Sparsification of SAT and CSP Problems via Tractable Extensions[*]

Victor Lagerkvist[‡] Magnus Wahlström[§]

March 18, 2020

Abstract

Unlike polynomial kernelization in general, for which many non-trivial results and methods exist, only few non-trival algorithms are known for polynomial-time sparsification. Furthermore, excepting problems on restricted inputs (such as graph problems on planar graphs), most such results rely upon encoding the instance as a system of bounded-degree polynomial equations. In particular, for SAT problems with a fixed constraint language Γ, every previously known result is captured by this approach, and for several such problems this is known to be tight. In this work, we investigate the limits of this approach – in particular, does it really cover all cases of non-trivial polynomial-time sparsification?

We generalize the method using tools from the algebraic approach to constraint satisfaction problems (CSP). Every constraint which can be modelled via a system of linear equations, over some finite field F, also admits a finite domain extension to a tractable CSP with a Maltsev polymorphism, and using known algorithms for Maltsev languages we can show that every problem of the latter type admits a "basis" of $O(n)$ constraints, which implies a linear sparsification for the original problem. This generalization appears to be strict; other special cases include constraints modelled via group equations over some finite group G. For sparsifications of polynomial but super-linear size we consider two extensions of this. Most directly, we can capture systems of bounded-degree polynomial equations in a "lift-and-project" manner, by finding Maltsev extensions for constraints over c-tuples of variables, for a basis with $O(n^c)$ constraints. Additionally, we may use extensions with k-edge polymorphisms instead of requiring a Maltsev polymorphism.

We also investigate characterizations of when such extensions exist. We give an infinite sequence of partial polymorphisms ϕ_1, ϕ_2, ... which characterizes whether a language Γ has a Maltsev extension (of possibly infinite domain). In the complementary direction of proving lower bounds on kernelizability, we prove that for any language not preserved by ϕ_1, the corresponding SAT problem does not admit a kernel of size $O(n^{2-\varepsilon})$ for any $\varepsilon > 0$ unless the polynomial hierarchy collapses.

1 Introduction

Kernelization is a preprocessing technique based on reducing an instance of a computationally hard problem in polynomial time to an equivalent instance, a *kernel*, whose size is bounded by a function f with respect to a given complexity parameter. The function f is referred to as the *size* of the kernel, and if the size is polynomially bounded we say that the problem admits a *polynomial kernel*. A classical kernelization example is Vertex Cover, which admits a kernel with $2k$ vertices, where k denotes the size of the cover [37]. Kernelization is a central topic in parameterized complexity, and many non-trivial upper and lower bounds on the kernelizability of various problems are known; see, e.g., the books of Fomin et al. [18] or Cygan et al. [13]. Instance reductions also carry practical significance in speeding up subsequent computations; e.g., the winning contribution in the 2016 PACE challenge for Feedback Vertex Set used a novel kernelization step as a key component (see `https://pacechallenge.wordpress.com/`).

[*] Parts of this article appeared as a preliminary version in Proceedings of the 23rd International Conference on Principles and Practice of Constraint Programming (CP 2017)

[†] This is the corresponding author

[‡] Department of Computer Science, Linköping university, Sweden; `victor.lagerkvist@liu.se`

[§] Department of Computer Science, Royal Holloway, University of London; `magnus.wahlstrom@rhul.ac.uk`

When the complexity parameter is the number of variables or vertices n of the instance, kernelization is also referred to as *sparsification*, although sparsifications are sometimes allowed to use superpolynomial time. A prominent example of the latter is the famous *sparsification lemma* that underpins research into the exponential time hypothesis [23]; according to this lemma, for every k there is a subexponential-time sparsification of k-SAT into $O(n)$ clauses, and hence a total size of $\tilde{O}(n)$ bits[1].

If the sparsification is restricted to polynomial time, results are more rare, but some do exist. On the one hand, for many problems, including k-SAT and Vertex Cover, it has been shown that non-trivial polynomial-time sparsification is impossible unless the polynomial hierarchy collapses [15]; concretely, under this assumption (which we will make implicitly in the sequel), there is no sparsification for k-SAT to $O(n^{k-\varepsilon})$ bits for any $\varepsilon > 0$, whereas an encoding in $O(n^k)$ bits is trivial. Similar negative results are known for other problems [26, 25]. On the other hand, there are examples of SAT problems for which non-trivial sparsification is possible. The first such result was by Bart Jansen (unpublished until recently [24]), who observed that 1-in-k-SAT admits a kernel with at most n constraints using Gaussian elimination. More surprisingly, Jansen and Pieterse [26] showed that the Not-All-Equal k-SAT problem admits a kernel with $O(n^{k-1})$ constraints, improving on the trivial bound by a factor of n and settling an implicit open problem. In later research, they improved and generalized the method, and showed that the bound of $O(n^{k-1})$ is tight [24]. These bounds follow by encoding the instance as the roots of a system of bounded-degree polynomial equations, and exploiting the bounded rank of the resulting system to eliminate superfluous constraints. This is a natural method, with strong explanatory power for non-trivial sparsifications both of SAT problems and more generally [24, 25]. But does it really cover all cases of non-trivial polynomial-time sparsification of SAT problems?

Let us make the question more precise. A *constraint language* Γ is a (finite or infinite) set of relations over a fixed domain D, and a constraint over Γ is (informally) a requirement that $R(X)$ holds, for some relation $R \in \Gamma$ and tuple of variables X. The *constraint satisfaction problem* $\text{CSP}(\Gamma)$ over Γ takes as input a set of constraints over Γ on a set of variables V, and asks whether there is an assignment $V \to D$ that satisfies every constraint. If $D = \{0, 1\}$ is the Boolean domain, then the problem is referred to as $\text{SAT}(\Gamma)$. For particular choices of Γ, this can define problems such as k-Coloring, k-SAT, 1-in-k-SAT and Not-All-Equal k-SAT. As a concrete example, the k-Coloring problem can be realized as $\text{CSP}(\{\neq_k\})$ where $\neq_k$ denotes the disequality relation on a domain with k elements. We then ask, for which languages Γ does $\text{CSP}(\Gamma)$ have a kernel of $O(n^c)$ constraints, for some constant $c \geq 1$, and in particular, for which languages is there a kernel with $O(n)$ constraints? (Note that for every finite Γ there is a trivial polynomial kernel in n, produced by simply discarding duplicate constraints, but as we saw above, for some languages this can be improved upon.)

We show how the above-described method of kernelization can be generalized using concepts from the study of constraint satisfaction problems. Consider a constraint $R(X)$ over the domain $D = \{0, 1\}$. By the above method, we would seek a low-degree polynomial $P(X)$ over some finite field F such that $P(X) = 0$ if and only if $R(X)$ holds. If this can be done for all relations $R \in \Gamma$, with polynomials of max-degree d, then $\text{SAT}(\Gamma)$ has a kernel of $O(n^d)$ constraints. We generalize this to finding an *extension* of R to some larger domain D', in a way such that the extension belongs to a language Γ' with certain algebraic properties (a Maltsev or k-edge polymorphism). Existing polynomial-time algorithms for $\text{CSP}(\Gamma')$ can then be adapted to find a polynomial kernel for the input instance. We show that this gives a direct and proper generalization of the method of bounded-degree polynomials, and we study in some detail the conditions required for such an extension to exist.

To describe our approach and results more fully, we review the algebraic approach to CSPs.

The Algebraic Approach in Parameterized and Fine-Grained Complexity

For any language Γ, the classical complexity of $\text{CSP}(\Gamma)$ (i.e., whether $\text{CSP}(\Gamma)$ is in P or is NP-complete) is determined by the existence of certain algebraic invariants of Γ known as *polymorphisms* [27]. These notions will be formally defined in Section 2, but at the moment we may think of the polymorphisms of a constraint language as higher-arity generalizations of homomorphisms, i.e., operations preserving the structure of the relations in the constraint language. The connection between constraint languages and their associated polymorphisms gave rise to the *algebraic approach* for characterizing the complexity of $\text{CSP}(\Gamma)$. For a long time it was conjectured not only that $\text{CSP}(\Gamma)$ is either in P or is NP-complete

[1] $\tilde{O}(n)$ supresses logarithmic factors.

for every Γ [17], but also that the tractability of a CSP problem can be characterized by a finite list of polymorphisms [8]. Recently, Zhuk and Bulatov have independently announced affirmative solutions to this conjecture [6, 41].

However, for purposes of parameterized and fine-grained complexity questions, looking at polymorphisms alone is too coarse. More technically, the polymorphisms of Γ characterize the expressive power of Γ up to *primitive positive definitions*, i.e., up to the use of conjunctions, equality constraints, and existential quantification, whereas for many questions a liberal use of existentially quantified local variables is not allowed. For example, k-ary clauses have primitive positive definitions using 3-clauses via the classical implementation

$$(x_1 \vee \ldots \vee x_k) \equiv \exists y_1, \ldots, y_{k-3} : (x_1 \vee x_2 \vee y_1) \wedge (\neg y_1 \vee x_3 \vee y_2) \wedge \ldots \wedge (\neg y_{k-3} \vee x_{k-1} \vee x_k).$$

Therefore, the 3-SAT and k-SAT problems are identical with respect to the polymorphisms of the language, despite having distinctly different properties with respect to kernel size and running time. In such cases, one may look at the expressive power under *quantifier-free* primitive positive definitions, allowing only conjunctions and equality constraints. This expressive power is characterized by more fine-grained algebraic invariants called *partial polymorphisms*. For example, there are numerous dichotomy results for the complexity of *parameterized* SAT(Γ) and CSP(Γ) problems, both for so-called FPT algorithms and for kernelization [30, 31, 32, 36], and in each of the cases listed, a dichotomy is given which is equivalent to requiring a finite list of partial polymorphisms of Γ. Similarly, Jonsson et al. [29] showed that the exact running times of NP-hard SAT(Γ) and CSP(Γ) problems, in terms of the number of variables n, are characterized by the partial polymorphisms of the constraint language Γ. Recently, this approach was also applied in a research programme of classifying NP-hard SAT(Γ) problems admitting exponentially improved upper bounds [34].

Unfortunately, studying properties of SAT(Γ) and CSP(Γ) for questions phrased in terms of the size parameter n is again more complicated than for more permissive parameters k. For example, it is known that for every finite set P of strictly partial polymorphisms, the number of relations invariant under P is double-exponential in terms of the arity n [33, Lemma 35]; hence, such relations can in general not be described by poly(n) bits. As we show later, it also holds that the existence of a polynomial kernel cannot be characterized by such a finite set P. Instead, such a characterization must be given in another way. For example, Lagerkvist et al. [35] provide a way to finitely characterize all partial polymorphisms of a finite Boolean language Γ, whereas in the present article, we pursue a model with an infinite set P represented via a uniformly described basis.

Our Results

We study the kernelizability of CSP(Γ) using methods inspired by the algebraic approach to CSP. Our kernelization method is valid for both finite and infinite languages but as the latter requires an additional technical assumption, we for the purposes of this introduction assume that Γ is finite. We show kernelizations based on extending an NP-hard language Γ on a domain D into a tractable language $\hat{\Gamma}$ on a domain $D' \supset D$, where $\hat{\Gamma}$ has a polymorphism that guarantees that instances of CSP($\hat{\Gamma}$) have small representations computable in polynomial time. We prove that this extension property is equivalent to the requirement that Γ is preserved by certain partial operations that are constructible by the polymorphisms of the larger language $\hat{\Gamma}$. This information can then be used to reduce an instance of CSP(Γ) on n variables to an equivalent instance with $O(n^c)$ constraints, where c is a constant that depends on the algebraic properties of $\hat{\Gamma}$. This method also solves the more difficult problem of finding a polynomial-sized *basis*, i.e., given an instance $I = (V, C)$ of CSP(Γ) on $|V| = n$ variables and with constraint set C, we compute a set $C' \subseteq C$ with $|C'| = O(n^c)$ such that every assignment satisfying C' also satisfies every constraint in C. Our results generalize and extend the kernelization results of Jansen and Pieterse [24]. (Recently, these results were further sharpened; see Chen et al. [11].)

When the host language $\hat{\Gamma}$ has a *Maltsev polymorphism*, we refer to this as a *Maltsev extension* of Γ, and we show that CSP(Γ) has a basis of $O(n)$ constraints using results and procedures from the so-called simple algorithm for Maltsev constraints by Bulatov and Dalmau [7]. Particular examples of this case include relations defined via linear equations over a finite field, such as the problem 1-IN-k-SAT, and relations defined via equations over a finite group (see Section 3.1). This case covers all known NP-hard CSP(Γ) problems with kernels of size $O(n)$.

We also consider two approaches that yield kernels of polynomial but not linear size. Most immediately, if $\hat{\Gamma}$ has a *k-edge polymorphism* for some $k \geq 2$, then $\mathrm{CSP}(\Gamma)$ has basis of $O(n^{k-1})$ constraints using the *few subpowers* algorithm of Idziak et al. [22], similarly to the application of the algorithm for the Maltsev case above. Indeed, a Maltsev operation is precisely a k-edge operation for $k = 2$. We refer to this as a *k-edge extension* of Γ. Extending this further, we consider a notion that generalizes the approach of defining constraints as roots of bounded-degree polynomials, as used by Jansen and Pieterse [24]. For this, we focus on the Boolean case ($D = \{0, 1\}$). Observe that a polynomial of degree d over a field $\mathbb{F}$, with variable set V, can be viewed as a linear equation over $\mathbb{F}$ with variables corresponding to tuples over V of size at most d. Therefore, a second natural way of showing a basis of $O(n^c)$ constraints, $c > 1$, for a language Γ is to show that every relation $R \in \Gamma$ can be extended to a relation R' over ($\leq d$)-ary tuples of variables, such that R' admits a Maltsev extension. We refer to R' as a *degree-d extension* of R. Computing a linear basis for the extended instance then gives us a basis for $\mathrm{SAT}(\Gamma)$ of $O(n^d)$ constraints. As outlined above, roots of bounded-degree polynomials are a special case of this approach. Combining the two approaches is also possible: if Γ has a degree-d extension which in turn has a k-edge extension, then $\mathrm{SAT}(\Gamma)$ has a basis of $O(n^{(k-1)d})$ constraints.

Interestingly, the notion of a k-edge extension is on its own insufficient to capture all Boolean languages Γ with a polynomial basis. That is, for every $k \geq 2$ there is a language Γ with a degree-2 extension with a Maltsev extension, but which does not have a k-edge extension. This is in contrast to the algebraic side, where there is a notion of an algebra having *few subpowers*, closely related to the idea of only being able to primitively positively define $2^{\mathrm{poly}(n)}$ distinct n-ary relations, where it is known that an algebra has few subpowers if and only if it has a k-edge term [2].

Finally, we turn to the question of lower bounds, and of obtaining a better understanding of the algebraic properties of languages with useful extensions. It turns out that this question has a strong correspondence to properties of the partial polymorphisms of the constraint language. As discussed earlier, the property of admitting an extension $\hat{\Gamma}$ with respect to a concrete operation p is tantamount to Γ being preserved by certain partial operations constructible by p. Moreover, if Γ is a constraint language not preserved by operations of this form, this can sometimes be exploited in order to prove lower bounds on kernelizability of $\mathrm{CSP}(\Gamma)$. For the case of Maltsev extensions, we show how to relax this to a set of *universal partial polymorphisms* P which characterize the existence of a Maltsev extension of Γ without needing to refer to a concrete operation p. We prove that the set P is necessarily infinite, but has a natural basis of gradually stronger partial operations ϕ_i, $i \geq 1$. We then show, towards establishing a dichotomy for $\mathrm{SAT}(\Gamma)$ problems with linear kernels, that if Γ is a Boolean language not preserved by ϕ_1 (and thus, $\mathrm{SAT}(\Gamma)$ does not admit a Maltsev extension), and if $\mathrm{SAT}(\Gamma)$ is NP-hard, then $\mathrm{SAT}(\Gamma)$ does not admit a kernel of $O(n^{2-\varepsilon})$ bits for any $\varepsilon > 0$ unless the polynomial hierarchy collapses.

