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

Edouard Bonnet[(a)1,2], Vangelis Th. Paschos[(b)1,2,3], and Florian Sikora[(c)1,2]

[1]PSL Research University, Université Paris-Dauphine, LAMSADE
[2]CNRS UMR 7243
[3]Institut Universitaire de France

September 18, 2013

Abstract

Given a family of subsets $\mathcal{S} = \{S_1, \ldots, S_m\}$ over a set of elements $X = \{x_1, \ldots, x_n\}$ and an integer p, MAX k-SET COVER consists of finding a set $\mathcal{T}$ of at most k subsets covering at least p elements. This problem, when parameterized by k, can be easily shown to be W[2]-hard. Here, we settle the parameterized complexity of MAX k-SET COVER under several parameters as $\max\{k, \Delta\}$, where $\Delta = \max_i\{|S_i|\}$, p and $\max\{k, f\}$, where $f = \max_i |\{j | x_i \in S_j\}|$. We also study parameterized approximability of the problem with respect to parameters k and p. We also study parameterization of a satisfiability problem that is linked to MAX k-SET COVER in a sense explained in the paper. Finally, we sketch an enhancement of the classes of the W[·] that seems more appropriate for showing completeness of hard W[·]-problems.

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 an integer p and ask for finding a subcollection $\mathcal{T}$ of at most k subsets, called *solution* in what follows, that covers at least p elements.

MAX k-SET COVER is known to be NP-hard (just set $p = n$; then MAX k-SET COVER becomes the seminal MIN SET COVER problem). Also, it is known to

(a) edouard.bonnet@dauphine.fr
(b) paschos@lamsade.dauphine.fr
(c) florian.sikora@dauphine.fr

be approximable within a factor $1-(1/e)$, but no polynomial algorithm can approximate it within ratio $1-(1/e)+\epsilon$, for any $\epsilon > 0$, unless P = NP [14].

Concerning parameterized complexity of the problem, it is immediate that MAX k-SET COVER is W[2]-hard for parameter k, by just setting (once more) $p = n$ and using the fact that MIN SET COVER is W[2]-hard too (for more about the definition of the W[$\cdot$] hierarchy, as well as for everything about the foundations of parameterized complexity, the interested reader can be referred to [11]). But, to the best of our knowledge, no further parameterization results exist about this problem. This is one of the main goals of this paper.

For MAX k-SET COVER, several natural parameters as p (commonly called standard parameter), k, $\Delta = \max_i\{|S_i|\}$ and $f = \max_i |\{j|x_i \in S_j\}|$, where the quantity$|\{j|x_i \in S_j\}|$ is commonly called the frequency of element x_i, can be involved to any complexity study of the problem. On the other hand, instead of some single parameter, one can use combinations of parameters. Note that in several papers (see, for example, [3, 5]), the parameterized complexity of a lot of problems has been studied via a multiparameterized analysis involving pairs of parameters.

In this paper we first study multiparameterization of MAX k-SET COVER with respect to pairs of the parameters mentioned just above (Section 2.1). For this, we use a technique for obtaining multiparameterized results developed in [3], called *greediness-for-parameterization*. Informally, if the analysis involves, for instance, k and Δ, the basic idea behind it is the following. Perform a branching with respect to a set chosen upon some greedy criterion. For instance, this criterion could be to consider some set S that maximizes the number of ground elements covered in the solution under construction. Without branching, such a greedy criterion is not optimal. However, if at each step either the greedily chosen set S, or some other set satisfying some other problem-specific criterion is a good choice (i.e., it is in an optimal solution), then a branching rule on these sets leads to a branching tree whose size is, for instance, bounded by a function of k and Δ, and at least one leaf of which is an optimal solution. Let us note that this technique has some common points with the so-called *greedy localization* technique (see [8, 10, 16] for some applications of it). Here, one, uses a local search approach, i.e., one starts from a greedily (or, in any case, not optimally) computed solution and then, based on some properties of it, extends it to an optimal one. Greedy localization technique seems, however, less general than greediness-for-parameterization, since it is more adapted to maximization problems.

