, and the *weft* of such circuit is the maximum number of *large* vertices on a path from an input vertex to the output vertex. A vertex is *large* if its in-degree exceeds some pre-agreed constant bound. Giving boolean values to the input vertices determines the value of every vertex in the classic way, and in particular, if the output vertex receive value true for a given assignment, we say that this assignment satisfies the circuit.

The WEIGHTED CIRCUIT SATISFIABILITY (WCS) problem takes as input a boolean circuit and an integer k and decides if there is a satisfying assignment for this circuit with exactly k input vertices set to true. A parameterized problem Π belongs to the class W[t], $t \geqslant 1$, if there is an FPT reduction from Π to WCS restricted to circuits of weft at most t. A parameterized problem Π is hard for the class W[t] (with $t \geqslant 1$) if, for any problem Π' in W[t], there is an FPT reduction from Π' to Π; or equivalently if there is an FPT reduction from WCS restricted to circuits of weft at most t to Π. A parameterized problem Π is W[t]-complete if it is W[t]-hard and in W[t]. For example, MAX INDEPENDENT SET parameterized by the size of the solution is W[1]-complete and MIN DOMINATING SET parameterized by the size of the solution is W[2]-complete. The class W[P] contains problems reducible to WCS without constraints on the weft of the circuits. The class XP contains problems solvable in time $|I|^{f(k)}$, where f is any computable function. See for example the monograph of Downey and Fellows for more details about fixed-parameter tractability [16].

We now go back to our problems. MAX k-SET COVER is W[2]-hard for the parameter k since, by setting $p = n$, we obtain an instance of MIN SET COVER which is W[2]-hard. An FPT algorithm with respect to the standard parameter p is given by Bläser [4]. Let us note that recently and independently certain aspects of parameterized complexity of MAX k-SET COVER have also been studied by Skowron and Faliszewski [29]. These results are summarized in Table 1.

Note that different parameterizations are possible for the same problem. In fact, one can also parameterize a problem by a combination of parameters (instead of just one parameter). A *multiparameterization* by parameters $k_1, k_2, \ldots, k_h$ consists of taking $k_1 + k_2 + \ldots + k_h$ as parameter. Multiparameterization poses novel and interesting open questions. For MAX k-SET COVER, for instance, several natural parameters as p (commonly called the "standard parameter"), k, $\Delta = \max_i\{|S_i|\}$ and $f = \max_i |\{j | x_i \in S_j\}|$ (commonly called the maximum frequency), can be jointly involved in a complexity study of the problem.

We first give multiparameterization of MAX k-SET COVER with respect parameters k and p (Section 3). The most important part of this section is Subsection 3.2 dedicated to the study

of the parameterized approximation of MAX k-SET COVER for these two parameters. Consider a problem Π, parameterized by some parameter π. Then, we say that Π is *parameterized r-approximable* if there exists an algorithm A that is FPT when parameterized by π such that:

- if Π is a minimization problem, then for any instance I of Π where $\pi \leqslant \beta$, A produces a solution with value at most $r\beta$; otherwise it returns any solution (which can be smaller or greater than $r\beta$);
- if Π is a maximization problem, then for any instance I of Π where $\pi \geqslant \beta$, A produces a solution with value at least $r\beta$; otherwise it returns any solution (which can be smaller or greater than $r\beta$).

This line of research was initiated by three independent works [17, 8, 11]. For an excellent overview, see the survey of Marx [24]. It aims at beating polynomial approximation barriers by offering more generous running time. The underlying question motivating Subsection 3.2 is *to what extent parameterized approximation is able to do it for* MAX k-SET COVER*?*.

Skowron and Faliszewski show, in [29], that it is possible when the parameter considered is $k + f$, where f is the frequency of the MAX k-SET COVER-instance, i.e., the maximum number of sets in $\mathcal{S}$, a ground element belongs to. But what happens when considering only k instead?

For parameter k we mainly show a conditional result, 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 $(1 - \varepsilon) \ln n$, for some fixed $\varepsilon > 0$. Even if this is a conditional result, we conjecture that the right answer is negative, i.e., that MAX k-SET COVER is inapproximable within ratio greater than $1 - 1/e + \epsilon$, for any $\epsilon > 0$, in FPT time parameterized by k. We also give in Subsection 3.2 a weaker negative result for MAX k-SET COVER, namely, that under the same parameterization, it is inapproximable within ratio $1 - (1/n)^{\sqrt[4]{\ln n}}$, unless ETH[3] fails.

Let us mention that MAX k-VERTEX COVER can be approximately solved within ratio $1 - \epsilon$, for any fixed $\epsilon > 0$, in FPT time parameterized by k [24].

For parameter p, we show that MAX k-SET COVER can be solved within ratio (strictly) greater than $1 - 1/e$ in time FPT parameterized by p, which is smaller than that needed for the exact solution of the problem.

In Section 4, we settle the parameterized complexity of MAX SAT-k, where, given a CNF on n variables and m clauses, one asks for setting to *true* at most k variables satisfying at least p clauses. The main result is that MAX SAT-k is FPT with respect to parameter p.

To prove that, we refine a technique for obtaining multiparameterized FPT algorithms developed by Bonnet et al. [5], 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 output. The basic idea of the technique is to branch on:

- a greedy extension of the partial solution;
- other extensions in the neighborhood 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. Although the techniques are not the same, greediness-for-parameterization shares some common points with the *greedy localization* technique (see [10, 15, 23] 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.