In addition, as discussed earlier, we show that for any finite set P of strictly partial operations, and for every $c > 1$, there is a Boolean language Γ preserved by P which does not admit a kernel of $O(n^{c-\varepsilon})$ bits for any $\varepsilon > 0$ unless the polynomial hierarchy collapses. Hence, finite lists of partial polymorphisms cannot give non-trivial size guarantees in kernelization of NP-hard SAT problems, which is in stark contrast to the setting of classical complexity of CSP, where tractability can always be explained by finite lists of polymorphisms.

Infinite languages. Our results apply in principle to infinite constraint languages, but there are some technical caveats. Let Γ be an infinite constraint language, and let $\hat{\Gamma}$ be a suitable extension (e.g., a Maltsev extension of Γ over a finite domain). Then briefly, all statements above about a *basis* for $\mathrm{CSP}(\Gamma)$ hold unchanged, but the computation of said basis can be more challenging. This has two sources. First, the construction of Γ' from Γ may itself be computationally challenging, even if Γ is fixed. Second, the direct application of the above-mentioned algorithm for $\mathrm{CSP}(\Gamma')$ would require a running time proportional to the number of tuples in the relations occurring in the instance, which may well be exponential in n. For these reasons, we cannot in the general case conclude the existence of a polynomial-time sparsification algorithm, even though a polynomial- or linear-sized basis can be shown.

Structure of the Article

Section 2 gives the required preliminaries. In Section 3 we present basic results on language extensions and show how having a Maltsev extension implies that $\mathrm{CSP}(\Gamma)$ has a linear basis. In Section 4 we present the approaches of k-edge extensions and bounded-degree extensions for computing a basis of $O(n^c)$ constraints, $c > 1$. In Section 5 we present the further algebraic characterisations of Maltsev extensions and pursue the problem of proving lower bounds on kernelizability. In Section 6 we summarize our results and discuss gaps in our knowledge together with future questions.

2 Preliminaries

In this section we introduce the constraint satisfaction problem, kernelization, and the algebraic machinery that will be used throughout the article.

2.1 Operations and Relations

For $n \geq 1$ we let $[n]$ denote the set $\{1, \ldots, n\}$. An n-ary function $f : D^n \rightarrow D$ over a domain D is typically referred to as a *operation* on D, although we will sometimes use the terms function and operation interchangeably. We let $\text{ar}(f) = n$ denote the arity of f. Similarly, if $R \subseteq D^n$ is an n-ary relation over D we let $\text{ar}(R) = n$. If $t \in D^n$ is a tuple we let $t[i]$ denote the ith element in t and for $n' \leq n$ we let

$$\text{pr}_{i_1,\ldots,i_{n'}}(t) = (t[i_1], \ldots, t[i_{n'}])$$

denote the *projection* of t on (not necessarily distinct) coordinates $i_1, \ldots, i_{n'} \in [n]$. Similarly, if R is an n-ary relation we let

$$\text{pr}_{i_1,\ldots,i_{n'}}(R) = \{\text{pr}_{i_1,\ldots,i_{n'}}(t) \mid t \in R\}.$$

We will often represent relations by logical formulas, and if ψ is a first-order formula with free variables $x_1, \ldots, x_k$ we write $R(x_1, \ldots, x_k) \equiv \psi(x_1, \ldots, x_k)$ to denote the relation $R = \{(f(x_1), \ldots, f(x_k)) \mid f \text{ is a satisfying assignment to } \psi\}$.

2.2 The Constraint Satisfaction Problem

A set of relations Γ is referred to as a *constraint language*. The *constraint satisfaction problem* over a constraint language Γ over D (CSP(Γ)) is the computational decision problem defined as follows.

Instance: A set V of variables and a set C of constraint applications $R(x_1, \ldots, x_k)$ where $R \in \Gamma$, $\text{ar}(R) = k$, and $x_1, \ldots, x_k \in V$.
Question: Is there a function $f : V \rightarrow D$ such that $(f(x_1), \ldots, f(x_k)) \in R$ for each $R(x_1, \ldots, x_k)$ in C?

In the particular case when Γ is Boolean we denote CSP(Γ) by SAT(Γ), and we let BR denote the set of all Boolean relations. Throughout the article we will assume that the constraints in an instance of CSP(Γ) are represented explicitly as a list of tuples. This is not the only possible choice of representation but is in line with the majority of theoretical CSP research.

Example 1. *Consider the ternary relation* $R_{1/3} = \{(0, 0, 1), (0, 1, 0), (1, 0, 0)\}$. *It is then readily seen that* SAT($\{R_{1/3}\}$) *can be viewed as an alternative formulation of the* 1-in-3-SAT *problem restricted to instances consisting only of positive literals. More generally, if we let*

$$R_{1/k} = \{(x_1, \ldots, x_k) \in \{0, 1\}^k \mid x_1 + \ldots + x_k = 1\},$$

then SAT($\{R_{1/k}\}$) *is a natural formulation of* 1-in-k-SAT *without negation.*

2.3 Kernelization

A *parameterized problem* is a subset of $\Sigma^* \times \mathbb{N}$ where Σ is a finite alphabet. Hence, each instance is associated with a natural number, called the *parameter*.

Definition 1. A *kernelization algorithm*, or a *kernel*, for a parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$ is a polynomial-time algorithm which, given an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, computes $(x', k') \in \Sigma^* \times \mathbb{N}$ such that (1) $(x, k) \in L$ if and only if $(x', k') \in L$ and (2) $|x'| + k' \leq f(k)$ for some function f.

The function f in the above definition is sometimes called the *size* of the kernel. In this article, we are mainly interested in the case where the parameter denotes the number of variables n in a given instance of CSP(Γ), and thus aim to construct kernels with a small number of constraints.

2.4 Polymorphisms and Partial Polymorphisms

In this section we define the link between constraint languages and algebras that was promised in Section 1. If f is an n-ary operation and $t_1, \ldots, t_n$ a sequence of k-ary tuples we can in a natural way obtain a k-ary tuple by applying f componentwise, i.e.,

$$f(t_1, \ldots, t_n) = (f(t_1[1], \ldots, t_n[1]), \ldots, f(t_1[k], \ldots, t_n[k])).$$

Definition 2. An n-ary operation f is a *polymorphism* of a k-ary relation R if $f(t_1, \ldots, t_n) \in R$ for each sequence of tuples $t_1, \ldots, t_n \in R$.

If f is a polymorphism of R we also say that R is *invariant* under f, or that f *preserves* R, and for a constraint language Γ we let $\mathrm{Pol}(\Gamma)$ denote the set of operations preserving every relation in Γ. Similarly, if F is a set of functions, we let $\mathrm{Inv}(F)$ denote the set of all relations invariant under each operation in F. Sets of functions of the form $\mathrm{Pol}(\Gamma)$ are referred to as *clones*. It is well known that $\mathrm{Pol}(\Gamma)$

1. for each $n \geq 1$ and each $1 \leq i \leq n$ contains the *projection* $\pi_i^n(x_1, \ldots, x_i, \ldots, x_n) = x_i$, and
2. is closed under *composition*, i.e., if $f, g_1, \ldots, g_m \in \mathrm{Pol}(\Gamma)$, where $\mathrm{ar}(f) = m$ and $\mathrm{ar}(g_i) = n$ for each $1 \leq i \leq m$, then the operation

   $$f \circ g_1, \ldots, g_m(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n))$$

   is included in $\mathrm{Pol}(\Gamma)$.

Similarly, sets of the form $\mathrm{Inv}(F)$ are referred to as *relational clones*, or *co-clones*, and are sets of relations closed under *primitive positive definitions* (pp-definitions), which are logical formulas consisting of existential quantification, conjunction, and equality constraints. In symbols, we say that a k-ary relation R has a pp-definition over a constraint language Γ over a domain D if

$$R(x_1, \ldots, x_k) \equiv \exists y_1, \ldots, y_{k'} : R_1(\mathbf{x_1}) \wedge \ldots \wedge R_m(\mathbf{x_m}),$$

where each $R_i \in \Gamma \cup \{\mathrm{Eq}\}$, $\mathrm{Eq} = \{(\mathrm{x}, \mathrm{x}) \mid \mathrm{x} \in \mathrm{D}\}$ and each $\mathbf{x_i}$ is an $\mathrm{ar}(R_i)$-ary tuple of variables over $x_1, \ldots, x_k$, $y_1, \ldots, y_{k'}$. Clones and co-clones are related via the following *Galois connection.*

Theorem 1 ([3, 4, 19])**.** *Let Γ and Γ' be two constraint languages. Then $\Gamma \subseteq \mathrm{Inv}(\mathrm{Pol}(\Gamma'))$ if and only if $\mathrm{Pol}(\Gamma') \subseteq \mathrm{Pol}(\Gamma)$.*

Example 2. *It is an easy exercise to verify that the only polymorphisms of the relation $R_{1/3}$ from Example 1 are the projections. In order words every Boolean relation is invariant under $\mathrm{Pol}(\{R_{1/3}\})$, and Theorem 1 then implies that $R_{1/3}$ can pp-define every Boolean relation.*

As a shorthand, we let $[F] = \mathrm{Pol}(\mathrm{Inv}(F))$ denote the smallest clone containing F[2] and $\langle \Gamma \rangle = \mathrm{Inv}(\mathrm{Pol}(\Gamma))$ be the smallest co-clone containing Γ. Using Theorem 1 Jeavons et al. proved that if Γ and Γ' are two finite constraint languages and $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma')$, then $\mathrm{CSP}(\Gamma')$ is polynomial-time many-one reducible to $\mathrm{CSP}(\Gamma)$ [27]. As remarked in Section 1, while this theorem is useful for establishing complexity dichotomies for CSP and related problems [1, 12], it offers little information on whether a problem admits a kernel of a particular size. Hence, in order to have any hope of studying kernelizability of SAT and CSP problems, we need algebras more fine-grained than polymorphisms. In our case these algebras will consist of *partial operations* instead of total operations.

Definition 3. An n-ary *partial operation* over a set D of values is a map of the form $f : X \to D$, where $X \subseteq D^n$ is called the *domain* of f, and denoted by $\mathrm{domain}(f)$.

As in the case of total operations we let $\mathrm{ar}(f) = n$ denote the arity of f. If f and g are n-ary partial operations such that $\mathrm{domain}(g) \subseteq \mathrm{domain}(f)$ and $f(x_1, \ldots, x_n) = g(x_1, \ldots, x_n)$ for each $(x_1, \ldots, x_n) \in \mathrm{domain}(g)$, then g is said to be a *subfunction* of f. A partial operation which is a subfunction of a total projection is called a *partial projection.* We are now ready to define the partial analogue of a polymorphism.

[2] Note that we also use $[n]$ for denoting the set $\{1, \ldots, n\}$, but the intended meaning will always be clear from the context.

Definition 4. An n-ary partial operation f is a *partial polymorphism* of a k-ary relation R if, for every sequence $t_1, \ldots, t_n \in R$, either $f(t_1, \ldots, t_n) \in R$ or there exists $i \in [k]$ such that $(t_1[i], \ldots, t_n[i]) \notin \text{domain}(f)$.

In essence this simply means that f either results in a tuple included in R, or is undefined for some application. Again, this notion easily generalizes to constraint languages, and if we let $\text{pPol}(\Gamma)$ denote the set of partial polymorphisms of the constraint language Γ, we obtain a *strong partial clone.* It is known that strong partial clones are sets of partial operations which (1) are closed under composition of partial operations and (2) contains all partial projections [38]. More formally, the first condition means that if $f, g_1, \ldots, g_m$ are included in the strong partial clone, where f is m-ary and every g_i is n-ary, then the partial operation

$$f \circ g_1, \ldots, g_m(x_1, \ldots, x_n) = f(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n))$$

is also included in the strong partial clone, and $(x_1, \ldots, x_n) \in \text{domain}(f \circ g_1, \ldots, g_m)$ if and only if

$$(x_1, \ldots, x_n) \in \bigcap_{i=1}^{m} \text{domain}(g_i)$$

and

$$(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n)) \in \text{domain}(f).$$

The second condition, containing all partial projections, is known to be equivalent to closure under taking subfunctions; a property which in the literature is sometimes called *strong.*

If F is a set of partial operations we let $\text{Inv}(F)$ denote the set of all relations invariant under F, but this time $\text{Inv}(F)$ is in general not closed under pp-definitions, but under *quantifier-free primitive positive definitions* (qfpp-definitions). As the terminology suggests, a relation R has a qfpp-definition over Γ if R is definable via a pp-formula which does not make use of existential quantification. Such formulas are sometimes simply called *conjunctive formulas.* We have the following Galois connection between $\text{Inv}(\cdot)$ and $\text{pPol}(\cdot)$.

Theorem 2 ([19, 38])**.** *Let* Γ *and* Γ' *be two constraint languages. Then* $\Gamma \subseteq \text{Inv}(\text{pPol}(\Gamma'))$ *if and only if* $\text{pPol}(\Gamma') \subseteq \text{pPol}(\Gamma)$.

As a shorthand we let $[F]_s = \text{pPol}(\text{Inv}(F))$ denote the smallest strong partial clone containing the set of partial operations F, and $\langle \Gamma \rangle_{\nexists} = \text{Inv}(\text{pPol}(\Gamma))$ for a constraint language Γ be the smallest set of relations containing Γ which is closed under qfpp-definitions. The intended mnemonic for the latter notation is that the pp-formulas defining the relations in $\langle \Gamma \rangle_{\nexists}$ are quantifier-free and thus cannot make use of existential quantification. Using the Galois connection in Theorem 2 Jonsson et al. [29] proved the following theorem.

Theorem 3. *Let* Γ *and* Δ *be two finite constraint languages. If* $\text{pPol}(\Gamma) \subseteq \text{pPol}(\Delta)$ *then there exists a polynomial-time many-one reduction from* $\text{SAT}(\Delta)$ *to* $\text{SAT}(\Gamma)$ *which maps an instance* (V, C) *of* $\text{SAT}(\Delta)$ *to an instance* (V', C') *of* $\text{SAT}(\Gamma)$ *where* $|V'| \leq |V|$ *and* $|C'| \leq c|C|$, *where* c *depends only on* Γ *and* Δ.

In particular this implies that if $\text{pPol}(\Gamma) \subseteq \text{pPol}(\Delta)$ and $\text{CSP}(\Gamma)$ is solvable in $O(c^n)$ time for a constant $c > 1$, then $\text{CSP}(\Delta)$ is solvable in $O(c^n)$ time, too. Note that Theorem 3 also shows a connection between partial polymorphims and kernelizability, in the sense that if $\text{pPol}(\Gamma) \subseteq \text{pPol}(\Delta)$ and $\text{CSP}(\Gamma)$ has a kernel with $O(f(n))$ constraints for some function f, then $\text{CSP}(\Delta)$ also admits a kernel with $O(f(n))$ constraints.