Then, in Section 2.2, we handle parameterized approximation of MAX k-SET COVER for parameters k and p. We say that a minimization (maximization, respectively) problem Π, together with a parameter π, is *parameterized r-approximable* if there exists an FPT-time algorithm which computes a solution of size at most (at least, respectively) $r\pi$ whenever the input instance has a solution of size at most (at least, respectively) π; otherwise, it gives an arbitrary results (i.e., indifferently $\leqslant r\pi$, or $\geqslant r\pi$). This line of research was initiated by three independent works [12, 6, 9]. For an excellent overview, see [17]. For parameter k we show a conditional result that is considered very likely to be

Parameter	$\Delta + f$	k	$k + f$	$(k + \Delta)$ & p
Complexity	$\notin$ XP	W[2]-hard, in W[P]	W[1]-hard	FPT

Table 1: Our main parameterized complexity results

true, 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 $o(\log n)$. For parameter p, we show that MAX k-SET COVER can be solved by an approximation schema in parameterized time smaller than that needed for the exact solution of the problem.

A careful reader of the parameterized complexity literature will surely remark that although there exist a lot of hardness results for several classes of the W[·] hierarchy there clearly exist much less completeness results. For instance, although as mentioned above, MAX k-SET COVER is W[2]-hard, the only inclusion that we are able to show for it, is in W[P] (see [7, 11] for its definition). In fact (see also [7]) there exists about a dozen of problems that behave in the same way, i.e., that are W[2]-hard, in W[P]. We give in Section 3 one more such problem, that can be seen as a kind of canonical problem, called MAX SAT-k. Here, given a CNF on n variables and m clauses, one asks for setting to true at most k variables satisfying at least p clauses. Then, we further settle parameterized complexity of MAX SAT-k.

Table 1 summarizes the main complexity results of the paper for both MAX k-SET COVER and MAX SAT-k.

The above observation about the dichotomy between hardness and completeness in the W[·] world, naturally gives rise to the following questions:

- "can one extend the definition of the classes of the W[·] hierarchy, in such a way that in the so obtained hierarchy (let us denote it by W$'$[·]), problems that are, say, W[1]-, or W[2]-hard (this is the case of the most of natural problems that are not FPT) correspondingly belong to classes W$'$[1] and W$'$[2]?";
- "can one, using standard FPT-reducibility, or extending it, in order to obtain a more appropriate one, prove the existence of complete problems for this new hierarchy?".

In Section 4, we try to give a preliminary answer to the first of the questions above. We sketch a hierarchy of circuits, called *counting weft hierarchy* the classes of which are larger than the corresponding in the weft hierarchy and show that several problems that only belong to W[P], belong to the first two classes of the counting weft hierarchy. Anyway, existence of complete problems for the classes of this new hierarchy remains open.

2 Parameterized complexity of MAX k-SET COVER

2.1 Exact parameterization

Let us first point out that MAX k-SET COVER, parameterized by $\max\{k, f\}$, is W[1]-hard. Indeed, for $f = 2$, i.e., when all the ground elements belong to at most 2 sets of $\mathcal{S}$, the instance of MAX k-SET COVER can be seen as a graph whose vertices correspond to the elements of $\mathcal{S}$ and edges to the elements of the ground set X. Then, MAX k-SET COVER becomes another well-known problem, the MAX k-VERTEX COVER problem where one wishes to cover at least p edges by k vertices. In this sense, MAX k-VERTEX COVER is a restricted case of MAX k-SET COVER. It is proved in [4] that this problem is W[1]-hard with respect to k.

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 XP-hard when parameterized by $\max\{\Delta, f\}$, since MAX k-VERTEX COVER is NP-hard even in graphs with bounded degree.

We now prove the main result of this section, namely that MAX k-SET COVER is FPT with respect to $\max\{k, \Delta\}$.

Proposition 1. MAX k-SET COVER *parameterized by* $\max\{k, \Delta\}$ *is FPT.*

Proof. We present a branching algorithm (Algorithm `ALGORITHM1`) whose particularity is to branch on two types of objects, a set T of subsets of $\mathcal{S}$ added to the solution and a subset C of the elements of X which we impose that they are covered. We call elements of C *candidate* elements. To understand the bound over the number of candidate elements used in `ALGORITHM1`, note that if there exists a solution $\mathcal{T}$ covering more than p elements, then there exists a solution $\mathcal{T}' \subseteq \mathcal{T}$ that covers more than p elements and less than $p + \Delta - 1$ ones. Indeed, when no subset can be removed from a solution $\mathcal{T}_0$ (without going under p elements covered) then the number of covered elements can not exceed $p + \Delta - 1$ since removing a subset can uncover at most Δ elements. For a set of subsets R, we denote by $\cup R$ the union of the elements of R, and by $\mathcal{S} \downarrow R$ the set of subsets $\{S_i \setminus \cup R : S_i \in \mathcal{S}\}$. The overall specification of Algorithm `ALGORITHM1` is the following:

