Parameterized complexity and kernelizability of Max Ones and Exact Ones problems*

Stefan Kratsch[1], Dániel Marx[2], and Magnus Wahlström[1]

[1] Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany
{skratsch,wahl}@mpi-inf.mpg.de
[2] Tel Aviv University, Israel
dmarx@cs.bme.hu

Abstract. For a finite set Γ of Boolean relations, Max Ones SAT(Γ) and Exact Ones SAT(Γ) are generalized satisfiability problems where every constraint relation is from Γ, and the task is to find a satisfying assignment with at least/exactly k variables set to 1, respectively. We study the parameterized complexity of these problems, including the question whether they admit polynomial kernels. For Max Ones SAT(Γ), we give a classification into 5 different complexity levels: polynomial-time solvable, admits a polynomial kernel, fixed-parameter tractable, solvable in polynomial time for fixed k, and NP-hard already for $k = 1$. For Exact Ones SAT(Γ), we refine the classification obtained earlier by having a closer look at the fixed-parameter tractable cases and classifying the sets Γ for which Exact Ones SAT(Γ) admits a polynomial kernel.

1 Introduction

The constraint satisfaction problem (CSP) provides a framework in which it is possible to express, in a natural way, many combinatorial problems encountered in artificial intelligence and computer science. A CSP instance is represented by a set of variables, a domain of values for each variable, and a set of constraints on the values that certain collections of variables can simultaneously take. The basic aim is then to find an assignment of values to the variables that satisfies the constraints. Boolean CSP (when all variables have domain $\{0, 1\}$) generalizes satisfiability problems such as 2SAT and 3SAT by allowing that constraints are given by arbitrary relations, not necessarily by clauses.

As Boolean CSP problems are NP-hard in general, there have been intensive efforts at finding efficiently solvable special cases of the general problem. One well-studied type of special cases is obtained by restricting the allowed constraint relations to a fixed set Γ; we denote by SAT(Γ) the resulting problem. We expect that if the relations in Γ are simple, then SAT(Γ) is easy to solve. For example, if Γ contains only binary relations, then SAT(Γ) is polynomial-time solvable, as it can be expressed by 2SAT. On the other hand, if Γ contains all the ternary relations, then SAT(Γ) is more general than 3SAT, and hence it is NP-hard.

* The second author is supported by ERC Advanced Grant DMMCA and the Hungarian National Research Fund (OTKA grant 67651).

A celebrated classical result of T.J. Schaefer [18] from 1978 characterizes the complexity of SAT(Γ) for *every* finite set Γ: it shows that if Γ has certain simple combinatorial properties, then SAT(Γ) is polynomial-time solvable, and if Γ does not have these properties, then SAT(Γ) is NP-hard. This result is surprising for two reasons. First, Ladner's Theorem [14] states that if P $\neq$ NP, then there are problems in NP that are neither in P nor NP-complete. Therefore, it is surprising that every SAT(Γ) problem is either in P or NP-complete, and no intermediate complexity appears for this family of problems. Second, it is surprising that the borderline between the P and NP-complete cases of SAT(Γ) can be conveniently characterized by simple combinatorial properties.

Schaefer's result has been generalized in various directions. Bulatov [3] generalized it from Boolean CSP to CSP over a 3-element domain and it is a major open question if it can be generalized to arbitrary finite domains (see [4, 10]). Creignou et al. [6] classified the polynomial-time solvable cases of the problem Exact Ones SAT(Γ), where the task is to find a satisfying assignment such that exactly k variables have value 1, for some integer k given in the input. Natural optimization variants of SAT(Γ) were considered in [5, 7, 12] with the goal of classifying the approximability of the different problems. In Max SAT(Γ) we have to find an assignment maximizing the number of satisfied constraints, while in Min UnSAT(Γ) we have to find an assignment minimizing the number of unsatified constraints. Min Ones SAT(Γ) and Max Ones SAT(Γ) ask for a satisfying assignment minimizing and maximizing, respectively, the number of variables having value 1.

Parameterized complexity. Recently, there have been investigations of the hardness of CSP from the viewpoint of parameterized complexity [15, 13]. This paradigm investigates hardness in finer detail than classical complexity, which focuses mostly on polynomial-time algorithms. A *parameterization* of a problem is assigning an integer k to each input instance. Consider, for example, two standard NP-complete problems Vertex Cover and Clique. Both have the natural parameter k: the size of the required vertex cover/clique. Both problems can be solved in time $n^{O(k)}$ on n-vertex graphs by complete enumeration. Notice that the degree of the polynomial grows with k, so the algorithm becomes useless for large graphs, even if k is as small as 10. However, Vertex Cover can be solved in time $O(2^k \cdot n^2)$ [11, 9]. In other words, for every fixed cover size there is a polynomial-time (in this case, quadratic in the number of vertices) algorithm solving the problem where the degree of the polynomial is independent of the parameter. Problems with this property are called *fixed-parameter tractable.* The notion of W[1]-hardness in parameterized complexity is analogous to NP-completeness in classical complexity. Problems that are shown to be W[1]-hard, such as Clique [11, 9], are very unlikely to be fixed-parameter tractable.