2.5 Maltsev Operations, Signatures and Compact Representations

A *Maltsev operation* over $D \supseteq \{0, 1\}$ is a ternary operation ϕ which for all $x, y \in D$ satisfies the two identities $\phi(x, x, y) = y$ and $\phi(x, y, y) = x$. Before we can explain the powerful, structural properties of relations invariant under Maltsev operations, we need a few technical definitions from Bulatov and Dalmau [7]. Let t, t' be two n-ary tuples over D. We say that (t, t') *witnesses* a tuple $(i, a, b) \in [n] \times D^2$

if $\mathrm{pr}_{1,\ldots,i-1}(t) = \mathrm{pr}_{1,\ldots,i-1}(t')$, $t[i] = a$, and $t'[i] = b$. The *signature* of an n-ary relation R over D is then defined as

$$\mathrm{Sig}(R) = \{(i, a, b) \in [n] \times D^2 \mid \exists t, t' \in R \text{ such that } (t, t') \text{ witnesses } (i, a, b)\},$$

and we say that $R' \subseteq R$ is a *representation* of R if $\mathrm{Sig}(R) = \mathrm{Sig}(R')$. If R' is a representation of R it is said to be *compact* if $|R'| \leq 2|\mathrm{Sig}(R)|$, and it is known that every relation invariant under a Maltsev operation admits a compact representation. Furthermore, we have the following theorem from Bulatov and Dalmau, where we let $\langle R\rangle_f$ denote the smallest superset of R preserved under the operation f.

Theorem 4 ([7]). *Let ϕ be a Maltsev operation over a finite domain, $R \in \mathrm{Inv}(\{\phi\})$ a relation, and R' a representation of R. Then $\langle R'\rangle_\phi = R$.*

We remark that $\langle R\rangle_f$ can also be defined as the relation

$$\langle R\rangle_f = \{t \mid g(t_1, \ldots, t_{\mathrm{ar}(g)}) = t, t_1, \ldots, t_{\mathrm{ar}(g)} \in R, g \in [\{f\}]\}.$$

Hence, relations invariant under Maltsev operations are reconstructible from their compact representations, which is one of the underlying ideas behind the polynomial-time algorithm for Maltsev constraints by Bulatov and Dalmau [7]. This property will prove to be crucial in the forthcoming section where we describe techniques for extending a constraint language into a constraint language preserved by a Maltsev operation.

3 Maltsev Extensions and Kernels of Linear Size

In this section we give general upper bounds for kernelization of NP-hard CSP problems based on algebraic conditions. We begin in Section 3.1 by outlining the polynomial-time algorithm for Maltsev constraints, and modify this algorithm in Section 3.2 to construct linear-sized kernels for $\mathrm{CSP}(\Gamma)$ problems satisfying an algebraic condition related to the existence of certain Maltsev operations.

3.1 The Simple Algorithm for Maltsev Constraints

At this stage the connection between Maltsev operations, compact representations and tractability of Maltsev constraints might not be immediate to the reader. We therefore give a brief description of the simple algorithm for Maltsev constraints from Bulatov and Dalmau [7], which will henceforth simply be referred to as the *Maltsev algorithm*. In a nutshell, the Maltsev algorithm operates as follows, where ϕ is a Maltsev operation over a finite set D (note that $\mathrm{Inv}(\{\phi\})$ is the infinite constraint language consisting of all relations preserved by ϕ).

1. Let $(V, \{C_1, \ldots, C_m\})$ be an instance of $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$, and S_0 a compact representation of $D^{|V|}$.
2. For each $i \in [m]$ compute a compact representation S_i of the solution space of the instance $(V, \{C_1, \ldots, C_i\})$ using S_{i-1}.
3. Answer yes if $S_m \neq \emptyset$ and no otherwise.

The second step is accomplished by removing the tuples from $\langle S_{i-1}\rangle_\phi$ that are not compatible with the constraint C_i. While the basic idea behind the Maltsev algorithm is not complicated, the intricate details of the involved subprocedures are outside the scope of this article, and we refer the reader to Bulatov and Dalmau [7], and Dyer and Richerby [16] for a slightly simplified presentation. We note that although the Maltsev algorithm applies to infinite languages, it is assumed that the relations in the input are specified by explicit lists of tuples, i.e., the running time includes a factor proportional to $\max |R|$ over relations R used in the input.

Example 3. *Let $G = (D, \cdot)$ be a group over a finite set D, i.e., $\cdot$ is a binary, associative operator, D is closed under $\cdot$ and contains an identity element 1_G, and each element $x \in D$ has an inverse element $x^{-1} \in D$ such that $x \cdot x^{-1} = 1_G$. The ternary operation $s(x, y, z) = x \cdot y^{-1} \cdot z$ is referred to as the* coset generating operation *of G, and is Maltsev since $s(x, y, y) = x \cdot y^{-1} \cdot y = x$ and $s(x, x, y) = x \cdot x^{-1} \cdot y = y$. The problem $\mathrm{CSP}(\mathrm{Inv}(\{s\}))$ is known to be tractable via the algorithm from Feder and Vardi [17], but since s is a Maltsev operation $\mathrm{CSP}(\mathrm{Inv}(\{s\}))$ can also be solved via the Maltsev algorithm.*

Another early class of tractable CSP problems was discovered via the observation that if R is preserved by a certain Maltsev operation, it can be viewed as the solution space of a system of linear equations.

Example 4. *An* Abelian *group $G = (D, +)$ is a group where $+$ is commutative. Similar to Example 3 we can consider the coset generating operation $s(x, y, z) = x - y + z$, where $-y$ denotes the inverse of the element y. If $|D|$ is prime it is known that $R \in \mathrm{Inv}(\{s\})$ if and only if R is the solution space of a system of linear equations modulo $|D|$ [28]. Hence, the problem $\mathrm{CSP}(\mathrm{Inv}(\{s\}))$ can efficiently be solved with Gaussian elimination, but can also be solved via the Maltsev algorithm.*

3.2 Upper Bounds Based on Maltsev Extensions

In this section we use a variation of the Maltsev algorithm to obtain kernels of CSP problems. First, observe that Γ is never preserved by a Maltsev operation when $\mathrm{CSP}(\Gamma)$ is NP-hard, since the Maltsev algorithm then solves $\mathrm{CSP}(\Gamma)$ in polynomial time. The gist of our approach is instead to find a closely related constraint language $\hat{\Gamma}$ which is preserved by a Maltsev operation. This will allow us to use the advantageous properties of relations invariant under Maltsev operations in order to compute a kernel for the original problem $\mathrm{CSP}(\Gamma)$. We thus begin by making the following definition.

Definition 5. A constraint language Γ over a domain D admits an *extension* over the constraint language $\hat{\Gamma}$ over $E \supseteq D$ if there exists a bijection $h : \Gamma \to \hat{\Gamma}$ such that $\mathrm{ar}(h(R)) = \mathrm{ar}(R)$ and $h(R) \cap D^{\mathrm{ar}(R)} = R$ for every $R \in \Gamma$.

In model-theoretic terminology Γ is sometimes referred to as an *induced substructure* of $\hat{\Gamma}$, and $\hat{\Gamma}$ is called either an extension of Γ or a *superstructure* of Γ [21]. Trivially, every constraint language is an extension of itself, but we are particularly interested in the case when $\hat{\Gamma}$ is preserved by an operation not present in $\mathrm{Pol}(\Gamma)$, and say that Γ admits a *Maltsev extension* if $\hat{\Gamma}$ is preserved by a Maltsev operation.

In general, we do not exclude the possibility that the domain E is infinite. In this section, however, we will only be concerned with finite domains, and therefore do not explicitly state this assumption. If the bijection h is efficiently computable and there exists a polynomial p such that $h(R)$ can be computed in $O(p(|R|))$ time for each $R \in \Gamma$, then we say that Γ admits a *polynomially bounded* extension. In particular, an extension over a finite domain of any finite Γ is polynomially bounded.

Example 5. *Recall from Section 2.2 that $R_{1/3}$ consists of the three tuples $(0, 0, 1), (0, 1, 0)$, and $(1, 0, 0)$. We claim that $R_{1/3}$ has a Maltsev extension over $\{0, 1, 2\}$. Let $\hat{R}_{1/3} = \{(x, y, z) \in \{0, 1, 2\}^3 \mid x + y + z = 1 \pmod 3\}$. By definition, $\hat{R}_{1/3} \cap \{0, 1\}^3 = R_{1/3}$, so all that remains to prove is that $\hat{R}_{1/3}$ is preserved by a Maltsev operation. But recall from Example 4 that a relation R is the solution space of a system of linear equations over D, where $|D|$ is prime, if and only if R is preserved by the operation $x - y + z$ over D. Hence, $\hat{R}_{1/3}$ is indeed a Maltsev extension of $R_{1/3}$. More generally, one can also prove that $R_{1/k}$ has a Maltsev extension to a finite domain D where $|D| \geq k$ and $|D|$ is prime.*

Example 5 also shows that the existence of a Maltsev extension of Γ in general cannot be witnessed by the polymorphisms of Γ, since $\mathrm{Pol}(\{R_{1/3}\})$ consists only of projections. However, we will now prove that the property of admitting an extension with respect to an operation f can be witnessed by the partial polymorphisms of the constraint language Γ. The basic idea is that one can construct partial polymorphisms of Γ by restricting $g \in [\{f\}]$ to the domain of Γ.

Definition 6. Let $f : E^k \to E$ be a k-ary operation over a domain E and let $D \subseteq E$. Define the k-ary partial operation $f_{|D}$ over D as

$$\mathrm{domain}(f_{|D}) = \{(x_1, \ldots, x_k) \in D^k \mid f(x_1, \ldots, x_k) \in D\},$$

and

$$f_{|D}(x_1, \ldots, x_k) = f(x_1, \ldots, x_k)$$

for every $(x_1, \ldots, x_k) \in \mathrm{domain}(f_{|D})$.

In other words $f_{|D}$ is the partial operation over D resulting by restricting f to tuples over D which also result in a value in D. As a shorthand we let $f_{|\mathbb{B}} = f_{|\{0,1\}}$ be the Boolean restriction.

Theorem 5. *Let Γ be a constraint language on a domain D and let f be an operation on a domain $E \supseteq D$. Define $\hat{\Gamma} = \{\langle R \rangle_f \mid R \in \Gamma\}$. Then the following statements are equivalent.*

1. *$\hat{\Gamma}$ is an extension of Γ preserved by f.*
2. *$g_{|D} \in \mathrm{pPol}(\Gamma)$ for every $g \in \mathrm{Pol}(\hat{\Gamma})$.*

Proof. For the first direction, assume that $\hat{\Gamma}$ is an extension of Γ, and assume that there exists $R \in \Gamma$ and an n-ary $g \in \mathrm{Pol}(\hat{\Gamma})$ such that $g_{|D}(t_1, \ldots, t_n) \notin R$ for $t_1, \ldots, t_n \in R$. By construction, $g_{|D}(t_1, \ldots, t_n) = t$ is a tuple in D^n. But since $\hat{R} \cap D^{\mathrm{ar}(R)} = R$, this implies that $t \notin \hat{R}$, hence $g(t_1, \ldots, t_n) = g_{|D}(t_1, \ldots, t_n) = t \notin \hat{R}$. Hence, g preserves neither $\hat{R}$ nor $\hat{\Gamma}$, which is a contradiction. We conclude that $g_{|D} \in \mathrm{pPol}(\Gamma)$.

For the other direction, assume that $\{g_{|D} \mid g \in \mathrm{Pol}(\hat{\Gamma})\} \subseteq \mathrm{pPol}(\Gamma)$ but that there exists $\hat{R} \in \hat{\Gamma}$ such that $\hat{R} \cap D^{\mathrm{ar}(R)} \supset R$. Let $t \in \hat{R} \cap D^{\mathrm{ar}(R)} \setminus R$. By construction of $\hat{R}$ it follows that there exists an n-ary $g \in [\{f\}]$ and $t_1, \ldots, t_n \in R$ such that $g(t_1, \ldots, t_n) = t \notin R$. But then it follows that $g_{|D}(t_1, \ldots, t_n)$ is defined as well, implying that $g_{|D}(t_1, \ldots, t_n) \notin R$. This contradicts the assumption that $g_{|D} \in \mathrm{pPol}(\Gamma)$ for every $g \in \mathrm{Pol}(\hat{\Gamma})$ (since $[\{f\}] \subseteq \mathrm{Pol}(\hat{\Gamma})$). □

Example 6. *Let us consider the operation $f(x, y, z) = x - y + z$ over the three element Abelian group from Example 5, which preserved the extension of $R_{1/3}$. One can verify that $f_{|\mathbb{B}}$ defines a ternary Boolean partial operation satisfying the two identities defining a Maltsev operation, and which is undefined otherwise. In fact, whenever Γ is a Boolean constraint language such that $\mathrm{SAT}(\Gamma)$ is NP-hard and Γ admits a Maltsev extension with respect to a Maltsev operation ϕ, $\phi_{|\mathbb{B}}$ will always result in $f_{|\mathbb{B}}$, which by Theorem 5 is guaranteed to preserve Γ. We will return to properties of this partial operation in Section 5.*

Hence, the existence of an extension $\hat{\Gamma}$ of a language Γ with respect to a concrete operation f can always be witnessed by the partial polymorphisms of Γ that are constructible via the operation f. It is also worth remarking that, while not complicated to prove, Theorem 5 also provides a novel, algebraic characterization of the substructure relationship between two relational structures. Note, however, that this theorem only applies when a concrete operation f is being considered, and thus cannot a priori be used to disprove the existence of a particular *type* of extension, e.g., a Maltsev extension. We revisit this issue in Section 5, where we describe a set of *universal* partial polymorphisms of Γ that guarantee the existence of a Maltsev extension.

We now proceed to describe the kernelization application of the Maltsev extension property. Given an instance $I = (\{x_1, \ldots, x_n\}, C)$ of $\mathrm{CSP}(\Gamma)$ we let

$$\Psi_I = \{(g(x_1), \ldots, g(x_n)) \mid g \,\text{satisfies}\, I\}$$

be the relation consisting of all satisfying assignments of I. If ϕ is a Maltsev operation and $I = (V, \{C_1, \ldots, C_m\})$ an instance of $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$ we let $\mathrm{Seq}(I) = (S_0, S_1 \ldots, S_m)$ denote the compact representations of the relations

$$\Psi_{(V,\emptyset)}, \Psi_{(V,\{C_1\})}, \ldots, \Psi_{(V,\{C_1,\ldots,C_m\})}$$

computed by the Maltsev algorithm. We remark that the ordering chosen in the sequence $\mathrm{Seq}(I)$ does not influence the upper bound in the forthcoming kernelization algorithm.

Definition 7. Let ϕ be a Maltsev operation, p a polynomial and let $\Delta \subseteq \mathrm{Inv}(\{\phi\})$. We say that Δ and $\mathrm{CSP}(\Delta)$ have *chain length* p if $|\{\langle S_i \rangle_\phi \mid i \in \{0, 1, \ldots, |C|\}\}| \leq p(|V|)$ for each instance $I = (V, C)$ of $\mathrm{CSP}(\Delta)$, where $\mathrm{Seq}(I) = (S_0, S_1, \ldots, S_{|C|})$.

We now have everything in place to define our kernelization algorithm.