- set $T = \emptyset$ and $C = \emptyset$;
- if $|T| < k$ and $|C| < p + \Delta - 1$ then:
  - pick a set $S_i \in \mathcal{S} \setminus T$ that covers the largest number of elements in $X \setminus C$;
  - branch on `ALGORITHM1`$(T \cup \{S_i\}, C \cup S_i)$ and, for each element $x \in S_i \setminus C$ branch on `ALGORITHM1`$(T, C \cup \{x\})$;
- else:
  - if $|T| = k$, then store T;

– else ($|C| = p + \Delta - 1$) store a solution covering C (if possible);

- output the best among the solutions stored.

Let us first establish the time complexity of `ALGORITHM1`. The number of children of a node of the branching tree is at most $\Delta + 1$. At each step, we add either a subset or an element, so the depth of the branching tree is, at most, $k + p + \Delta - 1$. Note that $p \leqslant k\Delta$ on non-trivial MAX k-SET COVER-instances. So, the branching tree has size $O((\Delta+1)^{(k+1)(\Delta+1)})$. On an internal node of the branching tree, `ALGORITHM1` only does polynomial computations. In a leaf of this tree, we find in time $O^*(2^{p+\Delta-1})$ if at most $k-|T|$ subsets of $(\mathcal{S} \downarrow T)[C]$ can cover at least $p-|\cup T|$ ground elements, where $\mathcal{S}[C]$ denotes the subsets in $\mathcal{S}$ entirely contained in C. So, `ALGORITHM1` works in time $O^*(2^{p+\Delta-1}(\Delta+1)^{(k+1)(\Delta+1)})$, i.e., it is fixed parameter with respect to $\max\{k, \Delta\}$.

Let us now show that `ALGORITHM1` is sound. Let $\mathcal{T}_0$ be a solution covering between p and $p + \Delta - 1$ elements. Recall that each node of the branching tree has one child adding a set to T and up to Δ children each adding one candidate element to C. Starting from the root, move to the one child adding a set in T while this set is indeed in the solution $\mathcal{T}_0$. We reach a node in the branching tree, where we have a set of common choices T_{cc} between `ALGORITHM1` and the solution $\mathcal{T}_0$, and we want to add a set S_i which is not in $\mathcal{T}_0$. We can suppose that $\mathcal{T}_0$ covers at least one element x in S_i, otherwise we substitute S_i to any subset of $\mathcal{T}_0 \setminus T_{cc}$ and, by assumption, we cover at least as many elements as $\mathcal{T}_0$. So, we move to one child that takes x as a candidate element. We continue to track $\mathcal{T}_0$ in the branching tree as just described until we reach a situation where either $k = |T|$ and T is at least as good as $\mathcal{T}_0$, or $\cup\mathcal{T}_0 \subseteq C$ and $T \subseteq \mathcal{T}_0$. To conclude just recall that `ALGORITHM1` computes a solution T_l containing at most $k - |T|$ subsets of $(\mathcal{S} \downarrow T)[C]$. So, $T \cup T_l$ is as good as $\mathcal{T}_0$, therefore, an optimal solution. ∎

The result of Proposition 1, has an interesting corollary about parameterization of MAX k-SET COVER by the standard parameter p. It is expressed in the following proposition.

Proposition 2. MAX k-SET COVER *parameterized by p is FPT.*

Proof. Let us first notice that, on non-trivial instances of MAX k-SET COVER, $p > \Delta$. Otherwise, it is a YES-instance taking the subset of size Δ in the solution and arbitrarily completing it. Furthermore, $n \geqslant p > k$, since if there exists an element which is not covered yet, adding any set containing such an element, adds at least 1 to the solution value. To summarize, when $p \leqslant \Delta$ or $p \leqslant k$ we can correctly answer in polynomial time and when $p > \Delta$ and $p > k$, the previous FPT algorithm works in $O^*(4^p \cdot p^{p^2})$, i.e., in time FPT with respect to p. ∎

Another problem, very similar to MAX k-SET COVER, is the MAX k-DOMINATING SET problem. It consists, given some graph, of determining a set V' of k vertices that dominate at least p vertices from $V \setminus V'$. Since MIN DOMINATING SET is known to be W[2]-hard, one can immediately derive that so is MAX k-DOMINATING SET with respect to k.