Kernelization. One of the most basic techniques for showing that a problem is fixed-parameter tractable is to show that the computationally hard "core" of the problem can be extracted in polynomial time. Formally, *kernelization* is a polynomial-time transformation that, given an instance I of problem P with parameter k, creates an equivalent instance I' of problem P with parameter $k' \leq f(k)$ such that the size of I' is at most $g(k)$ for some functions f, g (usually, $k' \leq k$ is achievable). For example, a classical result of Nemhauser and Trotter [16] shows that every instance I of Vertex Cover with parameter k can be transformed into an instance I' with parameter $k' \leq k$ such that I' has at most $g(k) = 2k$ vertices. Observe that the existence of a kernelization algorithm for P immediately implies that P is FPT, assuming that P is decidable: performing the kernelization and then doing a brute force solution on I' clearly takes only $n^{O(1)} + f(k)$ time for some function f. From the practical point of view, *polynomial kernels,* i.e., kernelization algorithms where $g(k)$ is a polynomial, are of particular interest. If a problem has this property, then this means that there is an efficient preprocessing algorithm for the problem with a provable bound on the way it shrinks the instance. Such a preprocessing can be an invaluable opening step in any practical solution for the problem. Very recently, however, it has been shown that under standard complexity assumptions, not every FPT problem has a polynomial kernel: e.g., the k-Path problem can be solved in (randomized) time $2^k \cdot n^{O(1)}$ [19], but has no polynomial kernel unless NP $\subseteq$ co-NP/poly [1]. The negative toolkit developed in [1] has been successfully applied to a number of other problems [2, 8].

Results. The parameterized complexity of Exact Ones SAT(Γ) was studied in [15], where it was shown that a property called weak separability characterizes the complexity of the problem: Exact Ones SAT(Γ) is FPT if Γ is weakly separable, and W[1]-complete otherwise. The problem Min Ones SAT(Γ) is FPT for every Γ by a simple branching algorithm, but it is not obvious to see for which Γ there is a polynomial kernel. This question has been resolved in [13] by showing that (unless NP $\subseteq$ co-NP/poly) Min Ones SAT(Γ) has a polynomial kernel if and only if Min Ones SAT(Γ) is in P or Γ has a property called mergebility.

We continue this line of research by considering the so far unexplored problem Max Ones SAT(Γ) and revisit Exact Ones SAT(Γ). We will characterize (under standard complexity assumptions) parameterized Max Ones SAT(Γ) problems for finite constraint languages Γ as the following 5 types: solvable in polynomial time; NP-hard, but having polynomial kernelization; being FPT but admitting no polynomial kernelization; being W[1]-hard and in XP; and not being in XP. The characterization uses results of Nordh and Zanuttini [17] on frozen co-clones. For Exact Ones SAT(Γ), we refine the classification of [15] by precisely characterizing those weakly separable sets Γ for which Exact Ones SAT(Γ) is not only FPT, but admits a polynomial kernel. Table 1 shows some examples.

The kernelization lower bounds for both problems use reductions from a maximization problem MULTIPLE COMPATIBLE PATTERNS, which is FPT but admits no polynomial kernelization unless NP $\subseteq$ co-NP/poly. This problem may be useful for other hardness reductions as well.

2 Preliminaries and Notation

Boolean CSP. A *formula* ϕ is a pair (V, C) consisting of a set V of *variables* and a set C of *constraints.* Each constraint $c_i \in C$ is a pair $\langle \overline{s}_i, R_i \rangle$, where

Γ	Min Ones	Exact Ones	Max Ones
width-2 affine	P	P	P
$\{\mathrm{ODD}_3\}$	PK	PK	P
$\{\mathrm{EVEN}_3\}$	P	FPT	PK
$\{\mathrm{EVEN}_3, (x)\}$	FPT	FPT	PK
$\{\mathrm{ODD}_4\}$, general affine	FPT	FPT	PK
$\{(x \vee y), (x \neq y)\}$	PK	PK	PK
$\{((x \rightarrow y) \wedge (y \neq z))\}$	PK	FPT	FPT
$\{(x \vee y), (x \neq y), (x \rightarrow y)\}$	PK	W[1]-complete	FPT
bijunctive	PK	W[1]-complete	W[1]-hard, XP
$\{R_{\text{1-IN-3}}\}$	PK	PK	not in XP
$\{\sum_i x_i = p \pmod q\}$	FPT	FPT	not in XP
general	FPT	W[1]	not in XP

Table 1. Examples of sets of relations Γ and the properties for Min Ones SAT(Γ), Exact Ones SAT(Γ), and Max Ones SAT(Γ). Problems marked PK have polynomial kernels; problems marked FPT are FPT but admit no polynomial kernelization unless NP $\subseteq$ co-NP/poly.