Theorem 6. *Let Γ be a constraint language which admits a polynomially bounded Maltsev extension $\hat{\Gamma}$ with chain length p. Then $\mathrm{CSP}(\Gamma)$ has a kernel with $O(p(|V|))$ constraints.*

Proof. Let $\phi \in \mathrm{Pol}(\hat{\Gamma})$ denote the Maltsev operation witnessing the extension $\hat{\Gamma}$. Given an instance $I = (V, C)$ of $\mathrm{CSP}(\Gamma)$ we can obtain an instance $I' = (V, C')$ of $\mathrm{CSP}(\hat{\Gamma})$ by replacing each constraint $R_i(\mathbf{x_i})$ in C by $\hat{R}_i(\mathbf{x_i})$. We arbitrarily order the constraints as $C' = (C_1, \ldots, C_m)$ where $m = |C'|$. We then iteratively compute the corresponding sequence $\mathrm{Seq}(I') = (S_0, S_1, \ldots, S_{|C'|})$. This can be done in polynomial time with respect to the size of I via the same procedure as the Maltsev algorithm, since we assume that the constraints are explicitly represented. For each $i \in [m]$ we then do the following.

1. Let the ith constraint be $C_i = \hat{R}_i(x_{i_1}, \ldots, x_{i_r})$ with $\mathrm{ar}(\hat{R}_i) = r$.
2. For each $t \in S_{i-1}$ determine whether $\mathrm{pr}_{i_1,\ldots,i_r}(t) \in \hat{R}_i$.
3. If yes, then remove the constraint C_i, otherwise keep it.

This can be done in polynomial time with respect to the size of the instance I', since $|S_{i-1}|$ is bounded by a polynomial in $|V|$ and since the test $\mathrm{pr}_{i_1,\ldots,i_r}(t) \in \hat{R}_i$ can naively be checked in linear time with respect to $|\hat{R}_i|$. We claim that the procedure outlined above will correctly detect whether the constraint C_i is redundant or not with respect to $\langle S_{i-1} \rangle_\phi$, i.e., whether $\langle S_{i-1} \rangle_\phi = \langle S_i \rangle_\phi$. First, observe that if there exists $t \in S_{i-1}$ such that $\mathrm{pr}_{i_1,\ldots,i_r}(t) \notin \hat{R}_i$, then the constraint is clearly not redundant. Hence, assume that $\mathrm{pr}_{i_1,\ldots,i_r}(t) \in \hat{R}_i$ for every $t \in S_{i-1}$. Then $S_{i-1} \subseteq \langle S_i \rangle_\phi$, hence also $\langle S_{i-1} \rangle_\phi \subseteq \langle S_i \rangle_\phi$. On the other hand, $\langle S_i \rangle_\phi \subseteq \langle S_{i-1} \rangle_\phi$ holds trivially. Therefore, equality must hold.

Let $I'' = (V, C'')$ denote the resulting instance. Since $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$ has chain length p it follows that the sequence $\langle S_0 \rangle_\phi, \langle S_1 \rangle_\phi, \ldots, \langle S_{|C'|} \rangle_\phi$ contains at most $p(|V|)$ distinct elements, hence $|C''| \leq p(|V|)$. It also holds that $\Psi_{I'} = \Psi_{I''}$. Let D be the domain of Γ. Clearly, it holds that $\Psi_I = (\Psi_{I'} \cap D^{|V|}) = (\Psi_{I''} \cap D^{|V|})$. Hence, we can safely transform I'' to an instance I^* of $\mathrm{CSP}(\Gamma)$ by replacing each constraint $\hat{R}_i(\mathbf{x_i})$ with $R_i(\mathbf{x_i})$. Then I^* is an instance of $\mathrm{CSP}(\Gamma)$ with at most $p(|V|)$ constraints, such that $\Psi_I = \Psi_{I^*}$. In particular, I^* has a solution if and only if I has a solution. □

As with the Maltsev algorithm, the procedure runs in polynomial time with respect to the total size of the instance. For languages with bounded arity this simply means time polynomial in n, but it is worth noting that if Γ is infinite but concisely represented, then the applicability of the above algorithm depends on whether the underlying operations of the Maltsev algorithm can be performed in polynomial time with respect to this representation.

All that remains now is to bound the chain lengths of Maltsev extensions. It might be tempting to argue that the compact representation S_i decreases in size if the corresponding constraint is not redundant, but this strategy is not guaranteed to work since the compact representation is reconstructed in every iteration of the Maltsev algorithm. Instead, we will prove that either the compact representation or the associated signature shrinks if the constraint is not redundant. To accomplish this we need two subsidiary lemmas.

Lemma 1. *Let ϕ be a Maltsev operation over D and I an instance of* $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$. *Then* $\mathrm{Sig}(S_{i-1}) \supseteq \mathrm{Sig}(S_i)$ *for each S_{i-1} in* $\mathrm{Seq}(I)$.

Proof. Let $I = (V, C)$, $(j, a, b) \in \mathrm{Sig}(S_i)$, where $j \in [|V|]$ and $a, b \in D$. Then there exists $t, t' \in S_i$ such that (t, t') witnesses (j, a, b), i.e., $\mathrm{pr}_{1,\ldots,j-1}(t) = \mathrm{pr}_{1,\ldots,j-1}(t')$, and $t[j] = a$, $t'[j] = b$. Since $\langle S_{i-1} \rangle_\phi \supseteq \langle S_i \rangle_\phi \supseteq S_i$, it follows that $t, t' \in \langle S_{i-1} \rangle_\phi$, and hence also that $(j, a, b) \in \mathrm{Sig}(\langle S_{i-1} \rangle_\phi)$. But since S_{i-1} is a representation of $\langle S_{i-1} \rangle_\phi$, $\mathrm{Sig}(S_{i-1}) = \mathrm{Sig}(\langle S_{i-1} \rangle_\phi)$, from which we infer that $(j, a, b) \in \mathrm{Sig}(S_{i-1})$. □

Lemma 2. *Let ϕ be a Maltsev operation over a finite domain D, and $R \in \mathrm{Inv}(\{\phi\})$. For every $i \in [\mathrm{ar}(R)]$, the tuples (i, a, b) in $\mathrm{Sig}(R)$ define an equivalence relation on $\mathrm{pr}_i(R) \subseteq D$.*

Proof. Define the relation $a \sim b$ if and only if $(i, a, b) \in \mathrm{Sig}(R)$. Note that $(i, a, a) \in \mathrm{Sig}(R)$ if and only if $a \in \mathrm{pr}_i(R)$, and that $(i, a, b) \notin \mathrm{Sig}(R)$ for any b if $a \notin \mathrm{pr}_i(R)$. Also note that $\sim$ is symmetric by its definition. It remains to show transitivity. Let $(i, a, b) \in \mathrm{Sig}(R)$ be witnessed by (t_a, t_b) and $(i, a, c) \in \mathrm{Sig}(R)$ be witnessed by (t'_a, t'_c). We claim that $t_c := \phi(t_a, t'_a, t'_c) \in R$ is a tuple such that (t_b, t_c) witnesses $(i, b, c) \in \mathrm{Sig}(R)$. Indeed, for every $j < i$ we have $\phi(t_a[j], t'_a[j], t'_c[j]) = \phi(t_a[j], t'_a[j], t'_a[j]) = t_a[j]$, whereas $\phi(t_a[i], t'_a[i], t'_c[i]) = (a, a, c) = c$. Since $t_a[j] = t_b[j]$ for every $j < i$, it follows that (t_b, t_c) witnesses $(i, b, c) \in \mathrm{Sig}(R)$. Hence $\sim$ is an equivalence relation on $\mathrm{pr}_i(R)$. □

We remark that each equivalence relation in Lemma 2 is a so-called *congruence* of the Maltsev operation ϕ. Congruences and the lattices induced by ordering congruences by inclusion, *congruence lattices*, are a well-studied topic within universal algebra [10, Section 2.5], and we will now see that the height of the congruence lattice can be used to bound the chain length of $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$.

Theorem 7. *Let ϕ be a Maltsev operation over a finite domain D. Then* $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$ *has chain length $O(|D||V|)$.*

Proof. Let $I = (V, C)$ be an instance of $\mathrm{CSP}(\mathrm{Inv}(\{\phi\}))$, with $|V| = n$ and $|C| = m$, and let $\mathrm{Seq}(I) = (S_0, S_1, \ldots, S_m)$ be the sequence of compact representations computed by the Maltsev algorithm. By Lemma 1, $\mathrm{Sig}(S_{i+1}) \subseteq \mathrm{Sig}(S_i)$ for every $i < m$, and by Lemma 2, the sets $(j, a, b) \in \mathrm{Sig}(S_i)$ induce an equivalence relation on $\mathrm{pr}_j(\langle S_i \rangle_\phi)$ for every $i \leq m$, $j \leq n$. (Lemma 2 applies here since $\mathrm{Sig}(S_i) = \mathrm{Sig}(\langle S_i \rangle_\phi)$ for every S_i in $\mathrm{Seq}(I)$, and $\langle S_i \rangle_\phi \in \mathrm{Inv}(\{\phi\})$.) We also note that if $\mathrm{Sig}(S_{i+1}) = \mathrm{Sig}(S_i)$, then $\langle S_i \rangle_\phi = \langle S_{i+1} \rangle_\phi$ since S_{i+1} is a compact representation of $\langle S_i \rangle_\phi$. Hence, we need to bound the number of times that $\mathrm{Sig}(S_{i+1}) \subset \mathrm{Sig}(S_i)$ can hold. Now note that whenever $\mathrm{Sig}(S_{i+1}) \subset \mathrm{Sig}(S_i)$, then either $\mathrm{pr}_j(\langle S_i \rangle_\phi) \subset \mathrm{pr}_j(\langle S_{i+1} \rangle_\phi)$ for some j, or the equivalence relation induced by tuples $(j, a, b) \in \mathrm{Sig}(S_{i+1})$ is a refinement of that induced by tuples $(j, a, b) \in \mathrm{Sig}(S_i)$ for some j. Both of these events can only occur $|D| - 1$ times for every position j (unless $S_m = \emptyset$). Hence the chain length is bounded by $2|V||D|$. □

This bound can be slightly improved for a particular class of Maltsev operations. Recall from Example 3 that $s(x, y, z) = x \cdot y^{-1} \cdot z$ is the coset generating operation of a group $G = (D, \cdot)$.

Lemma 3. *Let $G = (D, \cdot)$ be a finite group and let s be its coset generating operation. Then $\mathrm{CSP}(\mathrm{Inv}(\{s\}))$ has chain length $O(|V| \log |D|)$.*

Proof. Let $I = (V, C)$ be an instance of $\mathrm{CSP}(\mathrm{Inv}(\{s\}))$, where $|V| = n$ and $|C| = m$. Let $\mathrm{Seq}(I) = (S_0, S_1, \ldots, S_m)$ be the corresponding sequence. First observe that S_0 is a compact representation of D^n and that $(D^n, \cdot)$ is nothing else than the nth direct power of G. It is well-known that R is a coset of a subgroup of $(D^n, \cdot)$ if and only if s preserves R [14]. In particular, this implies that S_1 is a compact representation of a subgroup of $(D^n, \cdot)$, and more generally that each S_i is a compact representation of a subgroup of $\langle S_{i-1} \rangle_s$. An application of Lagrange's theorem reveals that $|\langle S_i \rangle_s|$ divides $|\langle S_{i-1} \rangle_s|$, which implies that the sequence $\langle S_0 \rangle_s, \langle S_1 \rangle_s, \ldots, \langle S_m \rangle_s$ contains at most $n \log_2 |D| + 1$ distinct elements. □

Note that if the domain $|D|$ is prime in Lemma 3 then the proof can be strengthened to obtain the bound $O(|V|)$.

Example 7. *Let us briefly return to Example 5, where we demonstrated that $R_{1/k}$ had a Maltsev extension over the coset generating operation of an Abelian group $(D, +)$ where $|D|$ is prime. Combining Theorem 6 and Lemma 3 we therefore conclude that $\mathrm{SAT}(\{R_{1/k}\})$ has a kernel with $O(|V|)$ constraints.*

More generally, we may interpret the results in this section as follows. If Γ admits a Maltsev extension over the coset generating operation of an Abelian group $(D, +)$, where $|D|$ is prime, then we obtain kernels with $O(|V|)$ constraints, closely mirroring the results from Jansen and Pieterse [24]. This is in turn a special case of constraint languages admitting Maltsev extensions over coset generating operations over arbitrary groups, where we obtain kernels with $O(|V| \log |D|)$ constraints. It is not hard to find examples of groups whose coset generating operations cannot be represented by the aforementioned Abelian groups. One such example is the group A_n of all even permutations over $[n]$ for $n \geq 3$. Last, in the most general case, where we obtain kernels with $O(|V||D|)$ constraints, we have extensions over arbitrary Maltsev operations. Furthermore, it is known that a Maltsev operation ϕ over D is the coset generating operation of a group $(D, \cdot)$ if and only if

$$\phi(\phi(x, y, z), z, u) = \phi(x, y, u)$$

and

$$\phi(u, z, \phi(z, y, x)) = \phi(u, y, x)$$

for all $x, y, z, u \in D$ [14]. Hence, any Maltsev operation not satisfying either of these two identities cannot be viewed as a coset generating operation of some group.[3]

4 Kernels of Polynomial Size

Section 3.2 gives a description of CSP problems admitting kernels with $O(n)$ constraints. In this section we study two generalizations which provide kernels with $O(n^c)$ constraints for $c > 1$.

[3] Recently, Chen et al. have shown that any Boolean language with a Maltsev extension over a group as above, also admits a Maltsev extension as equations over integer rings [11]. However, the general question of whether Maltsev extensions have greater expressive power than standard linear equations remains unanswered.

4.1 Moving Beyond Maltsev: k-Edge Extensions

It is known that Maltsev operations are particular examples of a more general class of operations called *k-edge operations*. Following Idziak et al. [2] we define a k-edge operation e as a $(k+1)$-ary operation satisfying

$$e(x,x,y,y,y,\ldots,y,y) = e(x,y,x,y,y,\ldots,y,y) = y$$

and for each $i \in \{4,\ldots,k+1\}$

$$e(y,\ldots,y,x,y,\ldots,y) = y,$$

where x occurs in position i. Note that a Maltsev operation is nothing else than a 2-edge operation with the first and second arguments permuted. A *k-edge extension* is then defined analogously to the concept of a Maltsev extension, with the distinction that the extension $\hat{\Gamma}$ must be preserved by a k-edge operation for some $k \geq 2$. It is known that k-edge operations satisfy many of the advantageous properties of Maltsev operations, and the basic definitions concerning signatures and representations are similar. Before the proof of Theorem 9 we need the following lemma from Idziak et al. [2, Lemma 2.13].

Lemma 4 ([2])**.** *If e is a k-edge operation over D then $[\{e\}]$ also contains a binary operation d and a ternary operation p satisfying*

$$p(x,y,y) = x, p(x,x,y) = d(x,y), d(x,d(x,y)) = d(x,y)$$

for all $x,y \in D$ and a k-ary operation s, which for all $x,y \in D$ satisfies

$$s(x,y,y,y,\ldots,y,y) = d(y,x)$$

and for each $i \in \{2,\ldots,k\}$,

$$s(y,y,\ldots,y,x,y,\ldots,y) = y,$$

where x appears in position i.

If e is a k-edge operation over D and d the operation in Lemma 4 then $(a,b) \in D^2$ is a *minority pair* if $d(a,b) = b$. Given an n-ary relation $R \in \mathrm{Inv}(\{e\})$ and $t,t' \in R$ we then say that the index $(i,a,b) \in [n] \times D^2$ *witnesses* (t,t') if (a,b) is a minority pair, $\mathrm{pr}_{1,\ldots,i-1}(t) = \mathrm{pr}_{1,\ldots,i-1}(t')$, and $t[i] = a$, $t'[i] = b$. We let $\mathrm{Sig}_e(R)$ denote the set of all indexes witnessing tuples of the relation $R \in \mathrm{Inv}(\{e\})$. Last, $R' \subseteq R$ is a *representation* of R if (1) $\mathrm{Sig}_e(R) = \mathrm{Sig}_e(R')$ and (2) for every $i_1,\ldots,i_{k'} \in [n]$, $k' < k$, $\mathrm{pr}_{i_1,\ldots,i_{k'}}(R) = \mathrm{pr}_{i_1,\ldots,i_{k'}}(R')$. Similar to the Maltsev case we have the following useful property of representations of relations invariant under k-edge operations.