Revisit the well-known approximability-preserving reduction from MIN DOMINATING SET to MIN SET COVER [1]. For memory, this reduction works as follows: for each vertex v, build a set $S_v = N[v]$, where $N[v]$ is the closed neighbourhood of v. Then, covering p elements in the so-built instance of MAX k-SET COVER is equivalent to dominating p vertices in the instance of MAX k-DOMINATING SET. Finally remark that this reduction preserves values of k, p and Δ (indeed, it transforms Δ to $\Delta + 1$). It is easy to see that this reduction is simultaneously an FPT reduction too and that identically works when dealing with MAX k-DOMINATING SET and MAX k-SET COVER. Then, one can conclude that MAX k-DOMINATING SET parameterized either by $\max\{k, \Delta\}$, or by p is FPT, where for the MAX k-DOMINATING SET instance, Δ is the maximum graph-degree. Note that these results about MAX k-DOMINATING SET have already been obtained by using the random separation technique [5]. But use of our technique (greediness-for-parameterization) is less costly in time.

As we have seen above MAX k-SET COVER is hard when parameterized by k, $\max\{k, f\}$ and $\max\{\Delta, f\}$. But, in which class of the W[$\cdot$] hierarchy belongs this problem? We show in the next proposition that it belongs to W[P]. Inclusion of it in some "lower" class remains open.

Proposition 3. MAX k-SET COVER *parameterized by k belongs to W[P].*

Proof. The proof is in exactly the same spirit with the proofs in [7]. We reduce MAX k-SET COVER to BOUNDED NON-DETERMINISTIC TURING MACHINE COMPUTATION which is a known W[P]-hard problem [11]. Let $\mathcal{I} = (\mathcal{S} = \{S_1, \ldots, S_m\}, p)$ be an instance of MAX k-SET COVER. Build a 3-tapes Turing Machine M with 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, $O(|\mathcal{I}|^2)$). ∎

By similar argument one can derive that MAX k-DOMINATING SET, parameterized by k is also in W[P].

2.2 Approximation issues

Let us now say some words about parameterized approximation of MAX k-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 NP = NP [14]. We mainly prove in the sequel that it is quite unlikely that we are ever able to get such a ratio even in time parameterized by k. More precisely, we prove that if such a situation was possible, then we should be able to get, in parameterized time, an approximation ratio of $o(\log n)$ for MIN SET COVER, fact considered as highly improbable. We also prove that approximating MAX k-SET COVER within ratios better than $1 - (c/n)$ for any fixed constant $c > 1$, in time parameterized by k, is W[2]-hard. Note, finally, that MIN SET

COVER is inapproximable in polynomial time within ratio $(1-\varepsilon)\ln n$, for any $\varepsilon > 0$, unless NP $\subset$ DTIME($n^{O(\log\log n)}$) [14].

Proposition 4. *The following holds for* MAX k-SET COVER*:*

1. *Unless* MIN SET COVER *is approximable within ratio* $O(\log(n/\log n))$*, in time parameterized by the value of the optimum,* MAX k-SET COVER *parameterized by* k *is inapproximable within ratio* $(1-(1/e)+\epsilon)$*, for any* $\epsilon > 0$*, ;*

2. *it is W[2]-hard with respect to* k *to approximate* MAX k-SET COVER *within ratio* $1-(c/n)$*, for any constant* $c > 1$*.*

Proof. The basic idea of the result is similar to the basic idea in [14] (Proposition 5.2). Its fundamental ingredient is the following. Consider some algorithm `kSCALGORITHM` that solves MAX k-SET COVER . Then, it can iteratively be used to solve MIN SET COVER as follows. Iteratively run `kSCALGORITHM` for $k = 1, \ldots m$ (where m is the size of the set system in the MIN SET COVER-instance). One of these k's will correspond to 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 Algorithm `kSCALGORITHM` achieves approximation ratio r for MAX k-SET COVER. Call it with value k_0, (note that now $p = n$, the size of the ground set), remove the ground elements covered, store the k_0 elements used and repeatedly relaunch it with value k_0, until all ground elements are removed. Since it is assumed achieving 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):

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

for some rational $\beta/\alpha > 1$. Taking $r = 1-(1/e)-\epsilon$, (1) leads to $t \leqslant \ln n/(1-\ln(1-\epsilon e))$, contradicting so the inapproximability bound of [14].