In Section 5, we suggest an enhanced weft hierarchy called *counting weft hierarchy* dedicated to those cardinality-constrained problems, such as MAX SAT-k and MAX k-SET COVER which are

[3] *Exponential Time Hypothesis*: there is a real number $\delta > 0$ such that 3-SAT is not solvable in $O(2^{\delta n})$ on instances with n variables.

W[i]-hard for some i, and in W[P], but not even known to be in W[j] for some integer j. Related issues have been discussed by Fellows et al. [20].

3 Parameterizations for MAX k-SET COVER

3.1 Exact Parameterization

As mentioned in Section 1, in the MAX k-SET COVER problem, we are given a family of subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$, and two integers p and k. The goal is to find a subcollection of $\mathcal{S}$ of size k that covers at least p elements of X.

Let us first note that, as $p \leqslant \Delta k$, the FPT result for MAX k-SET COVER with respect to parameter p presented by Bläser [4], immediately implies that this problem is also FPT when parameterized by $k + \Delta$. An alternative proof using greediness-for-parameterization is given by Bonnet et al. [6]. It might be worth reading it since it is a good introduction to the FPT algorithm for MAX SAT-k in Section 4.

We now explain why MAX k-SET COVER parameterized by $k + f$ is W[1]-hard. Each instance $(\mathcal{S}, X)$ of MAX k-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 MAX k-SET COVER with frequency 2 is equivalent to the MAX k-VERTEX COVER problem where, given a graph G and a number k, the goal is to cover at least p edges with k vertices. Thus, MAX k-VERTEX COVER, W[1]-hard with respect to k [7], is a restricted case of MAX k-SET COVER.

Note that in the reduction above the maximum set-cardinality Δ in an instance of MAX k-SET COVER with $f = 2$, coincides with the maximum degree of the derived graph. Hence, with the same argument, it can be shown that MAX k-SET COVER is not in XP when parameterized by $\Delta + f$, since MAX k-VERTEX COVER is NP-hard even in graphs with bounded degree (being, as mentioned above, a generalization of MIN VERTEX COVER that remains NP-hard even in these graphs).

Finally, in the following proposition, we prove that MAX k-SET COVER parameterized by k belongs to W[P].

Proposition 1. MAX k-SET COVER *parameterized by k belongs to W[P].*

Proof. The proof is in exactly the same spirit with the proof by Cesati [9]. We reduce MAX k-SET COVER to BOUNDED NON-DETERMINISTIC TURING MACHINE COMPUTATION which is a known W[P]-complete problem [16] and defined as follows. Given a non-deterministic Turing machine M, an input word w, an integer n encoded in unary and a positive integer k, does $M(w)$ non-deterministically accept in at most n steps and using at most k deterministic steps?

Let $\mathcal{I} = (\mathcal{S} = \{S_1, \ldots, S_m\}, p)$ be an instance of 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 $|\mathcal{I}|$) number of deterministic steps (precisely, $O(|\mathcal{I}|^2)$). □

The results mentioned in this paragraph as well as literature results are summarized in Table 1.

Parameter:	$\Delta + f$	k	$k + f$	$k + \Delta$ or p	$n - p$	$(n - p) + k$
Status:	$\notin$ XP	W[2]-hard in W[P] (Prop. 1)	W[1]-hard in W[P]	FPT [4, 6]	$\notin$ XP [29]	W[2]-complete [29]

Table 1. Exact parameterized complexity of MAX k-SET COVER for different parameters.

3.2 Approximation Issues

Let us now handle parameterized approximation of MAX k-SET COVER. We first prove the following basic lemma that is an easy generalization of Proposition 5.2 given by Feige [19].

Lemma 2. *Any r-approximation algorithm, parameterized by k, for* MAX k-SET COVER *can be transformed into an FPT t-approximation algorithm, parameterized by the standard parameter (optimum value), for* MIN SET COVER *where:*

$$t < \left\lceil \frac{-\ln n}{\ln(1-r)} \right\rceil$$

Proof. The basic idea is similar to the idea of Feige [19] (Proposition 5.2). Its key ingredient is the following. Consider some algorithm `kSC-ALG` that solves MAX k-SET COVER. Then, it can iteratively be used to solve MIN SET COVER as follows. Consider an instance $I = (\mathcal{F}, U)$ of MIN SET COVER where $\mathcal{F}$ is a family of subsets of a ground set U. Iteratively run `kSC-ALG` for $k = 1, \ldots, m$ (where m is the size of $\mathcal{F}$). Eventually, one value of k will equal the value of the optimal solution for MIN SET COVER. Let us reason with respect to this value of k, denoted by k_0. Furthermore, assume that `kSC-ALG` achieves approximation ratio r for MAX k-SET COVER. Invoke it with value k_0, (note that now $p = n$, the size of the ground set U), remove the ground elements covered, store the k_0 elements used and relaunch it with value k_0, until all ground elements of U are removed. Since it is assumed to achieve approximation ratio r after its ℓ-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):

$$(1-r)^t n < 1 \Longrightarrow t < \left\lceil \frac{-\ln n}{\ln(1-r)} \right\rceil \tag{1}$$

Moreover, observe that the complexity of the algorithm derived for MIN SET COVER is at most m times the complexity of `kSC-ALG`; so, it remains FPT with respect to the optimal value for MIN SET COVER . □

Recall that, as mentioned in the beginning of Section 1, MAX k-SET COVER is inapproximable in polynomial time within ratio $1 - 1/e + \epsilon$, for any $\epsilon > 0$, unless P = NP [19]. We first prove in the sequel a conditional result, informally, getting such a ratio even in FPT time parameterized by k, is a rather difficult task. More precisely, we prove that if this were possible, then we could get, in parameterized time, an approximation ratio of $(1-\epsilon)\ln(n/\ln n)$ for MIN SET COVER, for some fixed $\epsilon > 0$. Even if this result is, properly speaking, a conditional result, it gives, in some sense, the measure of the difficulty of approximating MAX k-SET COVER within ratio strictly better than $1 - 1/e$ in FPT time parameterized by k. Next, we prove an FPT inapproximability result, namely that FPT approximation of MAX k-SET COVER within ratio greater than $1 - (1/n)^{\sqrt[4]{\ln k}}$ is impossible unless W[2] = FPT.