$\overline{s}_i = (x_{i,1}, \ldots, x_{i,r_i})$ is an r_i-tuple of variables (the *constraint scope*) and $R_i \subseteq \{0,1\}^{r_i}$ is an r_i-ary Boolean relation (the *constraint relation*). A function $f : V \rightarrow \{0,1\}$ is a *satisfying assignment* of ϕ if $(f(x_{i,1}), \ldots, f(x_{i,r_i}))$ is in R_i for every $c_i \in C$. Let Γ be a set of Boolean relations. A formula is a *Γ-formula* if every constraint relation R_i is in Γ. In this paper, Γ is always a finite set containing only non-empty relations. For a fixed finite Γ, every Γ-formula $\phi = (V, C)$ can be represented with length polynomial in $|V|$ and $|C|$: each constraint relation can be represented by constant number of bits (depending only on Γ). The *weight* $w(f)$ of an assignment f is the number of variables x with $f(x) = 1$.

We also use some definitions from [17]. Let $\phi = (V, C)$ be a formula and $x \in V$ a variable. Then x is said to be *frozen* in ϕ if x takes the same value in every satisfying assignment of ϕ. Further, let Γ be a set of relations, and R an n-ary relation. Then Γ *freezingly implements* R if there is a formula ϕ over $\Gamma \cup \{=\}$ such that $R(x_1, \ldots, x_n) \equiv \exists X \phi$, where ϕ uses variables $X \cup \{x_1, \ldots, x_n\}$ only, and all variables in X are frozen in ϕ. If only relations of Γ are used, then we have a *frozen implementation without equality.* This will be our standard notion of implementation in the paper, and as such is shortened to simply "implements".

We recall some standard definitions concerning Boolean constraints (cf. [5]):

- R is *0-valid* if $(0, \ldots, 0) \in R$.
- R is *1-valid* if $(1, \ldots, 1) \in R$.
- R is *Horn* or *weakly negative* if it can be expressed as a conjunction of clauses such that each clause contains at most one positive literal. It is known that R is Horn if and only if it is *AND-closed:* if $(a_1, \ldots, a_r) \in R$ and $(b_1, \ldots, b_r) \in R$, then $((a_1 \wedge b_1), \ldots, (a_r \wedge b_r)) \in R$.
- R is *anti-Horn* or *weakly positive* if it can be expressed as the conjunction of clauses such that each clause contains at most one negated literal. It is

known that R is anti-Horn if and only if it is *OR-closed:* if $(a_1, \dots, a_r) \in R$ and $(b_1, \dots, b_r) \in R$, then $((a_1 \vee b_1), \dots, (a_r \vee b_r)) \in R$.
- R is *bijunctive* if it can be expressed as the conjunction of constraint such that each constraint is the disjunction of two literals.
- R is *affine* if it can be expressed as a conjunction of constraints of the form $x_1 + x_2 + \cdots + x_t = b$, where $b \in \{0, 1\}$ and addition is modulo 2. The number of tuples in an affine relation is always an integer power of 2. We denote by EVEN_r the r-ary relation $x_1 + x_2 + \cdots + x_r = 0$ and by ODD_r the r-ary relation $x_1 + x_2 + \cdots + x_r = 1$.
- R is *width-2 affine* if it can be expressed as a conjunction of constraints of the form $x = y$, $x \neq y$, (x), and $(\neg x)$.
- R is *monotone* if $a \in R$ and $b \geq a$ implies $b \in R$, where $\geq$ is applied component-wise. Such a relation is implementable by positive clauses, by adding a clause over the false positions of every maximal false tuple.
- The relation $R_{p\text{-IN-}q}$ (for $1 \leq p \leq q$) has arity q and $R_{p\text{-IN-}q}(x_1, \dots, x_q)$ is true if and only if exactly p of the variables $x_1, \dots, x_q$ have value 1.

The above is extended to properties of sets of relations, by saying that a set of relations Γ is 0-valid (1-valid, Horn, ...) if this holds for every $R \in \Gamma$.