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

Assume now that `kSCALGORITHM` is FPT in k and that achieves approximation ratio $1-(1/e)-\epsilon$ for some small ϵ.

Then, in order to prove item 1, fix some constant $\eta > 1$ and follow the procedure described above until there are at most $\ln^\eta n$ uncovered elements, stop it and solve the remaining instance by, say, the best known exact algorithm that works within $O^*(2^n)$ in instances with ground set-size n [2]. Since the surviving ground set has size $\log^\eta n$, it is polynomial to optimally solve it. After some easy algebra, one gets:

$$t \leqslant \frac{1}{1+\epsilon e}\ln\left(\frac{n}{c\ln n}\right)$$

that concludes the proof of item 1.

For item 2, just run kSCALGORITHM only once for any k. For k_0, such a run will leave uncovered at most $n - n(1 - (c/n)) = c$ elements. Any (non-trivial) cover for them uses at most c sets to cover them. In this case, the procedure above achieves an *additive* approximation ratio $c + 1$ (recall that c is fixed). Then, in order to complete the proof, revisit the well-known reduction from MIN DOMINATING SET to MIN SET COVER seen above and use the fact that, for the former problem, achievement of any constant additive approximation ratio is W[2]-hard [13]. ∎

For the rest of the section, we will relax the optimality requirement for the MAX k-SET COVER-solution and will show that we can devise an approximation schema in time parameterized by p, whose (parameterized) complexity is much lower (although depending on the accuracy) than the exact parameterized complexity given in Proposition 2.

Let us consider the following algorithm, called PSCSCHEMA in what follows:

1. fix some $\varepsilon > 0$, take $\epsilon \leqslant 1 - (\varepsilon/0.368)$ and set $k' = \epsilon k$;
2. run ALGORITHM1$(\Delta, \epsilon k')$, store the solution (denoted by $\mathcal{T}_1$) and remove from the ground set X the set X_1 of elements covered by $\mathcal{T}_1$;
3. set $\mathcal{S}' = \mathcal{S} \setminus \mathcal{T}_1$, $X' = X \setminus X_1$ and $k = (1 - \epsilon)k$;
4. run the polynomial approximation MAX k-SET COVER-algorithm of [14] on the MAX k-SET COVER-instance $(\mathcal{S}', X')$ and store the solution $\mathcal{T}_2$ computed;
5. output $\hat{\mathcal{T}} = \mathcal{T}_1 \cup \mathcal{T}_2$.

It is easy to see that solution$\hat{\mathcal{T}}$ computed by PSCSCHEMA has cardinality k, i.e., it is feasible for MAX k-SET COVER. We show in the following proposition that it is also a ε-approximation for this problem, for any $\varepsilon > 0$.

Proposition 5. MAX k-SET COVER *can be approximated within ratio* $1 - \varepsilon$*, for any* $\varepsilon > 0$*, in* $O^*(2^{(2-(\varepsilon/0.368))p} p^{\varepsilon p^2/0.368})$*-time.*

Proof. Fix an optimal solution $\mathcal{T}^*$ and denote by X_1^* the number of elements of X covered by the "best" ϵk sets of $\mathcal{T}^*$ and by X^* the subset of X covered by $\mathcal{T}^*$. Observe first that, by an easy average-argument it holds that:

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

Fix an optimal MAX $(1 - \epsilon)k$-SET COVER-solution of $(\mathcal{S}', X')$ and denote by $\bar{X}^*$ the set of elements of X covered by it and by X_2 the subset of X covered by $\mathcal{T}_2$. Obviously:

$$|X_2| \geqslant 0.632\,\left|\bar{X}^*\right| \tag{3}$$

where 0.632 is the third digit truncation of the quantity $(e - 1)/e$ that is the ratio achieved by the MAX k-SET COVER-algorithm in [14].