Proposition 3. MAX k-SET COVER *parameterized by k is inapproximable within ratio $(1-1/e+\epsilon)$, for any $\epsilon \in [0, 1/e)$, unless* MIN SET COVER *is approximable within ratio $(1-\eta)\ln(n/\ln n)$, for some fixed $\eta > 0$, in FPT time parameterized by the value of the optimum.*

Proof. Revisit Lemma 2, take $r = 1 - (1/e) + \epsilon$, for some $\epsilon \in [0, 1/e)$ and assume that the `kSC-ALG` of Lemma 2 (that is FPT in k) achieves approximation ratio r. Then, in order to prove the result claimed, follow the procedure described in Lemma 2 until there are at most $\ln n$, say $c \ln n$ for some $c \leqslant 1$, uncovered elements in U 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 [3]. Since the surviving ground set has size $c \ln n$, it is polynomial to optimally solve it. Reasoning exactly as in Lemma 2 we get:

$$n(1-r)^t = c\ln n \Longrightarrow t = \frac{\ln c + \ln\ln n - \ln n}{\ln(1-r)} \leqslant \frac{\ln\ln n - \ln n}{\ln(1-r)} \simeq \frac{\ln\ln n - \ln n}{-\epsilon e - 1}$$
$$= \frac{\ln n - \ln\ln n}{1+\epsilon e} = \frac{1}{1+\epsilon e}\ln\left(\frac{n}{\ln n}\right)$$

Setting $\eta = {}^{\epsilon e}/_{(1+\epsilon e)}$, the proof of the proposition is concluded. □

As one can see, the result of Proposition 3 is conditional and relates the parameterized approximability of MAX k-SET COVER within ratios better than the one achieved in polynomial time to the parameterized approximability of MIN SET COVER within ratios that are almost the same (in fact slightly smaller) as the one polynomially achieved for this problem. Furthermore this ratio is tight for the polynomial time (recall that it is NP-hard to approximate MIN SET COVER within ratio $(1-\varepsilon)\ln n$, for arbitrarily small $\varepsilon > 0$ [25]). We conjecture that the real parameterized (with respect to the optimum) inapproximability bound of MIN SET COVER is $O(\log n)$, so that the inapproximability bound (conditionally) conjectured by Proposition 3 is the correct one. But, unfortunately, we have not been able to prove it until now and the negative result by Moshkovitz in [25] does not seem to be usable as it is for the parameterized inapproximability of MAX k-SET COVER.

In what follows, in the spirit of Lemma 2 and of Proposition 3 and based upon a recent result by Chen and Lin [12], we show a weaker upper bound for the parameterized approximability of MAX k-SET COVER with respect to k.

Consider the MIN DOMINATING SET problem defined as follows: *given a graph $G = (V, E)$, determine a minimum size vertex subset $D \subseteq V$ such that every vertex of V is either in D or has a neighbor in D*. There exists a well-known approximability preserving reduction from MIN DOMINATING SET to MIN SET COVER that works as follows: given a graph $G = (V, E)$ of order n, instance of MIN DOMINATING SET, we transform it into an instance $I = (\mathcal{F}, U)$ of MIN SET COVER as follows:

- $\mathcal{F} = \{F_1, \ldots, F_n\}$;
- $U = \{u_1, \ldots, u_n\}$;
- $\forall i \in \{1, \ldots, n\}$, $F_i = \{u_j : v_j \in \Gamma[v_i]\}$, where $\Gamma[v_i]$ denotes the closed neighborhood of vertex $v_i \in V$.

Then, it is easy to see that any dominating set D of G corresponds to a set cover $\mathcal{F}'$ of I of the same cardinality by simply considering in $\mathcal{F}'$ the subsets of $\mathcal{F}$ having the same indices with the vertices of D and vice-versa. An immediate consequence of this reduction is that both problems share the same approximation ratios and inapproximability bounds.

Recently, Chen and Lin [12] have proved that, *under ETH, no FPT algorithm for* MIN DOMINATING SET *can achieve approximation ratio smaller than, or equal to, $\sqrt[4+\epsilon]{\ln \gamma(G)}$, for any positive constant ϵ, where $\gamma(G)$ denotes the cardinality of a minimum dominating set in G*. The reduction just described, immediately transfers this lower bound to MIN SET COVER; so the following holds for this latter problem: *under ETH, no FPT algorithm for* MIN SET COVER *can achieve approximation ratio smaller than, or equal to, $\sqrt[4+\epsilon]{\ln k_0}$, for any positive constant ϵ, where k_0 denotes the cardinality of a minimum set cover instance $(\mathcal{F}, U)$*.

Based upon the result by Chen and Lin [12], Lemma 2, and the approximation-preserving reduction from MIN DOMINATING SET to MAX k-SET COVER given just above, the following can be proved.