Theorem 1 (Schaefer [18]). *Let Γ be a set of Boolean relations. Then SAT(Γ) is in* P *if Γ has one of the following properties: 0-valid, 1-valid, Horn, anti-Horn, bijunctive, or affine. Otherwise, SAT(Γ) is* NP*-complete.*

Max Ones SAT(Γ) and Exact Ones SAT(Γ). For a fixed set of relations Γ, Max Ones SAT(Γ) is the following problem:

> **Input:** A formula ϕ over Γ; an integer k.
> **Parameter:** k.
> **Task:** Decide whether there is a satisfying assignment for ϕ of weight at least k.

For example, Max Ones SAT($\neg x \vee \neg y$) is equivalent to Independent Set, and is thus W[1]-complete. Further examples can be found in Table 1. Similarly, Exact Ones SAT(Γ), for a fixed set of relations Γ, is the following problem.

> **Input:** A formula ϕ over Γ; an integer k.
> **Parameter:** k.
> **Task:** Decide whether there is a satisfying assignment for ϕ of weight exactly k.

Parameterized complexity and kernelization. A parameterized problem $\mathcal{Q}$ is a subset of $\Sigma^* \times \mathbb{N}$; the second component is called the *parameter*. The problem $\mathcal{Q}$ is *fixed-parameter tractable* (FPT) if there is an algorithm that decides $(I, k) \in \mathcal{Q}$ in time $f(k) \cdot n^{O(1)}$, where f is some computable function. A *kernelization* is a polynomial-time mapping $K : (I, k) \mapsto (I', k')$ such that (I, k) and (I', k') are equivalent, $k' \leq f(k)$, and $|I'| \leq g(k)$, for some functions f and g. Usually, f can be taken as the identity function, i.e., $k' \leq k$; this will be the case throughout this paper. If $|I'|$ is bounded by a polynomial in k, then K is a *polynomial kernelization*. It is well-known that every decidable parameterized problem is fixed-parameter tractable if and only if it has a (not necessarily polynomial) kernelization [11]. A *polynomial time and parameter reduction* from $\mathcal{Q}$ to $\mathcal{Q}'$ is a polynomial-time mapping $\Phi : (I, k) \mapsto (I', k')$ such that $(I, k) \in \mathcal{Q}$ if and only if $(I', k') \in \mathcal{Q}$ and such that k' is polynomially bounded in k; we denote the existence of such a reduction by $\mathcal{Q} \leq_{Ptp} \mathcal{Q}'$. These reductions were introduced by Bodlaender et al. [2], who also showed that they preserve polynomial kernelizability.

The MCP problem. Our kernelization lower bounds will use the problem Multiple Compatible Patterns (MCP), defined as follows:

> **Input:** A set of patterns from $\{0, 1, \bigstar\}^r$, where $\bigstar$ (the wildcard character) matches 0 or 1; an integer k.
> **Parameter:** $r + k$.
> **Task:** Decide whether there is a string in $\{0, 1\}^r$ that matches at least k patterns.

A kernelization lower bound for MCP follows from the methods of [1]. Briefly, we get NP-completeness by a reduction from Clique, and compositionality by adding $\log t$ bits to compose t instances.

Lemma 2. Multiple Compatible Patterns *(MCP) is* FPT*,* NP*-complete, and admits no polynomial kernelization unless* NP $\subseteq$ co-NP/poly*.*

3 Max Ones Characterization

This section contains the our characterization of the parameterized complexity properties of Max Ones SAT(Γ) problems.

As a very first distinction, observe that if SAT(Γ) is NP-complete, then Max Ones SAT(Γ) is NP-complete even for a parameter $k = 0$. Thus, we know by Schaefer (Theorem 1) that Γ has to fall in one of the classes 0-valid, 1-valid, affine, Horn, anti-Horn, or bijunctive for the problem to be in XP. Of these, the classes of 1-valid relations and anti-Horn relations are polynomial-time solvable, leaving four classes to examine. The cases of affine, Horn, and 0-valid relations can be characterized without too much difficulty, and will be treated summarily, as we will focus on the more interesting cases that occur when Γ is bijunctive.

We begin with the polynomial cases, as proven by Khanna et al. [12].

Theorem 3 ([12]). Max Ones SAT(Γ) *is in* P *if* Γ *is 1-valid, weakly positive (i.e. anti-Horn), or width-2 affine, and APX-hard in all other cases.*

The following lemma covers the properties of every set of relations Γ except the bijunctive cases; full proofs will be found in the full version.