Set $\tilde{X} = X^* \cap X_1$ and remark that, the elements of $X^* \setminus \tilde{X}$ are still present in the instance $(\mathcal{S}', X')$ where the approximation algorithm of [14] is called, as well as the subsets of $\mathcal{S}$ covering them. Hence:

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

Putting together (2), (3) and (4), we get the following for the approximation ratio of Algorithm `PSCSCHEMA`:

$$\frac{|X_1| + |X_2|}{|X^*|} \geqslant \frac{|X_1^*| + 0.632\,(|X^*| - |X_1^*|)}{|X^*|} \geqslant 0.632 + 0.368\epsilon$$

This ratio is at least $1 - \varepsilon$, for $\epsilon \leqslant 1 - (\varepsilon/0.368)$.

For the overall running time, it suffices to observe that, since the algorithm of [14] runs in polynomial time, the running time of `PSCSCHEMA` is dominated by that of `ALGORITHM1` called in step 2. Then, setting ϵp instead of p in the exponent of 2 (in the complexity expression for `ALGORITHM1`) and ϵk instead of k in the exponent of Δ (in the same expression), derives the complexity claimed and completes the proof of the proposition. ∎

3 Satisfiability problems

As mentioned in Section 1, the status of MAX k-SET COVER to be W[2]-hard and in W[P] is shared by some other problems. We introduce handle another quite natural satisfiability problem, called MAX SAT-k, and prove that it behaves like MAX k-SET COVER, from a parameterized point of view.

Given a CNF on n variables and m clauses, MAX SAT-k consists of setting at most k variables to true, satisfying at least p clauses

Proposition 6. *MAX SAT-k is W[2]-hard for parameter k and in W[P].*

Proof. In order to prove hardness, we give an FPT reduction from MAX k-SET COVER to MAX SAT-k that works as follows. Let $(\mathcal{S} = \{S_1, \ldots, S_m\}, X)$ be an instance of MAX k-SET COVER. The instance of MAX SAT-k is built in such a way that set S_i is encoded by a variable X_i, and each element $x_j \in X$ is encoded by a clause $X_{i_1} \vee \ldots \vee X_{i_h}$ where $S_{i_1}, \ldots$, and S_{i_h} are the sets containing x_j. Then, one can see immediately that inclusion of MAX SAT-k in a lower W-class, would imply the same for MAX k-SET COVER.

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. ∎

In what follows, C denotes the set of clauses of an instance and C' any subset of this set. We denote by $\mathrm{occ}^+(X_i, C')$ the number of positive occurrences of the variable X_i in the instance, 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 $f = \max_i\{\mathrm{occ}^+(X_i, C) + \mathrm{occ}^-(X_i, C)\}$.

Proposition 7. MAX SAT-k *parameterized by* p *is FPT.*

Proof. Note first that instances such that $p < \frac{f}{2}$ are always YES-instances, since one can set one variable X_i with frequency f to true if $\text{occ}^+(X_i, C) \geqslant \text{occ}^-(X_i, C)$, and to false otherwise. and complete the solution arbitrarily. Note also that instances such that $p < k$ are all YES-instances, too. Indeed, iteratively one can 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. We may now assume that $p \geqslant \frac{f}{2}$ and $p \geqslant k$, so our parameter might as well be $p + f + k$.

Once again, we construct a branching algorithm which operates accordingly to a greedy criterion. A solution is given as a set S, of size up to k, containing all the variables set to true. Additionally, we maintain a list C_s of clauses that we satisfy or commit to satisfy. Set $d_i(C') = \text{occ}^+(X_i, C')$ and let $C^+(X_i, C')$ be the set of clauses in C' where X_i appears positively and $C^-(X_i, C')$ the set of clauses where X_i appears negatively. Set, finally, $C(X_i, C') = C^+(X_i, C') \cup C^-(X_i, C')$ and consider the following algorithm (`ALGORITHM2`):

- set $S = \emptyset$ and $C_s = \emptyset$;
- if $|S| < k$ and $|C_s| < p$ then:
  - set $C_u = C \setminus C_s$ and let X_i be a variable maximizing $d_i(C_u)$;
  - branch on `ALGORITHM2`$(S \cup \{X_i\}, C_s \cup C^+(X_i, C_u))$, and for each clause $c \in C(X_i, C_u)$, branch on `ALGORITHM2`$(S, C_s \cup \{c\})$;
- else:
  - if $|S| = k$, then store S as solution;
  - if $|C_s| \geqslant p$, then check if the solution can be extended to satisfy each clause of C_s;
- output the best among the solutions stored in the two previous steps.