Proposition 4. MAX k-SET COVER *is inapproximable within ratio $1 - ({}^1/_n)^{\sqrt[4]{\ln k}}$ in FPT time parameterized by k, unless ETH fails.*

Proof. The proof follows the one of Lemma 2. From (1), taking for t (the ratio of MIN SET COVER) the $\sqrt[4]{\ln k_0}$-inapproximability bound derived by the the reduction above and by Chen and Lin in [12] and omitting (in order to simplify calculations) the ceiling in (1), some elementary algebra leads to:

$$\ln(1-r) \leqslant \frac{-\ln n}{t} \leqslant \frac{-\ln n}{\sqrt[4]{\ln k_0}} \Longrightarrow 1 - r \geqslant \left(\frac{1}{n}\right)^{\sqrt[4]{\ln K_0}} \Longrightarrow r \leqslant 1 - \left(\frac{1}{n}\right)^{\sqrt[4]{\ln k_0}}$$

as claimed. □

Let us note that a parameterized inapproximability bound weaker than that of Proposition 4 can be obtained as follows. Consider an instance G of MIN DOMINATING SET and transform it to an instance $I = (\mathcal{F}, U)$ 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 (the value of the optimal solution for MIN SET COVER in instance I), 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 error $c + 1$ (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 error is W[2]-hard [18]. So, the following corollary holds.

Corollary 5. MAX k-SET COVER *is inapproximable within ratio* $1 - c/n$ *in time parameterized by* k*, for any constant* $c > 1$*, unless W[2] = FPT.*

For the rest of the section, we relax the optimality requirement for the MAX k-SET COVER-solution and we show that we can devise an approximation algorithm with ratio strictly better than $1 - 1/e$ (beating so the polynomial inapproximability bound of Feige [19]), that runs in FPT time parameterized by k and Δ but whose complexity is lower (although depending on the accuracy) than the best exact parameterized complexity for MAX k-SET COVER.

In fact, we are going to prove a stronger result, claiming that, for any polynomial approximation ratio r achieved by some polynomial time algorithm `APPROX` and any parameterized algorithm `PARAM`(π) for MAX k-SET COVER, where π can be a single parameter or a vector of parameters, r can be improved in FPT time parameterized by π by an algorithm whose running time is smaller than the one of `PARAM`(π).

The way of doing it is to built a kind of hybrid algorithm that, informally, in the case of MAX k-SET COVER we deal with, where $\pi = (k, \Delta)$, it works as follows:

- it solves MAX k-SET COVER by invoking `PARAM`(π'), with $\pi' = (k', \Delta)$ for some $k' < k$, stores the solution computed, and removes it from the initial instance;
- it invokes `APPROX` on the surviving instance and stores the solution obtained;
- it takes the union of the two solutions.

Here, our objective is to establish a trade-off between the solution quality and the running time. Therefore, the running time of the presented algorithms depend on the approximation ratio.

Assume an FPT (exact) algorithm running in time $O^*(F(k, \Delta))$ for some function F (that is Algorithm `PARAM`(k, Δ)) together with an approximation algorithm `APPROX` achieving approximation ratio r for MAX k-SET COVER and consider Algorithm 1, called `pSC-IMPROVED` in what follows, running on an instance $(\mathcal{S}, X)$ of MAX k-SET COVER, where X is the ground set and $\mathcal{S}$ a family of subsets of X.

Algorithm 1: A description of the algorithm `pSC-IMPROVED`.

Input: An instance $(\mathcal{S}, X, k)$ of MAX k-SET COVER.
Output: A subfamily $\hat{\mathcal{T}} \subset \mathcal{S}$ containing k subsets.
fix some $\varepsilon > 0$;
take $\mu \in (0, 1)$ such that $\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu$;
set $k' = \mu k$;
run Algorithm `PARAM`(k', Δ) and store the solution computed (denoted by $\mathcal{T}_1$); let X_1 be the subset of X covered by $\mathcal{T}_1$;
set $\mathcal{S}' = \mathcal{S} \setminus \mathcal{T}_1$, $X' = X \setminus X_1$ and $k'' = k - k' = (1 - \mu)k$;
run the r-approximation algorithm `APPROX` on the MAX k-SET COVER-instance $(\mathcal{S}', X', k'')$ and store the solution $\mathcal{T}_2$ computed;
return $\hat{\mathcal{T}} = \mathcal{T}_1 \cup \mathcal{T}_2$;

Solution $\hat{\mathcal{T}}$ computed by pSC-IMPROVED has cardinality k, i.e., it is feasible for MAX k-SET COVER. Let us now analyze it in the following proposition.

Proposition 6. *Given a polynomial time approximation algorithm with ratio r for* MAX k-SET COVER*, for any $\mu \in (2-(1/r), 1]$,* MAX k-SET COVER *can be approximated within ratio $r(1-\mu)^2+\mu > r$, in $O^*(F(\mu k, \Delta))$-time, where Δ is the maximum cardinality of a set.*