Lemma 4. *Let* Γ *be a set of relations; the following hold.*

1. *If* Γ *is affine, then* Max Ones SAT(Γ) *has a kernel with* $O(k)$ *variables.*

2. *If Γ is Horn, but not anti-Horn and not 1-valid, then* Max Ones SAT(Γ) *is W[1]-hard.*
3. *If Γ is 0-valid, but neither anti-Horn, 1-valid, affine, nor Horn, then* Max Ones SAT(Γ) *is NP-hard for $k = 1$.*

Proof (sketches). 1. We can check in polynomial time which variables have to be set to false in every solution, and remove these. For the rest, we can by a greedy procedure find a solution which sets at least half the remaining variables to be true. Thus, we either find a solution with weight at least k, or leave a kernel with at most $2k$ variables.

2. If Γ is Horn, and the listed cases do not apply, then Γ admits a reduction from Independent Set by implementing $(\neg x \vee \neg y)$; either directly, or, e.g., via relations $(x \wedge y \rightarrow z)$ and $(\neg z)$.

3. Let Γ be 0-valid such that no listed case applies. It can be shown that Γ implements $R(x, y, z) = \{(0, 0, 0), (1, 1, 0), (1, 0, 1)\}$; we will show that Max Ones SAT(R) is NP-hard for $k = 1$. By a trick of splitting variables, we can adjust a given formula to add a universal variable z_1 such that $z_1 = 1$ in any solution where at least one variable is true. Relations $R(z_1, x, y)$ then become $(x \neq y)$ in any such solution, effectively constructing a reduction from SAT($R, (x \neq y)$). By Theorem 1, this problem is NP-complete, and the claim follows. □

3.1 Bijunctive cases

In this subsection we treat the cases of Max Ones SAT(Γ) where Γ is bijunctive but not Horn, anti-Horn, or width-2 affine (or 0-valid, or 1-valid, but this follows implicitly). This corresponds to the sets Γ which, using existentially quantified variables, can implement all 2SAT clauses; see [17]. See also Table 1 for a summary of the maximal cases.

For the result, we will need the results of Nordh and Zanuttini [17]. Recall the definition of a frozen implementation (with equality). The *frozen partial co-clone* $\langle \Gamma \rangle_{fr}$ generated by Γ is the set of all relations that can be freezingly implemented by Γ. We will use the characterization of [17] of the frozen partial co-clones that our Γ can generate. The free use of equality constraints is somewhat more general than what we wish to allow, but we will find that it causes no problems.

We need the following special cases.

1. $\Gamma_2^{p\neq} = \{(x \vee y), (x \neq y)\}$
2. $R_3^n = (\neg x \vee \neg y) \wedge (x \neq z)$; $\Gamma_3^n = \{R_3^n\}$
3. $\Gamma_2^{p\neq i} = \{(x \vee y), (x \neq y), (x \rightarrow y)\}$

Finally, we need a technical lemma to show that we can assume that we have access to the constants. We refer to the full version for a proof.

Lemma 5. *If Γ is neither 0-valid, 1-valid, nor affine, but SAT(Γ) is not NP-hard, then the constants can be implemented.*

Let us now proceed with settling the remaining cases of Max Ones SAT(Γ).

Lemma 6. *Assume that Γ is bijunctive but not Horn or anti-Horn. Then the following hold.*

1. *If $\Gamma \subseteq \langle \Gamma_2^{p\neq} \rangle_{fr}$, where $\Gamma_2^{p\neq} = \{(x \vee y), (x \neq y)\}$, then* Max Ones SAT($\Gamma$) *has a polynomial kernel (of $O(k^2)$ variables). Otherwise,* Max Ones SAT(Γ) *admits no polynomial kernelization, unless* NP $\subseteq$ co-NP/poly*.*
2. *If $\Gamma \subseteq \langle \Gamma_2^{p\neq i} \rangle_{fr}$, where $\Gamma_2^{p\neq i} = \{(x \vee y), (x \neq y), (x \rightarrow y)\}$, then* Max Ones SAT($\Gamma$) *is FPT (with running time $O^*(2^k)$). Otherwise,* Max Ones SAT(Γ) *is W[1]-hard.*

Proof. Let (ϕ, k) be a Max Ones SAT(Γ) instance. Assume throughout that the instance is feasible (as otherwise, the problem is trivial). We split the proof into proofs of feasibility (1a, 2a), and lower bound proofs (1b, 2b).