Theorem 8 ([2])**.** *Let e be a k-edge operation over a finite domain, $R \in \mathrm{Inv}(\{e\})$ a relation, and R' a representation of R. Then $\langle R' \rangle_e = R$.*

Moreover, each n-ary relation invariant under a k-edge operation has a compact representation of size $O(n^{k-1})$. By this stage it should not come as a surprise to the reader that Maltsev algorithm outlined in Section 3.1 can be modified to solve $\mathrm{CSP}(\mathrm{Inv}(\{e\}))$ in polynomial time. We will refer to this algorithm as the *few subpowers* algorithm [22]. We then obtain the following, analogous to the Maltsev case from Section 3.2.

Theorem 9. *Let Γ be a constraint language which admits a polynomially bounded k-edge extension $\hat{\Gamma}$ over a finite domain D. Then $\mathrm{CSP}(\Gamma)$ has a kernel with $O(|D|^{k-1}|V|^{k-1})$ constraints.*

Proof. We only provide a proof sketch since the details are very similar to the Maltsev case. Assume $k \geq 3$, since otherwise the bound follows from Theorem 6 and 7, and let e denote the k-edge operation witnessing the extension $\hat{\Gamma}$. Given an instance $I = (V, \{C_1,\ldots,C_m\})$ of $\mathrm{CSP}(\Gamma)$, iteratively compute compact representations $S_0, S_1, \ldots, S_m$ of the solution space of $(V, \emptyset)$, $(V, \{C_1\})$, …, $(V, \{C_1,\ldots,C_m\})$. This can be done in polynomial time using the procedures from the few subpowers algorithm [22]. We then remove the constraint C_i if and only if $\langle S_i \rangle_e = \langle S_{i-1} \rangle_e$, which can be checked in polynomial time using arguments similar to those in Theorem 6.

All that remains to be proven is therefore that the number of distinct elements in the sequence $\langle S_0 \rangle_e, \langle S_1 \rangle_e, \ldots, \langle S_m \rangle_e$ is bounded by $O(|D|^{k-1}|V|^{k-1})$. For each S_i define

$$\mathrm{Proj}(S_i) = \{(J,R) \mid J \in [|V|]^j, R \subseteq D^j, j < k, \mathrm{pr}_J(S_i) = R\}.$$

If $\langle S_i \rangle_e \supset \langle S_{i-1} \rangle_e$ it can then be proven that either $\mathrm{Sig}_e(S_i) \supset \mathrm{Sig}_e(S_{i-1})$ or $\mathrm{Proj}(S_i) \supset \mathrm{Proj}(S_{i-1})$. This gives the bound $1 + |\mathrm{Sig}(D^n)| + |\mathrm{Proj}(D^n)| = O(|D|^{k-1}|V|^{k-1})$. □

4.2 Bounded-Degree Extensions

We now consider an alternative technique for obtaining kernels with $O(n^c)$ constraints, $c > 1$, which is useful for classes of languages that do not admit Maltsev or k-edge extensions. This will generalize the results on kernelization for constraints defined via non-linear polynomials over finite fields [24]. In the rest of this subsection, we assume that the language Γ is Boolean.

Definition 8. Let $c \in \mathbb{N}$ be a constant, $c \geq 2$, and for a set V let $V^{(c)} = \{S_1, \ldots, S_l\}$ be an enumeration of all subsets of V of size at most c in some fixed order $S_1, \ldots, S_l$. We make the following definitions.

1. Let $t \in \{0,1\}^r$ be a tuple of arity r and let $[r]^{(c)} = \{S_1, \ldots, S_l\}$. A tuple $\check{t} \in \{0,1\}^l$ is the *degree-c extension of* t with respect to the ordering $S_1, \ldots, S_l$ if $\check{t}[i] = \prod_{j \in S_i} t[j]$, $i \in [l]$.

2. A *degree-c extension of* Γ is a language $\check{\Gamma}$ with a bijection h between relations $R \in \Gamma$ and relations $\check{R} \in \check{\Gamma}$ such that for every $R \in \Gamma$ and for every tuple $t \in \{0,1\}^{\mathrm{ar}(R)}$, $t \in R$ if and only if $\check{t} \in \check{R}$ where $\check{t}$ is the degree-c extension of t with respect to some fixed ordering.

Note that, whereas tuples $t \in \{0,1\}^r$ have unique degree extensions $\check{t}$ up to the ordering of the sets S_l, a relation $R \subseteq \{0,1\}^r$ will have many degree-c extensions for $c > 1$, since it is not determined whether $t' \in \check{R}$ for tuples $t' \in \{0,1\}^{\mathrm{ar}(\check{R})}$ that are not extensions of tuples $t \in \{0,1\}^r$.

We now give the kernelization applications of degree extensions. Let $I = (V, C)$ be a $\mathrm{SAT}(\Gamma)$ instance for a Boolean constraint language Γ. Let $V^{(c)}$ be defined as above, for some constant $c \geq 2$, and from any assignment $g : V \to \{0,1\}$ define an assignment $g' : V^{(c)} \to \{0,1\}$ as $g'(S) := \prod_{v \in S} g(v)$ for every set $S \in V^{(c)}$. Degree-c extensions, Maltsev extensions and k-edge extensions are related by the following theorem.

Theorem 10. *Let Γ be a finite Boolean language and $\check{\Gamma}$ a degree-c extension of Γ. If $\check{\Gamma}$ admits a Maltsev extension, then* $\mathrm{SAT}(\Gamma)$ *admits a kernel of $O(n^c)$ constraints; if $\check{\Gamma}$ admits a k-edge extension, then* $\mathrm{SAT}(\Gamma)$ *admits a kernel of $O(n^{(k-1)c})$ constraints.*

Proof. Since Γ is finite and fixed, we assume that both extensions are efficiently computable. Let $I = (V, C)$, $|V| = n$, be an instance of $\mathrm{SAT}(\Gamma)$, and let $V^{(c)}$ be the degree-c extension of V. For each constraint $R(x_1, \ldots, x_m)$, $m = \mathrm{ar}(R)$, let $X_1, \ldots, X_l \in V^{(c)}$ denote the subsets of $\{x_1, \ldots, x_m\}$ of size at most c, and replace $R(x_1, \ldots, x_m)$ by the constraint $\check{R}(X_1, \ldots, X_l)$. Let I' be the instance of $\mathrm{SAT}(\check{\Gamma})$ resulting from repeating this for every constraint in the instance. Observe that if g is a satisfying assignment to I then $g'(X) = \prod_{x \in X} g(x)$, $X \in V^{(c)}$, is a satisfying assignment to I'. We now apply the kernelization for languages with Maltsev extensions, respectively k-edge extensions, to I', and let $I'' = (V, C')$ where $C' \subseteq C$ is the set of constraints kept by the kernelization. Note that the contents of the relation Ψ_I defined by I correspond directly to the relation $\{\check{t} \cap \Psi_{I'} \mid t \in \{0,1\}^n\}$ (recall from Section 3.2 that Ψ_I is the set of all satisfying assignments to the instance I). Since the kernelizations we use preserve the entire solution space, this kernelization procedure is sound, and the desired bound for the number of constraints in the output follows from Theorem 9. □

We observe that this captures the class of SAT problems which can be written as roots of low-degree polynomials from Jansen and Pieterse [24].

Theorem 11. *Let Γ be a Boolean language such that every relation $R \in \Gamma$ can be defined as the set of solutions in $\{0,1\}$ to a polynomial of degree at most d, over some fixed finite field F. Then Γ admits a degree-d extension with a Maltsev extension.*

Proof. We sketch the most important ideas. Let $G_1 = (D, \cdot)$ and $G_2 = (D, +)$ be the two Abelian groups representing the field F. For $R \in \Gamma$, let p_R be the polynomial defining R. Then p_R can be written as a sum of monomials over G_1, and each such monomial can simply be treated as conjunction of variables since p_R is evaluated over the Boolean domain, and thus corresponds to a member of V^d for $d \geq 1$. Hence, the extension $\check{R}$ of R can be written as a linear sum over G_2, and similar to Example 4 it is now clear that the coset generating operation of G_2 will preserve the resulting Maltsev extension. The result then follows from Theorem 10. □

Finally, we observe that the approach of k-edge extensions by itself is insufficient to prove tight polynomial kernel bounds for $\mathrm{SAT}(\Gamma)$ problems.

Lemma 5. *For every $k \geq 3$ there exists a finite Boolean language Γ such that* $\mathrm{SAT}(\Gamma)$ *is NP-hard and admits a kernel with $O(n^2)$ constraints, but Γ does not admit a k-edge extension.*

Proof. Let $R(x_1, \ldots, x_k)$ be the set of Boolean roots to the quadratic polynomial $p(x_1, \ldots, x_k) = x_1 + \ldots + x_k - x_1x_2 - 1$, evaluated over $\mathbb{Z}_p$ for some prime $p > k$. If we let $\Gamma = \{R\}$ it can then be verified that $\mathrm{SAT}(\Gamma)$ is NP-hard since it does not fall into one of Schaefer's tractable cases [39]. However, since R is described as the roots of a quadratic equation over a finite field, $\mathrm{SAT}(\Gamma)$ has a kernel with $O(n^2)$ constraints, by Theorem 11. On the other hand, assume towards a contradiction that Γ has an extension into a language $\hat{\Gamma}$ preserved by a k-edge operation e. By Theorem 5, Γ must then be preserved by the partial operation $e_{|\mathbb{B}}$. Let $t_1 = (1, 1, 0, \ldots, 0)$ and for $i = 2, \ldots, k+1$ let $t_i = (0, \ldots, 1, \ldots, 0)$ be the tuple with 1 in entry $i - 1$ and 0 in all other entries. It is then easy to verify that

$$e_{|\mathbb{B}}(t_1, \ldots, t_{k+1}) = (0, \ldots, 0)$$

is defined since e is a k-edge operation. Since $t_i \in R$ for every $i \in [k+1]$ but $(0, \ldots, 0) \notin R$, we have a witness against R being preserved by $e_{|\mathbb{B}}$, and since e was arbitrary, we conclude that Γ does not admit a k-edge extension. □

5 Universal Partial Maltsev Operations and Lower Bounds

We have seen that Maltsev extensions and, more generally, k-edge extensions, provide an algebraic criterion for determining that a $\mathrm{CSP}(\Gamma)$ problem admits a kernel of a fixed size. In this section we demonstrate that our approach can also be used to give lower bounds for the kernelization complexity of Boolean constraint languages. More specifically, we will use the fact that if a satisfiability problem $\mathrm{SAT}(\Gamma)$ admits a Maltsev extension, then this can be witnessed by certain canonical partial operations preserving Γ. We begin in Section 5.1 by studying properties of these canonical partial operations, *universal partial Maltsev operations*, and in Section 5.2 prove that the absence of these operations can be used to prove lower bounds on kernelizability.

5.1 Universal Partial Maltsev Operations

In this section we will study properties of partial operations preserving every Boolean constraint language admitting a Maltsev extension. We thus begin by making the following definition.

Definition 9. A Boolean partial operation f is a *universal partial Maltsev operation* if $f \in \mathrm{pPol}(\Gamma)$ for every Boolean Γ admitting a Maltsev extension.

In addition, we let UPM denote the set of all Boolean partial universal Maltsev operations, i.e., if we first let $X = \{\Gamma \mid \Gamma \subseteq BR \,\text{admits a Maltsev extension}\}$ we then define

$$\mathrm{UPM} = \bigcap_{\Gamma \in X} \mathrm{pPol}(\Gamma).$$

Note that UPM is a strong partial clone since it is defined as the intersection of a set of strong partial clones. We now proceed and give a complete characterization of the universal partial Maltsev operations. This characterization will show that there exists an operation u defined over an infinite domain such every Boolean Γ admitting a Maltsev extension also admits an extension with respect to u, which in particular will show that the universal partial Maltsev operations can be described by nested applications of u. This operation is defined as follows.

Definition 10. Let the infinite domain D_∞ be recursively defined to contain 0, 1, and ternary tuples of the form (x, y, z) where $x, y, z \in D_\infty$ and $x \neq y$, $y \neq z$. The ternary Maltsev operation u over D_∞ is defined as $u(x, x, y) = y, u(x, y, y) = x$, and $u(x, y, z) = (x, y, z)$ otherwise.

Now recall from Section 3.2 that if Γ admits a Maltsev extension with respect to a Maltsev operation ϕ then Γ is preserved by every partial operation $q_{|\mathbb{B}}$ for $q \in [\{\phi\}]$. Hence, our aim is to show that every universal partial Maltsev operation is of the form $q_{|\mathbb{B}}$ for $q \in [\{u\}]$. Before presenting this proof we need some additional notation. It is well-known that if $[F]$ is a clone over a domain D then $f \in [F]$ if and only if f is definable as a term function over the algebra (D, F) [20]. Given a term $T(x_1, \ldots, x_n)$ over an algebra (D, F) defining an operation $g \in [F]$ and $b_1, \ldots, b_n \in D$, we let $\mathrm{Val}(T(b_1, \ldots, b_n)) = g(b_1, \ldots, b_n)$.

Theorem 12. *Let $q \in [\{u\}]$. Then $q_{|\mathbb{B}} \in$ UPM.*

Proof. Let Γ be a Boolean constraint language which admits a Maltsev extension $\hat{\Gamma}$. We will prove that $q_{|\mathbb{B}} \in \mathrm{pPol}(\Gamma)$, which is sufficient to prove the claim since Γ was choosen arbitrarily. Let p be the Maltsev operation witnessing the extension $\hat{\Gamma}$, let n denote the arity of q, and let $q(x_1, \ldots, x_n) = T^u(x_1, \ldots, x_n)$ where T^u is the term over u defining q. Now, first consider the operation $q' \in [\{p\}]$ obtained by replacing each occurrence of u with p in the term $T^u(x_1, \ldots, x_n)$. Let $T^p(x_1, \ldots, x_n)$ denote this term over p, and for each term $T_i^u(\mathbf{x_i})$ occurring as a subterm in $T^u(x_1, \ldots, x_n)$ we let $T_i^p(\mathbf{x_i})$ denote the corresponding term over p.

Now observe that the partial operation $q'_{|\mathbb{B}}$ is included in $\mathrm{pPol}(\Gamma)$ via Lemma 5. We claim that $q_{|\mathbb{B}}$ can be obtained as a subfunction of $q'_{|\mathbb{B}}$, which implies that $q_{|\mathbb{B}} \in \mathrm{pPol}(\Gamma)$, since a strong partial clone is always closed under taking subfunctions. By definition, we have that $(b_1, \ldots, b_n) \in \mathrm{domain}(q_{|\mathbb{B}})$ if and only if $b_1, \ldots, b_n \in \{0, 1\}$ and $q(b_1, \ldots, b_n) \in \{0, 1\}$.