Proof. Fix an optimal solution $\mathcal{T}^*$ and denote by X^* the subset of X covered by $\mathcal{T}^*$. Recall that in Algorithm 1, we denote by X_1 the subset of X covered by the set $\mathcal{T}_1$ computed by PARAM(k', Δ). Obviously:

$$|X_1| \geqslant \mu\,|X^*| \tag{2}$$

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

1. $X^* \setminus \tilde{X}$ is covered with at least k'' sets in $\mathcal{T}^*$ (denote by $\mathcal{T}''^*$ this system); otherwise, the sets of $\mathcal{T}^*$ covering $X^* \setminus \tilde{X}$ together with $\mathcal{T}_1$ would be a solution better than $\mathcal{T}^*$; indeed, if the elements of $X^* \setminus \tilde{X}$ were covered with less than k'' sets, i.e., if $|\mathcal{T}''^*| < k''$, then, since $\tilde{X} \subseteq X_1$, $\mathcal{T}_1 \cup \mathcal{T}''^*$ would be a set system of size $|\mathcal{T}_1 \cup \mathcal{T}''^*| < k$ covering $|X^*|$ elements; in this case, completing (even greedily) the family $\mathcal{T}_1 \cup \mathcal{T}''^*$ with $k - |\mathcal{T}_1 \cup \mathcal{T}''^*|$ sets would lead to a k-sets subfamily of $\mathcal{S}$ covering more than $|X^*|$ ground elements, absurd since X^* is the value of an optimal solution for MAX k-SET COVER;
2. the elements of $X^* \setminus \tilde{X}$ are still present in the instance $(\mathcal{S}', X')$ where the r-approximation algorithm APPROX is invoked, as well as the subsets of $\mathcal{S}$ covering them, i.e., the sets of $\mathcal{T}''^*$;
3. hence, the k'' "best"[4] sets of $\mathcal{T}''^*$ form a feasible solution for MAX k''-SET COVER in $(\mathcal{S}', X')$ and cover more than $(k''/k)|X^* \setminus \tilde{X}|$ elements of $X^* \setminus \tilde{X}$.

Combining Facts 1, 2 and 3 and taking into account that $\mathcal{T}_2$ is an r-approximation for MAX k-SET COVER, the following holds denoting by X_2 the subset of X covered by $\mathcal{T}_2$:

$$\begin{aligned} |X_2| &\geqslant r \cdot \left|\bar{X}^*\right| \geqslant r \cdot \frac{k''}{k} \cdot \left(\left|X^* \setminus \tilde{X}\right|\right) = r \cdot \frac{k-k'}{k} \cdot \left(\left|X^* \setminus \tilde{X}\right|\right) \\ &= r \cdot (1-\mu) \left(\left|X^* \setminus \tilde{X}\right|\right) \end{aligned} \tag{3}$$

$$\left|X^* \setminus \tilde{X}\right| = |X^*| - \left|\tilde{X}\right| \geqslant |X^*| - |X_1| \tag{4}$$

Putting together (2), (3) and (4), we get the following for the approximation ratio of Algorithm pSC-IMPROVED:

$$\begin{aligned} \frac{|X_1| + |X_2|}{|X^*|} &\geqslant \frac{|X_1| + r \cdot (1-\mu) \cdot (|X^*| - |X_1|)}{|X^*|} \\ &\geqslant \frac{r \cdot (1-\mu) \cdot |X^*| + [1 - r \cdot (1-\mu)] \cdot |X_1|}{|X^*|} \\ &\geqslant \frac{|X^*| \cdot [r \cdot (1-\mu) + \mu \cdot [1 - r \cdot (1-\mu)]]}{|X^*|} \\ &= r \cdot (1-\mu) + \mu \cdot [1 - r \cdot (1-\mu)] = r(1-\mu)^2 + \mu \end{aligned} \tag{5}$$

Ratio in (5) is at least r, for any $\mu \in ((2 - (1/r)), 1]$.

For the overall running time, it suffices to observe that, since APPROX runs in polynomial time, the running time of pSC-IMPROVED is dominated by that of PARAM invoked within Algorithm 1. Thus, the whole complexity of Algorithm pSC-IMPROVED becomes $O^*(F(\mu k, \Delta))$, as claimed. □

[4] In the sense that they cover the most of the elements covered by any other union of k'' sets of $\mathcal{T}''^*$.

Revisit now the proof of Proposition 6 and set $r = 1 - e^{-1}$. Then, (5) becomes:

$$\begin{aligned}
\frac{|X_1| + |X_2|}{|X^*|} &\geqslant \frac{|X_1| + \left(1 - e^{-1}\right) \cdot (1 - \mu) \cdot (|X^*| - |X_1|)}{|X^*|} \\
&\geqslant \frac{\left(1 - e^{-1}\right) \cdot (1 - \mu) \cdot |X^*| + \left[1 - \left(1 - e^{-1}\right) \cdot (1 - \mu)\right] \cdot |X_1|}{|X^*|} \\
&\geqslant \frac{|X^*| \cdot \left[\left(1 - e^{-1}\right) \cdot (1 - \mu) + \mu \cdot \left[1 - \left(1 - e^{-1}\right) \cdot (1 - \mu)\right]\right]}{|X^*|} \\
&= \left(1 - e^{-1}\right) \cdot (1 - \mu) + \mu \cdot \left[1 - \left(1 - e^{-1}\right) \cdot (1 - \mu)\right] \\
&= \left(1 - e^{-1}\right) - \left(1 - 2e^{-1}\right) \cdot \mu + \left(1 - e^{-1}\right) \cdot \mu^2
\end{aligned}$$

This ratio is at least $1 - 1/e + \varepsilon$, for any $\varepsilon > (1 - e^{-1})\mu^2 - (1 - 2e^{-1})\mu$, and the following corollary holds.

Corollary 7. *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, let us take $\mu = 0.5 \geqslant (e-2)/(e-1)$. Then application of Corollary 7 leads, after some easy algebra to an approximation ratio at least $0.75 + (0.25/e) \geqslant 0.841$ achieved with complexity $F(k/2, \Delta)$. Even if, in the case of MAX k-SET COVER, the potential of the running time seems to be quite small, we think that looking for this type of trade-offs between approximation ratios and running times in order to try to beat polynomial approximation barriers is an interesting research program.