1a. By [17], every relation in Γ, and thus all of ϕ, has a frozen implementation over $\Gamma_2^{p\neq} \cup \{=\}$. We will refer to this implementation when inferring a kernel, but the kernelization will apply for the original Γ as well. Let a set of at least two variables which are connected by disequality or equality, with at least one disequality, be referred to as a *class* of variables. If there are at least k variable classes, then any solution will contain at least k true variables, and can be found in polynomial time. If any class contains at least $2k$ variables, then either the variables of this class have fixed values, in which case we make the corresponding assignments, or we can find a solution with at least k true variables. Finally, if any variable does not occur in a variable class, it can safely be set to 1. These observations leave a kernel with $O(k)$ variable classes and $O(k^2)$ variables in total. Finally, as the only changes we made to the formula were assignments, we can apply the kernelization using only relations in Γ by replacing all assigned variables by the constant variables z_1 or z_0.

1b. By [17], there is an implementation of R_3^n over $\Gamma \cup \{=\}$. As the equality constraint will not be useful in such an implementation, there is also an implementation directly over Γ, showing Max Ones SAT(Γ_3^n) $\leq_{Ptp}$ Max Ones SAT(Γ); we will in turn show that MCP $\leq_{Ptp}$ Max Ones SAT(Γ_3^n) (the problem MCP was defined in Section 2).

Observe that, renaming variables, R_3^n can be written as $(x \neq y) \wedge (z \rightarrow x)$. Let (I, k) be an instance of MCP, with string length r. Create variables $(x_i \neq y_i)$ for $1 \leq i \leq r$, coding the entries of the string; these variables contribute weight exactly r to any solution. Now for every pattern i, create a variable z_i, and for every position j of pattern i containing 0, add a constraint $(x_j \neq y_j) \wedge (z_i \rightarrow x_j)$. For positions containing 1, create the same constraint with an implication instead to y_j. Any solution with $r + k$ true variables corresponds one-to-one to a string in $\{0, 1\}^r$ and k patterns matching it. Thus (by [2]), Max Ones SAT(Γ) admits no polynomial kernelization unless NP $\subseteq$ co-NP/poly.

2a. As before, there is an implementation of ϕ over $\Gamma_2^{p\neq i} \cup \{=\}$. Again consider the variable classes; if they number at least k, then find a solution in polynomial time. Otherwise, we check all $O(2^k)$ assignments to variables of the variable classes. For each such assignment, propagate assignments to the remaining variables. Any formula that remains after this is 1-valid.

2b. By [17], there is an implementation of $(\neg x \vee \neg y)$ over $\Gamma \cup \{=\}$, and again the equality constraint would not be useful. Thus there is an FPT reduction from Independent Set to Max Ones SAT(Γ).

Our results for Max Ones SAT(Γ) summarize into the following picture.

Theorem 7. *Let Γ be a finite set of Boolean relations. Then* Max Ones SAT(Γ) *falls into one of the following cases.*

1. *If Γ is 1-valid, anti-Horn, or width-2 affine, then* Max Ones SAT(Γ) *is in* P*; in the remaining cases, it is* NP*-complete.*
2. *If Γ is affine, or if $\Gamma \subseteq \langle (x \vee y), (x \neq y) \rangle_{fr}$, then* Max Ones SAT($\Gamma$) *has a polynomial kernel.*
3. *If $\Gamma \subseteq \langle (x \vee y), (x \neq y), (x \rightarrow y) \rangle_{fr}$, then* Max Ones SAT($\Gamma$) *is in FPT, with a running time of $O^*(2^k)$, but if the previous case does not apply, then there is no polynomial kernelization unless* NP $\subseteq$ co-NP/poly.
4. *If none of these cases applies, then* Max Ones SAT(Γ) *is W[1]-hard; if Γ is Horn or bijunctive, then* Max Ones SAT(Γ) *is in XP.*
5. *Otherwise* Max Ones SAT(Γ) *is* NP*-complete for $k = 1$.*

4 Exact Ones CSP

In this section we classify Exact Ones SAT(Γ) into admitting or not admitting a polynomial kernelization depending on the set of allowed relations Γ. We start from the characterization of its fixed-parameter tractability [15] as well as the characterization of when Min Ones SAT(Γ) admits a polynomial kernelization [13]. To this end we recall the invariants called weak separability and mergeability used for the respective characterization. We also introduce a joined, stronger version of the two partial polymorphisms defining weak separability; this will be used to characterize kernelizability of Exact Ones SAT(Γ).

Definition 8. *A t-ary* partial polymorphism *is a partially defined function $f : \{0,1\}^t \rightarrow \{0,1\}$. For an r-ary relation R, we say that R is* invariant under f *if for any t tuples $\alpha_1, \ldots, \alpha_t \in R$, such that $f(\alpha_1(i), \ldots, \alpha_t(i))$ is defined for every $i \in [r]$, we have $(f(\alpha_1(1), \ldots, \alpha_t(1)), \ldots, f(\alpha_1(r), \ldots, \alpha_t(r))) \in R$.*

We present partial polymorphisms in a matrix form, where the columns represent the tuples for which f is defined, and the value below the horizontal line is the corresponding value of f.