The branching tree has depth at most $k+p$ and width at most $f+1$, so the running time of `ALGORITHM2` is $O^*(2^p f^{k+p})$ that is FPT with respect to p, since completing a solution to satisfy all the clauses of C_s can be done in time $O^*(2^{|C_s|})$, by simply solving a MIN HITTING SET, considering C_s as the ground set of the MIN HITTING SET-instance.

Let now S_0 be an optimal solution. From the root of the branching tree, follow a maximal branch where the variables set to true are all in S_0, and the clauses in C_s are satisfied by S_0. Let S_c be the set of variables set to true along this branch (by definition, $S_c \subseteq S_0$), and set $S_n = S_0 \setminus S_c$. By maximality of the branch, at its extremity, `ALGORITHM2` deviates from S_0, i.e., no child of the branching tree is conform to S_0. Let X_i be the variable chosen at this point by `ALGORITHM2` and consider $C_d = C(X_i, C_u)$ that is the set of clauses not yet in C_s and where X_i appears positively or negatively. We know that no clause in C_d is satisfied by S_0. Let X_j be any variable in S_n. We claim that

$S_h = (S_0 \setminus \{X_j\}) \cup \{X_i\}$ is also optimal and, by a straightforward induction, one solution at the leaves of the branching tree is as good as S_0. Indeed, setting X_j to false can lose at most $\mathrm{occ}^+(X_j, C_u) - \mathrm{occ}^-(X_j, C_u) \leqslant d_j(C_u)$ clauses and setting X_i to true gains $d_i(C_u)$ clauses and, by construction, $d_i(C_u) \geqslant d_j(C_u)$. ∎

We now show that MAX SAT-k, parameterized by $\max\{k, f\}$ is W[1]-hard. For this, we prove the following intermediate result.

Proposition 8. 3-SAT-k *parameterized by* $\max\{k, f\}$ *is W[1]-hard.*

Proof. Before giving our main reduction, we will first prove that MAX 3-SAT-k remains W[1]-hard when all variables occur the same number of times. To do so, let f be the frequency in the input formula. Then, we add some dummy clauses in order that all variables appear $f + 2$ times. For each variable x, we add a sufficient number of clauses $(x \vee \neg x)$ and, if $f - f(x) = 1 \bmod 2$, one clause $(x \vee \neg x \vee x)$ such that the frequency of x equals $f+2$. These clauses are trivially true, for any assignment of x and thus does not change the number of variables set to true in a solution. Since f is bounded by the number of clauses m, the size of the new formula is polynomially bounded with respect the size of the old formula.

Starting from an instance of MAX 3-SAT-k where all variables occur the same number f of times, we rewrite it in such a way that each variable appears at most three times, as in the classical polynomial reduction (see for example [19]). More precisely, for each variable x appearing f times in the formula, we introduce f new variables $x_1, x_2, \ldots, x_f$ and replace the first occurrence of x by x_1, the second by x_2, and so on. Then, to enforce that all these f variables get the same value, we add in the formula the clauses $(\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge \ldots \wedge (\neg x_f \vee x_1)$. The number of variables and clauses in this new formula is polynomially bounded by the number of variables and clauses in the original formula. It suffices now to prove that there exists a solution with kf variables set to true in this formula iff there is a solution with k variables set to true in the original formula. If the original formula is true with k variables set to true, then, for each true variable x, we put to true all variables $x_1, \ldots, x_f$, implying that kf variables are set to true. It is clear that the so-constructed formula is true too (the added clauses are true by x_i, if x is true, or by $\neg x_i$ if x was false, thus with 0 variables set to true). Conversely, we see that considering our added clauses, for a given variable x, all variables x_i, $1 \leq i \leq f$, must have the same value. The fact that MAX 3-SAT-k is W[1]-hard [18] concludes the proof. ∎

Then, the following result immediately holds.

Corollary 1. MAX 3-SAT-k *parameterized by* $\max\{k, f\}$ *is W[1]-hard.*

4 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 solu-

tion only satisfies the property a "sufficient number of times". In this sense, as mentioned MAX k-SET COVER where one has to cover at least p elements, generalizes MIN SET COVER, where all the elements must be covered. Similarly MAX k-VERTEX COVER where one has to find a minimum subset of vertices which covers at least p edges, generalizes MIN VERTEX COVER, where one has to cover all the edges; yet, MAX SAT generalizes SAT.