In [14] an analogous result is given. There, the authors try to beat the APX-hardness of MAX k-VERTEX COVER by a moderately exponential approximation algorithm, i.e., by an approximation algorithm achieving ratio $1 - \epsilon$, for any $\epsilon > 0$, and running in time that, although exponential, remains smaller than the running time of the best known exact algorithm for this problem. More precisely, the following is proved by Della Croce and Paschos [14]. *If $T(n, k)$ is the running time of an exact algorithm and ρ the approximation ratio of some polynomial approximation algorithm for* MAX k-VERTEX COVER*, then, for any $\epsilon > 0$,* MAX k-VERTEX COVER *can be approximated within ratio $1 - \epsilon$ with worst-case running time $T(n, [(2\rho - 1) + \sqrt{1 - 4\epsilon\rho}]k/2\rho)$.*

4 MAX SAT-k

We now study a generalization of MAX k-SET COVER, the MAX SAT-k problem. Recall that in MAX SAT-k, given a CNF formula on n variables and m clauses, the objective is to satisfy at least p clauses, by setting at most k variables to *true*.

Proposition 8. MAX SAT-k *parameterized by k is W[2]-hard and in W[P].*

Proof. Setting $p = m$, MAX SAT-k becomes SAT-k that is W[2]-hard [26] (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. □

Consider an instance ϕ of MAX SAT-k on a set of C clauses. Given any subset C' of C, we denote by $\mathrm{occ}^+(x_i, C')$ the number of positive occurrences of the variable x_i in C', and by $\mathrm{occ}^-(x_i, C')$ the number of its negative occurrences. We set $f(x_i) := \mathrm{occ}^+(x_i, C) + \mathrm{occ}^-(x_i, C)$; so, the frequency of the formula is $f := \max_i f(x_i)$.

Before proving that MAX SAT-k is FPT with respect to parameter p, let us introduce some vocabulary on 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 labeled with a sub-instance of I together

with a partial solution, or more generally with some data maintained by the algorithm. The *root* is labeled with I and a *leaf* is a sub-instance 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 sub-instance), the *children* of a sub-instance are the subinstances which label the children of v in the branching tree.

We now prove this simple lemma.

Lemma 9. MAX SAT-k *is solvable in time* $O^*(2^m)$.

Proof. We take any variable x that appears positively in at least one clause *and* negatively in at least one clause. We do the standard branching: either set x to *true* (and decrease k by 1), or set x to *false*. This branching satisfies at least one more clause in each branch. Therefore, the branching tree is a subtree of the full binary tree with 2^m leaves. At a leaf ℓ of the branching tree, the remaining number of clauses is at most $m - \lambda$ where λ is the depth of ℓ. The branching stops when each variable appears only negatively, or only positively. At this 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. Therefore, at the leaves, the instances can now be seen as instances of MAX k-HITTING SET with at most $m - \lambda$ sets; or equivalently, MAX k-SET COVER with at most $m - \lambda$ elements which can be solved by standard dynamic programming in time $O^*(2^{m-\lambda})$. So, the overall running time is $O^*(2^m)$. □

We identify a solution of an instance of MAX SAT-k to the set S of variables put to *true*. It induces a set C' of satisfied clauses. The algorithm solving MAX SAT-k we will present, performs two kinds of choices: (1) setting a variable to *true* and (2) putting a clause which is not satisfied yet into a set C_s of clauses that should eventually be satisfied. Putting a clause in C_s means that we commit to satisfy it later. At any node of the branching tree, a child corresponds to either performing choice (1) for some given variable, or choice (2) for some given clause. A choice (1) is *in accordance* with S if it sets to *true* a variable in S. A choice (2) is *in accordance* with S if it puts in C_S a clause satisfied by solution S (i.e., this clause is in C'). A node v of the branching tree is *in accordance* with S if all the choices made from the root to this node are in accordance with 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.

Let us give a toy example to clarify those notions. Assume a solution which sets x_2 and x_3 to *true* and all the other variables to *false*. So, $S = \{x_2, x_3\}$. And, this assignment satisfies the following set of clauses: $\{c_2, c_4, c_5, c_6, c_8, c_9\}$. Say, the root of the branching tree has 4 children: setting x_1 to *true*, committing to satisfy c_1, committing to satisfy c_2, and committing to satisfy c_6. The two first children are not in accordance with S, but the two last are. Indeed, c_2 and c_6 are satisfied by S. Let us move to the child where we commit to satisfy c_2. Suppose this node has three children: setting x_2 to *true*, committing to satisfy c_1, committing to satisfy c_3. We now move to the child where we set x_2 to *true*. So far, we have only done choices in accordance with S, so our current node is in accordance with S. Now, say, this new node v has three children: setting x_1 to *true*, committing to satisfy c_3, committing to satisfy c_7. None of those choices is in accordance with S, so v deviates from S.

Proposition 10. MAX SAT-k *parameterized by* p *is FPT.*

Proof. Let (C, k, p) be an instance of MAX SAT-k where C is a set of clauses over a set of variables $X = \{x_1, x_2, \ldots, x_n\}$, k is the maximum number of variables that can be set to true, and p the minimum number of clauses to satisfy. We can assume that $p < m/2$. Indeed, if $p \geqslant m/2$, the algorithm of Lemma 9 is an FPT algorithm. We also assume that the number of clauses containing only negative literals is bounded above by $m/2$. 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 $p < f/2$ are always YES-instances, since one can set one variable x_i with frequency f to *true* if $\mathrm{occ}^+(x_i, C) \geqslant \mathrm{occ}^-(x_i, C)$, and to *false*

otherwise, and set all the other variables to false. This assignment does indeed satisfy at least $\max(\text{occ}^+(x_i, C), \text{occ}^-(x_i, C)) \geqslant f/2 < p$ clauses.