Definition 9 ([15, 13]). *Let FPT(1), FPT(2), and FPT($1 \bowtie 2$) denote the following partial polymorphisms:*

FPT(1)	FPT(2)	FPT($1 \bowtie 2$)
0 0 0 1	0 1 0 1	0 1 0 0 1
0 0 1 1	0 1 1 1	0 1 0 1 1
0 1 0 1	0 0 0 1	0 0 1 0 1
0 1 1 1	0 0 1 1	0 0 1 1 1

A boolean relation R is weakly separable *if it is invariant under* FPT(1) and FPT(2)*. It is* semi-separable *if it is invariant under* FPT($1 \bowtie 2$)*. Finally, a relation is* mergeable *if it is invariant under the following partial polymorphism:*

Mergeable		
0	1 0 1 1 0	1
0	1 0 0 0 0	1
0	0 1 1 0 1	1
0	0 1 0 0 0	1
0	1 0 1 0 0	1

Theorem 10 ([15]). Exact Ones SAT(Γ) *is fixed-parameter tractable if every relation $R \in \Gamma$ is weakly separable. In the remaining cases it is* W[1]-complete*.*

Since any kernelization for a problem also implies fixed-parameter tractability, we will only need to further classify the fixed-parameter tractable cases.

By a simple observation, Min Ones SAT(Γ) reduces to Exact Ones SAT(Γ) by a polynomial time and parameter reduction. This allows us to transfer lower bounds from the min ones to the exact ones setting.

Lemma 11. Min Ones SAT(Γ) *reduces to* Exact Ones SAT(Γ) *by a polynomial time and parameter reduction.*

Thus, using the kernelization dichotomy for Min Ones SAT(Γ) [13], we may exclude further cases.

Theorem 12 ([13]). *Unless* NP $\subseteq$ co-NP/poly*,* Min Ones SAT(Γ) *admits a polynomial kernel if and only if every relation in Γ is mergeable or* Min Ones SAT(Γ) *is in* P*.*

Corollary 13. *If Γ is not mergeable and* Min Ones SAT(Γ) *is* NP*-hard then* Exact Ones SAT(Γ) *does not admit a polynomial kernel unless the polynomial hierarchy collapses.*

According to Khanna et al. [12] Min Ones SAT(Γ) is in P when Γ is 0-valid, weakly negative, or width-2 affine; in all other cases it is NP-hard (APX-hard).

Theorem 14. *Let Γ be a finite set of weakly separable relations.*

1. *If Γ is width-2 affine then* Exact Ones SAT(Γ) *is in* P*; this includes the cases where Γ is Horn, or both 0-valid and mergeable. In the remaining cases, the problem is* NP*-complete.*
2. *If Γ is anti-Horn, or both mergeable and semi-separable, then* Exact Ones SAT(Γ) *admits a polynomial kernelization.*
3. *In all other cases* Exact Ones SAT(Γ) *does not admit a polynomial kernelization unless* NP $\subseteq$ co-NP/poly*.*

We only give an outline of the proof; the full proof will be given in the full version.

Proof (outline). We first consider the cases when Min Ones SAT(Γ) is in P, i.e., when Γ is zero-valid, Horn, or width-2 affine [12]. In all other cases, due to Corollary 13, we may then use that Γ is mergeable (since otherwise Exact Ones SAT(Γ) does not admit a polynomial kernel).

If Γ is width-2 affine, then by Creignou et al. [6], Exact Ones SAT(Γ) is in P; otherwise it is NP-complete. *If Γ is Horn* we show that it can be implemented by $\{=, (x), (\neg x)\}$ and Exact Ones SAT(Γ) is in P. The same is true *if Γ is zero-valid and mergeable.*

If Γ is zero-valid but not mergeable then (unless Γ is Horn) we are able to reduce Exact Ones SAT(Γ') to Exact Ones SAT(Γ) where $\Gamma' = \Gamma \cup \{(x), (\neg x)\}$ by a polynomial time and parameter reduction. Since Γ' is neither zero-valid, Horn, nor width-2 affine we conclude that Min Ones SAT(Γ') is NP-hard. This implies that Exact Ones SAT(Γ') does not admit a polynomial kernelization by Corollary 13, which extends also to Exact Ones SAT(Γ) through our reduction.

For all further choices of Γ (i.e., neither zero-valid, Horn, nor width-2 affine) we have that Min Ones SAT(Γ) and Exact Ones SAT(Γ) are NP-hard. Therefore, by Corollary 13, we assume that Γ is mergeable.