We will prove that for each sequence of Boolean arguments $b_1, \ldots, b_n$, if $q(b_1, \ldots, b_n) = b \in \{0, 1\}$ then $q'(b_1, \ldots, b_n) = b$. First, let us illustrate the intuition behind this by an example. Assume that $n = 7$ and that $T^u(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = u(u(x_1, x_2, x_3), u(x_4, x_5, x_6), x_7)$. In this case we will e.g. have that

$$\mathrm{Val}(T^u(0, 1, 0, 0, 1, 0, 1)) = 1$$

since

$$u(u(0, 1, 0), u(0, 1, 0), 1) = u((0, 1, 0), (0, 1, 0), 1) = 1,$$

due to the fact that u always respect the Maltsev identities. But since p is also a Maltsev operation it must also be the case that $\mathrm{Val}(T^p(0, 1, 0, 0, 1, 0, 1)) = 1$, even if $u(0, 1, 0)$ and $p(0, 1, 0)$ might differ.

The general case can be proven by a case inspection of the term T^u. First, assume that T^u contains a term of the form $u(x_{i_1}, x_{i_2}, x_{i_3})$. If $b_{i_1}, b_{i_2}, b_{i_3} \in \{0, 1\}$ then $u(b_{i_1}, b_{i_2}, b_{i_3}) \in \{0, 1\}$ if and only if $b_{i_1} = b_{i_2}$ or $b_{i_2} = b_{i_3}$. But this implies that $p(b_{i_1}, b_{i_2}, b_{i_3}) = u(b_{i_1}, b_{i_2}, b_{i_3})$ since p is Maltsev. Second, assume that T^u contains a term of the form $u(T_1^u(\mathbf{x_1}), T_2^u(\mathbf{x_2}), T_3^u(\mathbf{x_3}))$ where $\mathbf{x_1}, \mathbf{x_2}$ and $\mathbf{x_3}$ are tuples of variables over $x_1, \ldots, x_n$. Let $\mathbf{b_1}$, $\mathbf{b_2}$ and $\mathbf{b_3}$ be Boolean tuples matching the length of $\mathbf{x_1}$, $\mathbf{x_2}$ and $\mathbf{x_3}$, and assume that $\mathrm{Val}(T_1^u(\mathbf{b_1})) = \mathrm{Val}(T_1^p(\mathbf{b_1}))$, $\mathrm{Val}(T_2^u(\mathbf{b_2})) = \mathrm{Val}(T_2^p(\mathbf{b_2}))$ and $\mathrm{Val}(T_3^u(\mathbf{b_3})) = \mathrm{Val}(T_3^p(\mathbf{b_3}))$. Similarly to the first case we have that

$$u(\mathrm{Val}(T_1^u(\mathbf{b_1})), \mathrm{Val}(T_2^u(\mathbf{b_2})), \mathrm{Val}(T_3^u(\mathbf{b_3}))) \in \{0, 1\}$$

if and only if $\mathrm{Val}(T_1^u(\mathbf{b_1})) = \mathrm{Val}(T_2^u(\mathbf{b_2}))$ or $\mathrm{Val}(T_2^u(\mathbf{b_2})) = \mathrm{Val}(T_3^u(\mathbf{b_3}))$, and since p is Maltsev this implies that

$$p(\mathrm{Val}(T_1^p(\mathbf{b_1})), \mathrm{Val}(T_2^p(\mathbf{b_2})), \mathrm{Val}(T_3^p(\mathbf{b_3}))) = u(\mathrm{Val}(T_1^u(\mathbf{b_1})), \mathrm{Val}(T_2^u(\mathbf{b_2})), \mathrm{Val}(T_3^u(\mathbf{b_3}))).$$

Hence, for each $(b_1, \ldots, b_n) \in \mathrm{domain}(q_{|\mathbb{B}})$ we have that $(b_1, \ldots, b_n) \in \mathrm{domain}(q'_{|\mathbb{B}})$ and that $q_{|\mathbb{B}}(b_1, \ldots, b_n) = q'_{|\mathbb{B}}(b_1, \ldots, b_n)$. This implies that $q_{|\mathbb{B}}$ is a subfunction of $q'_{|\mathbb{B}}$, that $q_{|\mathbb{B}} \in \mathrm{pPol}(\Gamma)$, and, finally, that $q_{|\mathbb{B}}$ is a universal partial Maltsev operation. □

Using Theorem 12 we can now prove that every Boolean language Γ invariant under the universal partial Maltsev operations admits a Maltsev extension over D_∞.

Theorem 13. *Let Γ be a Boolean constraint language. Then* UPM $\subseteq \mathrm{pPol}(\Gamma)$ *if and only if Γ has a Maltsev extension $\hat{\Gamma}$ over D_∞.*

Proof. For the first direction, let u be the Maltsev operation from Definition 10 over the infinite domain D_∞. For each relation $R \in \Gamma$ we let $\hat{R} = \langle R \rangle_u$. Let $\hat{\Gamma}$ denote the resulting constraint language over D_∞. By definition, $u \in \mathrm{Pol}(\hat{\Gamma})$, and everything that remains to be proven is that $\hat{R} \cap \{0, 1\}^{\mathrm{ar}(R)} = R$ for each $\hat{R} \in \hat{\Gamma}$. Hence, assume that there exists at least one tuple $t \in (\hat{R} \cap \{0, 1\}^{\mathrm{ar}(R)}) \setminus R$. This implies that there exists a term T over u such that $\mathrm{Val}(T(t_1[i], \ldots, t_m[i])) = t[i]$ for each $i \in [\mathrm{ar}(R)]$, where $R = \{t_1, \ldots, t_m\}$. Let q denote the function corresponding to the term T and observe that $q \in [\{u\}]$. According to Theorem 12 this implies that $q_{|\mathbb{B}}$ is a universal partial Maltsev operation and, furthermore, that $q_{|\mathbb{B}}(t_1[i], \ldots, t_m[i])$ is defined for each $i \in [\mathrm{ar}(R)]$, since $q(t_1[i], \ldots, t_m[i]) \in \{0, 1\}$. Hence, $q_{|\mathbb{B}}(t_1, \ldots, t_m) = t \notin R$, which contradicts the assumption that Γ was invariant under all universal partial Maltsev operations.

The second direction is trivial since if Γ has a Maltsev extension over D_∞ then Γ by definition is preserved by every universal partial Maltsev operation. $\square$

Moreover, Theorem 12 and Theorem 13 implies that every universal partial Maltsev operation can be described via Theorem 12. Hence, we have obtained a complete understanding of the universal partial Maltsev operations.

Corollary 1. *Let f be a Boolean partial operation. Then $f \in \mathrm{UPM}$ if and only if $f = q_{|\mathbb{B}}$ for $q \in [\{u\}]$.*

Hence, we have obtained a complete characterization of universal partial Maltsev operations, in terms of restrictions of term functions over u. We now proceed by studying additional properties of the strong partial clone UPM, and will not only give a generating set of UPM, but also show that any such generating set needs to be infinite.

Definition 11. Let $u^1 = u$ and fix $d \geq 1$. We define

1. $u^{d+1}(x_1, \ldots, x_{3^d}, y_1, \ldots, y_{3^d}, z_1, \ldots, z_{3^d}) = u(u^d(x_1, \ldots, x_{3^d}), u^d(y_1, \ldots, y_{3^d}), u^d(z_1, \ldots, z_{3^d}))$, and
2. $\phi_d = u^d_{|\mathbb{B}}$.

We refer to ϕ_d as the *dth universal partial Maltsev operation.* We will now prove that these operations are expressive enough to generate all other universal partial Maltsev operations.

Theorem 14. $[\{\phi_1, \phi_2, \ldots\}]_s = \mathrm{UPM}$.

Proof. Let $f \in \mathrm{UPM}$ be an n-ary universal partial Maltsev operation. By Theorem 13 there exists an operation $q \in [\{u\}]$ such that $f = q_{|\mathbb{B}}$. Let T^u be the term over u defining q and let $d \geq 1$ be the maximum depth of T_u. We claim that $f \in [\{\phi_d\}]_s$. To prove this we will show that f can be reconstructed from ϕ_d by gradually transforming the term defining ϕ_d. Hence, let S^u denote the term from Definition 11 and consider the following recursive procedure.

1. Let $e \leq d$ denote the current depth.
2. Let $u(T_1, T_2, T_3)$ be the current term at depth e in T^u and $u(S_1, S_2, S_3)$ be the term in the corresponding position in S^u.
3. If $T_i = x_j$ for a single variable x_j, then identify the tuple of variables occurring in S_i with x_j.
4. Otherwise recursively apply the procedure for T_i and increase e by one.

It is then straightforward to see that the resulting term over u defines an operation g whose restriction $g_{|\mathbb{B}}$ is exactly f, implying that $f \in [\{\phi_d\}]_s$. $\square$

In particular the proof implies that $\phi_e \in [\{\phi_d\}]_s$ for every $e \leq d$. Hence, the set of universal partial Maltsev operations can be generated by the sequence $\phi_1, \phi_2, \ldots$ of increasingly stronger operations. We will now prove that this sequence of operations in fact strictly increases in expressive power and that no finite set of (strictly partial) operations can generate UPM. Thus, say that a strong partial clone $\mathrm{pPol}(\Gamma)$ is *finitely generated* if there exists a finite set of partial operations F such that $[F]_s = \mathrm{pPol}(\Gamma)$.

Theorem 15. UPM *is not finitely generated.*

Proof. Assume that there exists a finite set of partial universal Maltsev operations M such that $[M]_s = \mathrm{UPM}$. By Theorem 14 there exists a d such that $M \subseteq [\{\phi_1, \ldots, \phi_d\}]_s$, implying that $[\{\phi_1, \ldots, \phi_d\}]_s = \mathrm{UPM}$. We will prove that $\phi_{3^d} \notin [\{\phi_d\}]_s$ for every $d \geq 1$, which by Theorem 14 is sufficient to contradict this assumption. To accomplish this we first define a function $f : \mathbb{N} \to \mathbb{N}$ with the property that if f is unbounded, i.e., for every $n \in \mathbb{N}$ there exists $m \in \mathbb{N}$ such that $f(n) < f(m)$, then $\phi_{3^d} \notin [\{\phi_d\}]_s$ for every $d \geq 1$. This function is defined as follows: $f(d)$ is equal to the largest $1 \leq n \leq 3^d$ such that there for every tuple $t \in \{0,1\}^{3^d}$ and indices $i_1, \ldots, i_e \in [3^d]$, $e \leq n$, exist $t_1, t_2 \in \mathrm{domain}(\phi_d)$ such that $\mathrm{pr}_{i_1,\ldots,i_e}(t_1) = \mathrm{pr}_{i_1,\ldots,i_e}(t_2) = \mathrm{pr}_{i_1,\ldots,i_e}(t)$, and $\phi_d(t_1) = 0$ and $\phi_d(t_2) = 1$. The intuition behind this function is that it, given input d, returns the largest possible n such that whenever we fix at most n values, it is possible to find tuples in the domain of ϕ_d matching these fixed values, and the result of applying ϕ_d to these tuples is 0 and 1, respectively.

It can be verified that the first values are $f(1) = 1$ and $f(2) = 3$. We prove by induction that $f(d) \geq d$ for any $d \geq 3$, which is sufficient to show that f is unbounded. Let $f(d) = n$ and consider the term over u corresponding to ϕ_d from Definition 11. At the uppermost level this term has the form

$$u(T_1(x_1, \ldots, x_{3^{d-1}}), T_2(y_1, \ldots, y_{3^{d-1}}), T_3(z_1, \ldots, z_{3^{d-1}})).$$

Now assume that we are given indices $i_1, \ldots, i_e \in [3^d]$, $e \leq n$. Regardless of the values assigned to the corresponding variables, we need to prove that ϕ_d can be evaluated to both 0 and 1. First, if $e < d$ then the inductive hypothesis is applicable and we are already done. Therefore assume that $e = d$. In this case it is easy to see that all of the fixed values belong to exactly one of the subterms T_1, T_2, T_3 since the inductive hypothesis is applicable otherwise. Assume without loss of generality that this subterm is T_1. It is then clear that if $y_{i_1}, \ldots, y_{i_e}$ are assigned the same values as $x_{i_1}, \ldots, x_{i_e}$, then regardless of the values of the other variables, the result of evaluating the two terms will result in the same tuple. Since u is a Maltsev operation this implies that the output of ϕ_d in this case only depends on the third subterm T_3, which, for example, will output 0 if $z_1, \ldots, z_{3^d}$ are assigned 0, and 1 if $z_1, \ldots, z_{3^d}$ are assigned 1.

For the second part of the proof, fix $d \geq 1$, and observe that $f(3^d) > f(d)$, as otherwise ϕ_d would be a total operation which is not a projection. We claim that $\phi_{3^d} \notin [\{\phi_d\}]_s$. Assume that $\phi_{3^d} \in [\{\phi_d\}]_s$ and let

$$\phi_{3^d}(x_1, \ldots, x_{3^{3^d}}) = \phi_d(g_1(x_1, \ldots, x_{3^{3^d}}), \ldots, g_{3^d}(x_1, \ldots, x_{3^{3^d}}))$$

be the composition witnessing this, where $g_1, \ldots, g_{3^d} \in [\{\phi_d\}]_s$. It is clear that there exists a g_i, indices $j_1, \ldots, j_{3^d}$, and a subterm of the form $\phi_d(x_{j_1}, \ldots, x_{j_{3^d}})$ in the definition of g_i over ϕ_d. Let X denote the set of variables used in this term, i.e., $X = \{x_{j_1}, \ldots, x_{j_{3^d}}\}$. Now note that the assumption $f(3^d) \geq 3^d \geq |X|$ implies that $\phi_d(x_{j_1}, \ldots, x_{j_{3^d}})$ cannot be undefined for any sequence of arguments. But this would imply that ϕ_d is a total operation which is not a projection, which is impossible. We thus conclude that $\phi_{3^d} \notin [\{\phi_d\}]_s$ for every $d \geq 1$, implying that UPM cannot be finitely generated. □

A more concrete interpretation of Theorem 15 states the following: for any Boolean constraint language Γ, the existence of a Maltsev extension can be determined by checking if $\phi_i \in \text{pPol}(\Gamma)$ for every $i \geq 1$, but in general there is no finite set of partial operations guaranteeing the existence of a Maltsev extension. We briefly return to this question in Section 6.3 where we discuss a meta problem in kernelization.

5.2 Lower Bounds

In this section we will prove that the absence of partial universal Maltsev operations can be used to prove lower bounds on kernelizability for SAT. First, recall from Definition 11 that the first universal partial Maltsev operation ϕ_1 is a ternary partial operation satisfying the two identities $\phi_1(x, y, y) = x$ and $\phi_1(x, x, y) = y$ for all $x, y \in \{0, 1\}$, and note that

$$\text{domain}(\phi_1) = \{(0,0,0), (1,1,1), (0,0,1), (1,1,0), (1,0,0), (0,1,1)\}.$$

Since ϕ_1 is universal by Theorem 12, we can already conclude that ϕ_1 preserves any Boolean Γ admitting a Maltsev extension, but we will shortly see that $\phi_1 \in \text{pPol}(\Gamma)$ is in fact also a necessary condition for the existence of a linear-sized kernel for SAT(Γ), modulo a standard complexity theoretical assumption. A pivotal part of this proof is that if $\phi_1 \notin \text{pPol}(\Gamma)$, then Γ can qfpp-define a relation Φ_1, which can be used as a gadget in a reduction from the Vertex Cover problem. This relation is defined as

$$\Phi_1(x_1, x_2, x_3, x_4, x_5, x_6) \equiv (x_1 \lor x_4) \land (x_1 \neq x_3) \land (x_2 \neq x_4) \land (x_5 = 0) \land (x_6 = 1),$$

and if we explicitly enumerate the tuples of Φ_1 we see that

$$\Phi_1 = \{(0,0,1,1,0,1), (1,1,0,0,0,1), (1,0,0,1,0,1)\},$$

implying that each argument of Φ_1 exactly corresponds to a tuple in domain(ϕ_1). However, as made clear in the following lemma, there is an even stronger relationship between ϕ_1 and Φ_1.