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 $p \geqslant f/2$ and $p \geqslant k$, so our parameter might as well be $p + f + k$.

We construct a branching algorithm which operates accordingly to a greedy criterion. A solution, or *complete assignment*, is given by a set S of size up to k which contains all the variables set to *true*. Additionally, we maintain a list C_s of clauses that we satisfy or commit to satisfy. For notational convenience we define $C_u := C \setminus C_s$, $d_i(C') := \text{occ}^+(x_i, C')$, 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. Finally, $C(x_i, C') := C^+(x_i, C') \cup C^-(x_i, C')$. Algorithm `pSAT-k` (see Algorithm 2) is fairly simple. We find the variable x that, if set to *true*, would satisfy the maximum number of clauses among the still unsatisfied clauses. We branch on setting x to *true* (choice (1)) or for each still unsatisfied clause c that x would satisfy, on putting c in C_s (choice (2)).

Algorithm 2: A description of the algorithm `pSAT-k`.

Input: A set C of clauses on a set X of variables.
Output: A subset $S \subseteq X$ of size at most k such that setting all the variables in S to *true* and all the variables in $X \setminus S$ to *false*, satisfies the greatest number of clauses in C.

```
set S = ∅, C_s = ∅;
ALG1(S, C_s):
if |S| < k and |C_s| < p then
    pick the variable x_i maximizing d_i(C \ C_s);
    run ALG1(S ∪ {x_i}, C_s ∪ C^+(x_i, C \ C_s));
    for each clause c ∈ C(x_i, C \ C_s) do
        run ALG1(S, C_s ∪ {c});
else
    if |S| = k then
        store S;
    else
        (|C_s| ⩾ p) store a complete assignment satisfying C_s, if possible;
return the best among the solutions stored;
```

The branching tree has depth at most $k + p$ and arity at most $f + 1$, so the running time of `pSAT-k` is $O^*(2^p(f+1)^{k+p}) = O^*(p^{O(p)})$, that is FPT with respect to parameter p, because completing a solution to satisfy all the clauses of C_s can be done in time $O^*(2^{|C_s|})$ since MAX SAT-k can be solved in $O^*(2^m)$ time by Lemma 9.

We now show the soundness of the algorithm. Let S_0 be an optimal solution. From the root of the branching tree, while it is possible, we follow a branch where all the nodes are in accordance with 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 construction, this branch terminates at v, which is either a leaf or a node that deviates from S_0. The leaf case being a special case of v being a deviating node, we assume that v 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 `pSAT-k` and consider $C_d := C(x_i, C_u)$ that is the set of clauses not yet in C_s in which 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, a solution at the leaves of the branching tree is as good as S_0. Setting x_j to *false*, one loses at

most $d_j(C_u)$ clauses and setting x_i to *true*, one gains exactly $d_i(C_u)$ clauses. Indeed, we recall that no clause of C_d can be satisfied by S_0, and *a fortiori* by S_n, since otherwise, v would not deviate from S_0 (if a clause $c \in C_d$ is satisfied by S_0, then the child of v that commits to satisfy c remains in accordance with S_0). And, by our greedy choice, $d_i(C_u) \geqslant d_j(C_u)$. □

We may observe that the previous algorithm has a worse time complexity than the already known FPT algorithm for MAX k-SET COVER [4]. This is rather not surprising since, as we recall, MAX k-SET COVER is a particular case of MAX SAT-k corresponding to CNFs without negative literals.

We close this section by recalling that if the length of the clauses is also part of the parameter, the decision version SAT-k is FPT [26]. In other words, denoting by l-SAT-k, the version SAT-k where each clause contains at most l literals, the following holds.

Proposition 11. *[26] l-SAT-k parameterized by $k + l$ is FPT.*

The proof of Proposition 11, as it is given by Niedermeier [26] works only because one has to satisfy all the clauses. The parameterized complexity of MAX SAT-k with respect to $k + l$ still remains unclear and, to our opinion, deserves further investigation.

5 Some Preliminary Thoughts About an Enhanced Weft Hierarchy: the Counting Weft Hierarchy

A natural way to generalize any problem Π where one has to find a solution which universally satisfies a property is to define PARTIAL Π, 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. Cai studied the parameterized complexity of such partial problems (and others) in [7].

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 by Guo et al. [22]). This is a quite important asymmetry between classical complexity theory as we know it from the literature (see, for example, [21, 27, 28]) 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 depth and weft 1 or 2. By definition of the W-hierarchy, the fact that k *input*-vertices of the boolean circuit can be set to true permits to deal with cardinality constraint problems, but it is not suitable to problems, such as MAX k-SET COVER, where both 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 the corresponding ones in the weft hierarchy (W-hierarchy). Basically, in the boolean circuit, we generalize the *and*-vertex to a *counting*-vertex.

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

Based upon the sketchy definition just above, the following can be proved by just taking the usual circuits for MIN SET COVER and SAT (recall for completeness that for MIN SET COVER, the

input-vertices are the sets, elements are large *or*-vertices taking as input the sets where they appear, and the *output*-vertex is a large *and*-vertex taking all or-vertices as input) and replacing the corresponding large *and*-vertices by vertices C_p.