If Γ is anti-Horn (and weakly separable) we show that it can be implemented by equality, negative assignments, and positive clauses. This also means that Γ is semi-separable and mergeable. Now one of two cases applies. If Γ is monotone, then Exact Ones SAT(Γ) reduces to d-Hitting Set and we are done. Otherwise, Exact Ones SAT($\Gamma \cup \{(x), (\neg x)\}$) reduces to Exact Ones SAT(Γ), implying that we have constants available. We will later show that for any semi-separable and mergeable Γ that contains (x) and $(\neg x)$ Exact Ones SAT(Γ) admits a polynomial kernel.

Otherwise, in particular, if Γ is not Horn or anti-Horn, we show that Exact Ones SAT($\Gamma \cup \{\neq, (x), (\neg x)\}$) reduces to Exact Ones SAT(Γ) by a polynomial time and parameter reduction; i.e., as above we may assume to have disequality and constants available in Γ. Then if Γ is not semi-separable, we show that Exact Ones SAT(Γ) does not admit a polynomial kernel by a polynomial time and parameter reduction from the MCP problem: The central fact is that we must have a witness against semi-separability (i.e., invariant under FPT($1 \bowtie 2$)), but all relations in Γ are weakly separable (i.e., invariant under FPT(1) and FPT(2)). Using disequality this witness permits us to implement $(x \to y) \wedge (y \neq z)$; we then use the reduction from MCP as in Lemma 6.

To conclude our proof it now suffices to give a polynomial kernelization for the case that Γ is mergeable, semi-separable, and contains positive and negative assignments. To this end we use a sunflower lemma for tuples to repeatedly find and simplify sunflowers while there are too many non-zero-valid constraints. The crucial part is that semi-separability allows us to essentially split constraints that form a sunflower into a core constraint and independent petal constraints: The core assignment and the petal assignment are independent for all feasible assignments to the core variables. Mergeability of Γ restricts zero-valid constraints to be implementable by equality and assignments, which can be handled in a straightforward way. □

Corollary 15. *Let Γ be a finite set of relations. Then* Exact Ones SAT(Γ) *is FPT if and only if Γ is weakly separable, unless FPT = W[1]; and admits a polynomial kernel if and only if Γ is semi-separable and mergeable, unless* NP $\subseteq$ co-NP/poly*.*

References

1. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels (extended abstract). In *ICALP (1)*, volume 5125 of *LNCS*, pages 563–574. Springer, 2008.
2. H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. In *ESA*, volume 5757 of *LNCS*, pages 635–646. Springer, 2009.
3. A. A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proc. 43th Symp. Foundations of Computer Science*, pages 649–658. IEEE, 2002.
4. A. A. Bulatov. Tractable conservative constraint satisfaction problems. In *LICS*, page 321. IEEE, 2003.
5. N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. 2001.
6. N. Creignou, H. Schnoor, and I. Schnoor. Non-uniform boolean constraint satisfaction problems with cardinality constraint. In *CSL*, volume 5213 of *LNCS*, pages 109–123. Springer, 2008.
7. P. Crescenzi and G. Rossi. On the Hamming distance of constraint satisfaction problems. *Theoretical Computer Science*, 288(1):85–100, 2002.
8. M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and ids. In *ICALP (1)*, volume 5555 of *LNCS*, pages 378–389. Springer, 2009.
9. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer, New York, 1999.
10. T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1999.
11. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
12. S. Khanna, M. Sudan, L. Trevisan, and D. P. Williamson. The approximability of constraint satisfaction problems. *SIAM J. Comput.*, 30(6):1863–1920, 2000.
13. S. Kratsch and M. Wahlström. Preprocessing of min ones problems: A dichotomy. In *ICALP*, 2010. To appear.
14. R. E. Ladner. On the structure of polynomial time reducibility. *J. Assoc. Comput. Mach.*, 22:155–171, 1975.
15. D. Marx. Parameterized complexity of constraint satisfaction problems. *Computational Complexity*, 14(2):153–183, 2005.
16. G. L. Nemhauser and L. E. Trotter, Jr. Vertex packings: structural properties and algorithms. *Math. Programming*, 8:232–248, 1975.
17. G. Nordh and B. Zanuttini. Frozen boolean partial co-clones. In *ISMVL*, pages 120–125, 2009.
18. T. J. Schaefer. The complexity of satisfiability problems. In *STOC*, pages 216–226. ACM, 1978.
19. R. Williams. Finding paths of length k in $O^*(2^k)$ time. *Inf. Process. Lett.*, 109(6):315–318, 2009.