Lemma 6. *If Γ is a Boolean constraint language such that $\langle \Gamma \rangle = BR$ and $\phi_1 \notin \text{pPol}(\Gamma)$ then $\Phi_1 \in \langle \Gamma \rangle_{\not\exists}$.*

Proof. Before the proof we need two central observations. First, the assumption that $\langle \Gamma \rangle = BR$ is well-known to be equivalent to that $\mathrm{Pol}(\Gamma)$ consists only of projections [5]. Second, Φ_1 consists of three tuples which can be ordered as s_1, s_2, s_3 in such a way that there for every $s \in \mathrm{domain}(\phi_1)$ exists $1 \leq i \leq 6$ such that $s = (s_1[i], s_2[i], s_3[i])$.

Now, assume that $\langle \Gamma \rangle = BR$, $\phi_1 \notin \mathrm{pPol}(\Gamma)$, but that $\Phi_1 \notin \langle \Gamma \rangle_{\nexists}$. Then, due to the Galois connection in Theorem 2, there exists an n-ary partial operation $f \in \mathrm{pPol}(\Gamma)$ such that $f \notin \mathrm{pPol}(\{\Phi_1\})$, and $t_1, \ldots, t_n \in \Phi_1$ such that $f(t_1, \ldots, t_n) \notin \Phi_1$. Now consider the value $k = |\{t_1, \ldots, t_n\}|$, i.e., the number of distinct tuples in the sequence. If $n > k$ then it is known that there exists a closely related partial operation g of arity at most k such that $g \notin \mathrm{pPol}(\{\Phi_1\})$ [35], and we may therefore assume that $n = k \leq |\Phi_1| = 3$. Assume first that $1 \leq n \leq 2$. It is then not difficult to see that there for every $t \in \{0,1\}^n$ exists i such that $(t_1[i], \ldots, t_n[i]) = t$. But then it follows that f is in fact a total operation which is not a projection, which is impossible since we assumed that $\langle \Gamma \rangle = BR$. Hence, it must be the case that $n = 3$, and that $\{t_1, t_2, t_3\} = \{s_1, s_2, s_3\} = \Phi_1$. Assume without loss of generality that $t_1 = s_1$, $t_2 = s_2$, $t_3 = s_3$, and note that this implies that $\mathrm{domain}(f) = \mathrm{domain}(\phi_1)$ (otherwise the arguments of f can be described as a permutation of the arguments of ϕ_1). First, we will show that $f(0,0,0) = 0$ and that $f(1,1,1) = 1$. Indeed, if $f(0,0,0) = 1$ or $f(1,1,1) = 0$, it is possible to define a unary total operation f' as $f'(x) = f(x,x,x)$ which is not a projection, since either $f'(0) = 1$ or $f'(1) = 0$. Second, assume there exists $(x,y,z) \in \mathrm{domain}(f)$, distinct from $(0,0,0)$ and $(1,1,1)$, such that $f(x,y,z) \neq \phi_1(x,y,z)$. Without loss of generality assume that $(x,y,z) = (a,a,b)$ for $a, b \in \{0,1\}$, and note that $f(a,a,b) = a$ since $\phi_1(a,a,b) = b$. If also $f(b,b,a) = a$ it is possible to define a binary total operation $f'(x,y) = f(x,x,y)$ which is not a projection, therefore $f(b,b,a) = b$. We next consider the values taken by f on the tuples (b,a,a) and (a,b,b). If $f(b,a,a) = f(a,b,b)$ then we can again define a total, binary operation which is not a projection, and it must hold that $f(b,a,a) \neq f(a,b,b)$. However, regardless of whether $f(b,a,a) = b$ or $f(b,a,a) = a$, it is not difficult to verify that f must be a partial projection. This contradicts the assumption that $f \notin \mathrm{pPol}(\{\Phi_1\})$, and we conclude that $\Phi_1 \in \langle \Gamma \rangle_{\nexists}$. □

We will now use Lemma 6 to give a reduction from the VERTEX COVER problem, since it is known that VERTEX COVER does not admit a kernel with $O(n^{2-\varepsilon})$ edges for any $\varepsilon > 0$, unless $\mathrm{NP} \subseteq \mathrm{co\text{-}NP/poly}$ [15]. Before the reduction we will need the following lemma, stating that if Γ can pp-define every Boolean relation, then for each k, Γ can pp-define the relation which is true if and only if the Hamming weight of its arguments is exactly k, using only a linear number of constraints and existentially quantified variables. Thus, for each n and k, let $H_{n,k}$ denote the relation $\{(b_1, \ldots, b_n) \in \{0,1\}^n \mid b_1 + \ldots + b_n = k\}$.

Lemma 7. *Let Γ be a constraint language such that $\langle \Gamma \rangle = BR$. Then Γ can pp-define $H_{n,k}$ with $O(n+k)$ constraints and $O(n+k)$ existentially quantified variables.*

Proof. We first observe that one can recursively design a circuit consisting of fan-in 2 gates which computes the sum of n input gates as follows. At the lowest level, we split the input gates into pairs and compute the sum for each pair, producing an output of 2 bits for each pair. This can clearly be done with $O(1)$ gates. At every level i above that, we join each pair of outputs from the previous level, of i bits each, into a single output of $i+1$ bits which computes their sum. This can be done with $O(i)$ gates by chaining full adders. Finally, at level $\lceil \log_2 n \rceil$, we will have computed the sum. The total number of gates will be

$$\sum_{i=1}^{\lceil \log_2 n \rceil} (\frac{n}{2^i}) \cdot O(i),$$

and it is a straightforward exercise to show that this sums to $O(n)$. Let $z_1, \ldots, z_{\log_2 n}$ denote the output gates of this circuit. By a standard Tseytin transformation we then obtain an equisatisfiable 3-SAT instance with $O(n)$ clauses and $O(n)$ variables [40]. Next, for each $1 \leq i \leq \log_2 n$, add the unary constraint $(z_i = k_i)$, where k_i denotes the ith bit of k written in binary. Each such unary constraint can clearly be pp-defined with $O(1)$ existentially quantified variables over Γ. We then pp-define each 3-SAT clause in order to obtain a pp-definition of R over Γ, which in total only requires $O(n)$ existentially quantified variables. Note that this can be done since we assumed that $\langle \Gamma \rangle = BR$ which implies that Γ can pp-define every Boolean relation. □

Using Lemma 6 and Lemma 7 we can now proceed to prove our lower bound by a reduction from VERTEX COVER.

Theorem 16. *Let Γ be a finite Boolean constraint language such that $\langle\Gamma\rangle = BR$ and $\phi_1 \notin \mathrm{pPol}(\Gamma)$. Then $\mathrm{SAT}(\Gamma)$ does not have a kernel of size $O(n^{2-\varepsilon})$ for any $\varepsilon > 0$, unless NP $\subseteq$ co-NP/poly.*

Proof. We will give a polynomial-time many-one reduction from VERTEX COVER parameterized by the number of vertices to $\mathrm{SAT}(\Gamma \cup \{\Phi_1\})$, which via Theorem 3 and Lemma 6 has a reduction to $\mathrm{SAT}(\Gamma)$ which does not increase the number of variables. Let (V, E) be the input graph and let k denote the maximum size of the cover. First, introduce two fresh variables x_v and x'_v for each $v \in V$, and one variable y_i for each $1 \leq i \leq k$. Furthermore, introduce two variables x and y. For each edge $\{u, v\} \in E$ introduce a constraint $\Phi_1(x_u, x'_v, x'_u, x_v, x, y)$, and note that this enforces the constraint $(x_u \vee x_v)$. Let

$$\exists z_1, \ldots, z_m : \phi(x_1, \ldots, x_{|V|}, y_1, \ldots, y_k, z_1, \ldots, z_m)$$

denote the pp-definition $H_{|V|+k,k}$ over Γ where $m \in O(k + |V|)$, and consisting of at most $O(k + |V|)$ constraints. Such a pp-definition must exist according to Lemma 7. Drop the existential quantifiers and add the constraints of $\phi(x_1, \ldots, x_{|V|}, y_1, \ldots, y_k, z_1, \ldots, z_m)$. Let (V', C) denote this instance of $\mathrm{SAT}(\Gamma \cup \{\Phi_1\})$. Assume first that (V, E) has a vertex cover of size $k' \leq k$. We first assign x the value 0 and y the value 1. For each v in this cover assign x_v the value 1 and x'_v the value 0. For any vertex not included in the cover we use the opposite values. We then set $y_1, \ldots, y_{k-k'}$ to 1, and $y_{k-k'+1}, \ldots, y_k$ to 0. For the other direction, assume that (V', C) is satisfiable. For any x_v variable assigned 1 we then let v be part of the vertex cover. Since

$$x_1 + \ldots + x_{|V|} + y_1 + \ldots + y_k = k,$$

the resulting vertex cover is smaller than or equal to k. □

Example 8. *Consider the relation $R^k = \{(b_1, \ldots, b_k) \in \{0,1\}^k \mid b_1 + \ldots + b_k \in \{1, 2\} \pmod 6\}$ and let $P = \{R^k \mid k \geq 1\}$. The kernelization status of $\mathrm{SAT}(P)$ was left open in Jansen and Pieterse [24], and while a precise upper bound seems difficult to obtain, we can at least prove that this problem does not admit a kernel of linear size, unless NP $\subseteq$ co-NP/poly. To see this, observe that $(0,0,1), (0,1,1), (0,1,0) \in R^3$ but $\phi_1((0,0,1), (0,1,1), (0,1,0)) = (0,0,0) \notin R^3$. The result then follows from Theorem 16.*

Although $\phi_1 \in \mathrm{pPol}(\Gamma)$ is a necessary condition for the existence of a linear kernel, it is crucial to note that this condition does not even guarantee the existence of a Maltsev extension. To see this, note first that Theorem 15 implies that there for every $d \geq 1$ exists $e > d$ such that $\phi_e \notin [\{\phi_1, \ldots, \phi_d\}]_s$, which by Corollary 1 disproves the existence of a Maltsev extension. This furthermore implies that there exists relations preserved by $\phi_1, \ldots, \phi_d$ but not by ϕ_e for sufficiently large $e > d$, and the proof of Theorem 16 would not go through easily for such relations, simply because Φ_1 is not preserved by ϕ_1, and thus no language Γ invariant under ϕ_1 can qfpp-implement Φ_1. We return to this question in Section 6.4 and give a concrete example of a relation R preserved by ϕ_1 but not by ϕ_2, for which we have been unable to determine the kernelization status.

Finally, regardless of whether Maltsev extensions are necessary for linear kernels, we can show that any characterisation of kernelizability in terms of a finite set of partial operations P is incomplete. As discussed in the introduction, this is in contrast to the existing parameterized dichotomy results for CSP for more permissive parameters k [9, 30, 31, 32, 36], but it is in line with previous observations on the complexity of $\mathrm{Inv}(P)$ for finite P; cf. [33, Lemma 35]. To prove this we will use a *padding procedure* that, starting from a relation R of arity r, defines a padded relation R' of arity $r' = r^{O(1)}$ such that R' is preserved by all partial operations of sufficiently small arity. The principle is to pad R by arguments that correspond to d-ary functions over R, such that, for any low-arity partial operation p and every sequence of tuples $t_1, \ldots, t_c \in R'$, the value of $p(t_1, \ldots, t_c)$ is undefined in one of the padding columns. This construction is formalized in the following lemma, where we for notational convenience sometimes will treat a tuple of variables as a set, and for example write $R(X) \equiv \exists Y : R'(X, Y)$ instead of $R(x_1, \ldots, x_{|X|}) \equiv \exists y_1, \ldots, y_{|Y|} : R'(x_1, \ldots, x_{|X|}, y_1, \ldots, y_{|Y|})$.

Lemma 8. *Fix a constant $c \in \mathbb{N}$. For every relation $R \subseteq \{0,1\}^r$, there is a relation R' of arity $r^{O(1)}$ created by adding padding columns to R (where the exponent depends on c), such that $R(X) \equiv \exists Y : R'(X, Y)$ and R' is invariant under every non-total partial operation of arity at most c.*

Proof. Let $X = \{x_1, \ldots, x_r\}$, $d = c^2$, and define a set of *padding variables* $Y = \{y_{\bar{x},f} \mid \bar{x} \in X^d, f : \{0,1\}^d \to \{0,1\}\}$, one for every $(x_1, \ldots, x_d) \in X^d$ and every d-ary operation f. Define a relation R' of arity $|X| + |Y|$ as

$$R'(X, Y) \equiv R(X) \wedge \bigwedge_{y_{\bar{x},f} \in Y} (y_{\bar{x},f} = f(\bar{x}[1], \ldots, \bar{x}[c])).$$

Then $R(X) \equiv \exists Y : R'(X, Y)$ by design, and $|Y| = r^d 2^{2^d} = r^{O(1)}$ since $d = c^2$ is a constant. We show that R' is preserved by all partial operations of arity lower than or equal to c.

Let ϕ be a non-total partial operation of arity $c' \leq c$, and assume that R' is not invariant under ϕ, i.e., there are tuples $t_1, \ldots, t_{c'} \in R'$ such that $\phi(t_1, \ldots, t_{c'})$ is defined and not contained in R'. We assume that all tuples t_i are distinct, as otherwise the application $\phi(t_1, \ldots, t_c)$ defines an operation ϕ' of arity $|\{t_1, \ldots, t_{c'}\}|$ for which we can repeat the argument below. Let $u_1, \ldots, u_{c'}$ be the projections of the tuples onto X, i.e., $u_i = \mathrm{pr}_{1,\ldots,|X|}(t_i)$. Note that the tuples $u_1, \ldots, u_{c'}$ are distinct, and that $\phi(u_1, \ldots, u_{c'})$ is defined. Let $I \subseteq [r]$ be a minimal set of "witness positions" for the distinctness of u, i.e., for every pair $i, j \in [c']$, $i \neq j$, there is a position $p \in I$ such that $u_i[p] \neq u_j[p]$. Note that $|I| \leq c^2$. Let $t \in \{0,1\}^{c'}$ be a tuple for which ϕ is undefined. Then there exists a function $f : \{0,1\}^{|I|} \to \{0,1\}$ such that $f(\mathrm{pr}_I(u_i)) = t[i]$ for each $i \in [c']$, since the projection onto I is distinct for all tuples u_i, and since $|I| \leq d^2$, there exist a variable $y_{\bar{x},f}$ in Y. This implies that $\phi(t_1, \ldots, t_c)$ is undefined, since in particular ϕ is undefined when applied to the position corresponding to $y_{\bar{x},f}$. Since ϕ was generically chosen, the claim follows. □

The following theorem now follows without too much difficulty.

Theorem 17. *Let P be a finite set of partial polymorphisms such that $\langle \mathrm{Inv}(P) \rangle = BR$. Then for every $c \geq 2$ there is a finite Boolean language Γ such that $\Gamma \subset \mathrm{Inv}(P)$ but $\mathrm{SAT}(\Gamma)$ does not admit a kernel of $O(n^{c-\varepsilon})$ bits for any $\varepsilon > 0$ unless the polynomial hierarchy collapses. In particular, $\mathrm{SAT}(\mathrm{Inv}(P))$ does not admit any polynomial kernel under the same assumption.*

Proof. We will show that for a constant q that only depends on P, and for every $k \geq 3$, there is a finite language Γ_k such that there is a polynomial-time reduction from k-SAT on n variables to $\mathrm{SAT}(\Gamma_k)$ on $O(n^q)$ variables. Since k-SAT admits no kernel of size $O(n^{k-\varepsilon})$ for any $\varepsilon > 0$ unless NP $\subseteq$ co-NP/poly [15], and since q is independent of k, the result will follow.