Proposition 12. *The following inclusions hold for the counting weft hierarchy: both* MAX k-SET COVER *and* MAX SAT-k *are in CW[2].*

As mentioned in the introduction, the results by Fellows et al. [20] which also focus on parallel W-hierarchy with other types of vertices, cannot be used here since the counting vertices are symmetric but not bounded.

Acknowledgement. The pertinent suggestions and comments of two anonymous referees have greatly improved the quality of this paper.

References

1. A. A. Ageev and M. Sviridenko. Approximation algorithms for maximum coverage and max cut with given sizes of parts. In G. Cornuéjols, R. E. Burkard, and G. J. Woeginger, editors, *Proc. Conference on Integer Programming and Combinatorial Optimization, IPCO'99*, volume 1610 of *Lecture Notes in Computer Science*, pages 17–30. Springer-Verlag, 1999.
2. A. Badanidiyuru, R. Kleinberg, and H. Lee. Approximating low-dimensional coverage problems. In T. K. Dey and S. Whitesides, editors, *Proc. Symposuim on Computational Geometry, SoCG'12, Chapel Hill, NC*, pages 161–170. ACM, 2012.
3. A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.
4. M. Bläser. Computing small partial coverings. *Inform. Process. Lett.*, 85(6):327–331, 2003.
5. E. Bonnet, B. Escoffier, V. Th. Paschos, and E. Tourniaire. Multi-parameter complexity analysis for constrained size graph problems: using greediness for parameterization. *Algorithmica*, 71(3):566–580, 2015.
6. E. Bonnet, V. Th. Paschos, and F. Sikora. Multiparameterizations for MAX k-SET COVER and related satisfiability problems. *CoRR*, abs/1309.4718, 2013.
7. L. Cai. Parameter complexity of cardinality constrained optimization problems. *The Computer Journal*, 51:102–121, 2008.
8. L. Cai and X. Huang. Fixed-parameter approximation: conceptual framework and approximability results. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 96–108. Springer-Verlag, 2006.
9. M. Cesati. The turing way to parameterized complexity. *J. Comput. System Sci.*, 67(4):654–685, 2003.
10. J. Chen, D. K. Friesen, W. Jia, and I. A. Kanj. Using nondeterminism to design deterministic algorithms. In R. Hariharan, M. Mukund, and V. Vinay, editors, *Proc. Foundations of Software Technology and Theoretical Computer Science, FSTTCS'01*, volume 2245 of *Lecture Notes in Computer Science*, pages 120–131. Springer-Verlag, 2001.
11. Y. Chen, M. Grohe, and M. Grüber. On parameterized approximability. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 109–120. Springer-Verlag, 2006.
12. Y. Chen and B. Lin. The constant inapproximability of the parameterized dominating set problem. *CoRR*, abs/1511.00075, 2015.
13. G. Cornuejols, M. L. Fisher, and G. L. Nemhauser. Location of bank accounts to optimize float: an analytic study of exact and approximate algorithms. *Management Sci.*, 23(8):789–810, 1977.
14. F. Della Croce and V. Th. Paschos. Efficient algorithms for the MAX k-VERTEX COVER problem. *J. Comb. Optim.*, 28(3):674–691, 2014.
15. 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. In R. G. Downey, M. R. Fellows, and F. Dehne, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'04*, volume 3162 of *Lecture Notes in Computer Science*, pages 271–280. Springer-Verlag, 2004.
16. R. G. Downey and M. R. Fellows. *Parameterized complexity.* Monographs in Computer Science. Springer, New York, 1999.

17. R. G. Downey, M. R. Fellows, and C. McCartin. Parameterized approximation problems. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 121–129. Springer-Verlag, 2006.
18. 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.
19. U. Feige. A threshold of $\ln n$ for approximating set cover. *J. Assoc. Comput. Mach.*, 45:634–652, 1998.
20. M. R. Fellows, J. Flum, D. Hermelin, M. Müller, and F. A. Rosamond. W-hierarchies defined by symmetric gates. *Theory Comput. Sys.*, 46(2):311–339, 2010.
21. M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness*. W. H. Freeman, San Francisco, 1979.
22. J. Guo, R. Niedermeier, and S. Wernicke. Parameterized complexity of vertex cover variants. *Theory Comput. Syst.*, 41(3):501–520, 2007.
23. Y. Liu, S. Lu, J. Chen, and S.-H. Sze. Greedy localization and color-coding: improved matching and packing algorithms. In H. L. Bodlaender and M. A. Langston, editors, *Proc. International Workshop on Parameterized and Exact Computation, IWPEC'06*, volume 4169 of *Lecture Notes in Computer Science*, pages 84–95. Springer-Verlag, 2006.
24. D. Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008.
25. D. Moshkovitz. The projection games conjecture and the NP-hardness of $\ln n$-approximating set-cover. In A. Gupta, K. Jansen, J. D. P. Rolim, and R. A. Servedio, editors, *Proc. Workshop on Approximation Algorithms for Combinatorial Optimization Problems and Workshop on Randomization and Computation, APPROX-RANDOM'12*, volume 7408 of *Lecture Notes in Computer Science*, pages 276–287. Springer-Verlag, 2012.
26. R. Niedermeier. *Invitation to fixed-parameter algorithms*. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford, 2006.
27. C. H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
28. C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice Hall, New Jersey, 1981.
29. P. Skowron and P. Faliszewski. Fully proportional representation with approval ballots: approximating the maxcover problem with bounded frequencies in FPT time. In *AAAI Conference on Artificial Intelligence*, 2015.