These partial problems come along with two parameters: the size of the solution, frequently denoted by k and the "sufficient number of times" quantified by p. These problems parameterized by k are shown to be either W[1]- or W[2]-hard, but we can not do better than showing their membership in W[P]. This is a quite important asymmetry between classical complexity theory as we know it from the literature (see, for example, [15, 19, 20]) and parameterized complexity theory.

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

A counting gate C_j with fan-in i and constant integer $j \in \{0, \ldots, i\}$ has fan-out 1 and outputs 1 iff exactly j of its i inputs are 1's. Note that C_i corresponds to an *and* gate. Also, the *or* gate can be emulated by a negation of a gate C_0. The negation is also a counting gate as a unitary C_0. A *counting* circuit is a circuit with some input gates and counting gates and exactly one output gate. Correspndingly, 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 height h and weft k, and $I \in \Pi$ iff $\mathcal{C}(I) = 1$. It can be immediately seen that the cweft hierarchy has exactly the same definition as the weft hierarchy up to replacing a circuit by a counting circuit.

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

Proposition 9. *The following inclusions hold for the counting weft hierarchy:* MAX k-SET COVER *and* MAX SAT-k *are in CW[2].*

5 Conclusion

He have studied the parameterized complexity of MAX k-SET COVER and MAX SAT-k with respect to pairs of natural instance parameters as k, Δ (the maximum set-cardinality) and f, the maximum frequency of the ground elements, for the former, or the number of variables set to true in a feasible assignment and the maximum number of occurrences of the variables in the input-formula, for

the latter. For this we have used the greediness-for-parameterization technique that seem to be very appropriate for this kind of multi-parameter studies. An important (and pleasant) corollary of the multiparameterization studied here for MAX k-SET COVER is that it has shown that this problem is FPT for its standard parameterization. We have also sketched an enhancement of the classical weft hierarchy that seems to be able to help us proving completeness for known W[·]-hard problems. Existence of such complete problems for the new hierarchy is, to our opinion, the major open problem drawn in this paper.

References

[1] R. Bar-Yehuda and S. Moran. On approximation problems related to the independent set and vertex cover problems. *Discrete Appl. Math.*, 9:1–10, 1984.

[2] A. Björklund, T. Husfeldt, and M. Koivisto. Set partitioning via inclusion-exclusion. *SIAM J. Comput.*, 39(2):546–563, 2009.

[3] E. Bonnet, B. Escoffier, V. Th. Paschos, and E. Tourniaire. Multi-parameter complexity analysis for constrained size graph problems: using greediness for parameterization. *CoRR*, abs/1306.2217, 2013.

[4] L. Cai. Parameter complexity of cardinality constrained optimization problems. *The Computer Journal*, 51:102–121, 2008.

[5] L. Cai, S. M. Chan, and S. O. Chan. Random separation: a new method for solving fixed-cardinality optimization 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 239–250. Springer-Verlag, 2006.

[6] 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.

[7] M. Cesati. Compendium of parameterized problems. `http://www.sprg.uniroma2.it/home/cesati/`.

[8] 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.

[9] 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.

[10] 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, *International Workshop on Parameterized and Exact Computation, IWPEC'04*, volume 3162 of *Lecture Notes in Computer Science*, pages 271–280. Springer-Verlag, 2004.

[11] R. G. Downey and M. R. Fellows. *Parameterized complexity.* Monographs in Computer Science. Springer, New York, 1999.

[12] 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.

[13] 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.

[14] U. Feige. A threshold of $\ln n$ for approximating set cover. *J. Assoc. Comput. Mach.*, 45:634–652, 1998.

[15] M. R. Garey and D. S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness.* W. H. Freeman, San Francisco, 1979.

[16] 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.

[17] D. Marx. Parameterized complexity and approximation algorithms. *The Computer Journal*, 51(1):60–78, 2008.

[18] R. Niedermeier. *Invitation to fixed-parameter algorithms.* Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford, 2006.

[19] C. H. Papadimitriou. *Computational complexity.* Addison-Wesley, 1994.

[20] C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization: algorithms and complexity.* Prentice Hall, New Jersey, 1981.