Let c be the largest arity of a partial polymorphism in P, and let (X, C) be an instance of k-SAT, $|X| = n$. Create a set of "global" padding variables $Y = \{y_{\bar{x},f} \mid \bar{x} \in X^d, f : \{0,1\}^d \to \{0,1\}\}$, one for every $(x_1, \ldots, x_d) \in X^d$ and every d-ary operation, $d = c^2$. We will constrain so that for every $y_{\bar{x},f} \in Y$ and every satisfying assignment, we have $y_{\bar{x},f} = f(\bar{x}[1], \ldots, \bar{x}[d])$.

To do this, let $R_0 = \{0,1\}^k$ and let R'_0 be its padded version as in Lemma 8. For every k-tuple $(x_1, \ldots, x_r)$ of variables from X, add a constraint $R'_0(x_1, \ldots, x_r, y_1, \ldots, y_t)$, where R'_0 is the padded version of R_0 and where $y_1, \ldots, y_t$ is the enumeration of padding variables $y_i \in Y$ corresponding to operations over $(x_1, \ldots, x_k)$. This locks in the value of y for every $y \in Y$, without constraining the possible values of X. Additionally, create a padded relation for every possible k-clause, and for every k-clause in the input, defining a relation $R(V)$ for some $V \subseteq X$, output the constraint $R'(V, Y_V)$ where R' is the padded version of R and the variables $Y_V \subseteq Y$ are chosen accordingly. Note that the padding variables of Y can be "reused" between different constraint applications, since they are simply defined by operations over X.

Let Γ be the language containing the relations R'_0 and R' for every k-clause R; note that it suffices that $|\Gamma| = k + 2$. We have thus defined a reduction from k-SAT on n variables to $\mathrm{SAT}(\Gamma)$ on $O(|Y|) = n^{O(1)}$ variables, where the constant in the exponent only depends on c (and in particular not on k). The theorem follows. □

6 Concluding Remarks and Open Questions

We have studied the kernelization properties of SAT and CSP problems parameterized by the number of variables with tools from universal algebra. We particularly focused on problems with linear kernels, and showed that a CSP problem has a kernel with $O(n)$ constraints if it can be extended into a CSP problem on a larger domain preserved by a Maltsev operation. In a similar vein, if the CSP is extended into a CSP preserved by a k-edge operation, $k > 2$, then the problem has a kernel with $O(n^{k-1})$ constraints.

For Boolean languages, we additionally considered a generalisation of the method of encoding relations as the roots of bounded-degree polynomials, as previously employed by Jansen and Pieterse [24]. For this, we study the *degree-c extension* of the language for some $c > 1$, roughly corresponding to relations defined over c-tuples of variables from the original language. We show that if the degree-c extension of a Boolean language Γ has a Maltsev extension, then $\mathrm{SAT}(\Gamma)$ has a kernel with $O(n^c)$ constraints, and if the extension has a k-edge extension then $\mathrm{SAT}(\Gamma)$ has a kernel with $O(n^{(k-1)c})$ constraints. In particular, if R is definable as the roots of a polynomial of degree at most c over some finite field, then the degree-c extension of R has a natural Maltsev extension. Therefore our approach directly generalizes previous results on polynomial sparsification of $\mathrm{SAT}(\Gamma)$ problems [24].

Finally, we also considered lower bounds and algebraic characterisations of the languages for which our approach applies. We give a complete algebraic characterisation of languages with a Maltsev extension, up to a distinction between finite-domain and infinite-domain extensions, and for Boolean languages we give partial corresponding lower bounds against linear kernels, assuming the polynomial hierarchy does not collapse.

6.1 Linear Kernels for Boolean Languages

Let us now in some more detail discuss the state of our results for linear kernels. In the following, let Γ be a finite Boolean language such that $\mathrm{SAT}(\Gamma)$ is NP-hard. Then we have shown the following.

- If every relation $R \in \Gamma$ has a Maltsev extension $\hat{R}$ into a finite domain D_R, then $\mathrm{SAT}(\Gamma)$ has a basis of $O(n)$ constraints, i.e., any set C of constraints using relations of Γ and on n variables contains a basis $C' \subseteq C$ such that $|C'| = O(n)$ and every assignment that satisfies C' also satisfies C. This basis can be computed in polynomial time, and consequently $\mathrm{SAT}(\Gamma)$ has a linear kernel.
- There is a set P of *universal partial Maltsev operations*, such that Γ admits a Maltsev extension if and only if Γ is preserved by every partial operation in P. However, the resulting extension is over an infinite domain. The set P has no finite basis, but it has an easily described countable basis $\{\phi_1, \phi_2, \ldots\}$ where ϕ_i implies ϕ_j for every $j < i$.
- If Γ is not preserved by the first operation ϕ_1 in this enumeration, then $\mathrm{SAT}(\Gamma)$ admits no kernel of $O(n^{2-\varepsilon})$ bits for any $\varepsilon > 0$ unless the polynomial hierarchy collapses. Consequently, the language also has no linear basis.
- Finally, for every finite set P' of strictly partial Boolean operations and every constant $c > 1$, there is a finite language Γ such that Γ is preserved by P' and $\mathrm{SAT}(\Gamma)$ admits no kernel with $O(n^{c-\varepsilon})$ bits for any $\varepsilon > 0$ unless the polynomial hierarchy collapses.

In particular, our work leaves the following questions open.

1. Is there a finite language (Boolean or otherwise) which admits an infinite-domain Maltsev extension but no Maltsev extension into a finite domain?
2. Does the existence of an infinite-domain Maltsev extension for a language Γ itself imply that $\mathrm{CSP}(\Gamma)$ has a linear basis?
3. Conversely, assume that Γ does not admit a Maltsev extension, i.e., $\phi_i \notin \mathrm{pPol}(\Gamma)$ for some $i > 1$ (since the case $i = 1$ is known). Does this imply that $\mathrm{SAT}(\Gamma)$ does not have a linear basis, or even no linear kernel?

6.2 Polynomial, Super-Linear Kernels

Our investigation into the existence of kernels of polynomial but superlinear size is less complete than that for linear kernels, but we wish to highlight a few questions.

1. Are k-edge extensions a necessary ingredient for Boolean $\mathrm{SAT}(\Gamma)$ problems? That is, assume that Γ has a degree-c extension with a k-edge extension, for some constants $c, k > 1$. Does the degree-c' extension of Γ for $c' = (k-1)c$ admit a Maltsev extension?

2. What is the proper generalisation of bounded-degree extensions to CSP problems on a higher domain? Is it always asymptotically optimal to use padding variables corresponding to all c-ary functions over the main variables, or do some problems only admit a kernel of degree c by using a very specific set of c-ary padding functions?

See also the discussion about padding following problem 2 in Section 6.4.

6.3 The Meta-Problem in Kernelization

Theorem 16 can be seen as a first step in answering a meta problem in kernelization, i.e., given a constraint language Γ, decide whether CSP(Γ) admits a kernel of, for example, linear size. A reasonable starting point is to first try to construct an algorithm for checking whether Γ admits a Maltsev extension, which we by Theorem 14 already know is equivalent to checking whether the partial operations $\phi_1, \phi_2, \ldots$ preserve Γ. Due to Theorem 15, such an algorithm is likely not possible for arbitrary infinite languages, but it is known that the partial polymorphisms of any finite constraint language can be characterized by a finite set of partial operations [35]. Thus, is it for every finite Γ possible to find d such that if $\phi_1, \ldots, \phi_d$ preserve Γ, then Γ is preserved by ϕ_e for any $e \geq d$, and therefore admits a Maltsev extension?

6.4 Concrete Open Problems

Finally, let us provide a few concrete languages for which the correct kernel size is unknown.

Problem 1. The following is the smallest example we have found of a problem with unknown kernelization status, found via a computer search. As shorthand, let us omit spaces and commas when describing tuples, i.e., the tuple $(0, 0, 1, 1)$ would be written as 0011. Then the problem is a relation $R \subset \{0, 1\}^{10}$ defined as

$$R = \{0000000001, 1000100010, 0100011000, 0011000100, 1000010100, 0010101000\}.$$

Exhaustive testing will verify that R is preserved by ϕ_1. But define the 6-ary operation

$$q = u(u(x_1, x_2, x_3), u(x_4, x_2, x_5), x_6),$$

where u is the Maltsev operation on D_∞ from Definition 10. Then $q_{|\mathbb{B}}$ is a universal partial Maltsev operation, but

$$q_{|\mathbb{B}}(1000100010, 0000000001, 0100011000, 1000010100, 0010101000, 0011000100) = 0101000010,$$

where $0101000010 \notin R$. Observe also that this implies that ϕ_2 does not preserve R, since $q_{|\mathbb{B}} \in [\{\phi_2\}]_s$ by the proof of Theorem 14. On the other hand, define

$$R' = R \cup \{0101000010\}.$$

Then R' can be implemented using 1-in-4 constraints. Indeed, name the arguments of R and R' as $R(x_{1,1}, x_{1,2}, x_{1,3}, x_{2,1}, \ldots, x_{3,3}, z)$. Then we see that either $z = 1$ or there is exactly one non-zero entry in every "row" $x_{i,*}$ and in every "column" $x_{*,i}$, $i = 1, 2, 3$, and that this describes R' completely. Thus SAT($\{R'\}$) admits a Maltsev extension on domain $\{0, 1, 2, 3\}$ and a linear kernel, whereas SAT($\{R\}$) does not admit a Maltsev extension but the existence of a linear kernel is an open question.

Problem 2. The next question concerns the precise degree of polynomial but super-linear kernels for padded languages. Let Γ_3 be the language of 3-clauses, and let $\Gamma_3^{(2)}$ be the result of padding every relation $R \in \Gamma_3$ with all binary conjunctions as in the degree-2 extension (so the relations in $\Gamma_3^{(2)}$ have arity 6). Then SAT($\Gamma_3^{(2)}$) has a kernel with $O(n^2)$ constraints, since the cube term xyz over a constraint $R(x, y, z)$ can be written as a binary term $((xy)z$ over the constraint $R(x, y, z, xy, xz, yz)$; and as a lower bound we note that SAT($\Gamma_3^{(2)}$) cannot have a kernel of size $O(n^{3/2-\varepsilon})$ for any $\varepsilon > 0$ unless the polynomial hierarchy collapses, since such a kernel would contradict known lower bounds for 3-SAT [15]. Can we close the gap between these upper and lower bounds?

The same question can also be generalized to k-SAT for any $k > 3$, i.e., if $\Gamma_k^{(k-1)}$ is attained by padding k-clauses using $(k-1)$-ary functions, then we have an upper bound of a kernel of $O(n^2)$ constraints and a lower bound against kernels of size $O(n^{\frac{k}{k-1}-\varepsilon})$ for $\varepsilon > 0$. Furthermore, the languages Γ_k^{k-1} will for sufficiently large $k = k(r)$ be closed under all partial operations of arity up to r; cf. the proof of Lemma 8. The same general pattern will hold for any other language $\mathrm{SAT}(\Gamma)$ with a non-trivial lower bound on the kernel size: if $\mathrm{SAT}(\Gamma)$ admits no kernel better than size $O(n^r)$, then its padded version $\mathrm{SAT}(\Gamma^{(r-1)})$ will not admit a linear kernel. However, this implication does not necessarily run two ways; it is not known to us whether a lower bound against a linear kernel for $\mathrm{SAT}(\Gamma^{(c)})$ for some $c > 1$ implies a lower bound against a kernel of $O(n^c)$ bits for $\mathrm{SAT}(\Gamma)$.

References

[1] L. Barto. Constraint satisfaction problem and universal algebra. *ACM SIGLOG News*, 1(2):14–24, October 2014.

[2] J. Berman, P. Idziak, P. Marković, R. McKenzie, M. Valeriote, and R. Willard. Varieties with few subalgebras of powers. *Transactions of the American Mathematical Society*, 362(3):1445–1473, 2010.

[3] V. G. Bodnarchuk, L. A. Kaluzhnin, V. N. Kotov, and B. A. Romov. Galois theory for Post algebras. I. *Cybernetics and Systems Analysis*, 5:243–252, 1969.

[4] V. G. Bodnarchuk, L. A. Kaluzhnin, V. N. Kotov, and B. A. Romov. Galois theory for Post algebras. II. *Cybernetics and Systems Analysis*, 5:531–539, 1969.

[5] E. Böhler, H. Schnoor, S. Reith, and H. Vollmer. Bases for Boolean co-clones. *Information Processing Letters*, 96(2):59–66, 2005.

[6] A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS-2017)*, pages 319–330, Washington, DC, USA, 2017. IEEE Computer Society.

[7] A. Bulatov and V. Dalmau. A simple algorithm for Mal'tsev constraints. *SIAM Journal on Computing*, 36(1):16–27, 2006.

[8] A. Bulatov, P. Jeavons, and A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34(3):720–742, March 2005.

[9] A. Bulatov and D. Marx. Constraint satisfaction parameterized by solution size. *SIAM Journal on Computing*, 43(2):573–616, 2014.

[10] S. Burris and H.P. Sankappanavar. *A course in universal algebra*. Graduate texts in mathematics. Springer-Verlag, Berlin, Heidelberg, 1981.

[11] H. Chen, B. M. P. Jansen, and A. Pieterse. Best-case and worst-case sparsifiability of boolean csps. In *Proceedings of the 13th International Symposium on Parameterized and Exact Computation (IPEC-2018)*, volume 115 of *LIPIcs*, pages 15:1–15:13, Oktavie-Allee, 66687 Wadern, Germany, 2018. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

[12] N. Creignou and H. Vollmer. Boolean constraint satisfaction problems: When does Post's lattice help? In N. Creignou, P. G. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, volume 5250 of *Lecture Notes in Computer Science*, pages 3–37. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[13] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer, New York, NY, USA, 2015.

[14] V. Dalmau and P. Jeavons. Learnability of quantified formulas. *Theoretical Computer Science*, 306(1-3):485 – 511, 2003.

[15] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *Journal of the ACM*, 61(4):23:1–23:27, 2014.

[16] M. Dyer and D. Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM Journal on Computing*, 42(3):1245–1274, 2013.

[17] T. Feder and M. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.

[18] F. V. Fomin, D. Lokshtanov, S. Saurabh, and M. Zehavi. *Kernelization: Theory of Parameterized Preprocessing.* Cambridge University Press, New York, NY, USA, 2019.

[19] D. Geiger. Closed systems of functions and predicates. *Pacific Journal of Mathematics*, 27(1):95–100, 1968.

[20] M. Goldstern and M. Pinsker. A survey of clones on infinite sets. *Algebra universalis*, 59(3):365–403, 2008.

[21] W. Hodges. *A Shorter Model Theory.* Cambridge University Press, New York, NY, USA, 1997.

[22] P. Idziak, P. Marković, R. McKenzie, M. Valeriote, and R. Willard. Tractability and learnability arising from algebras with few subpowers. *SIAM Journal on Computing*, 39(7):3023–3037, June 2010.

[23] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63:512–530, 2001.

[24] B. M. P. Jansen and A. Pieterse. Optimal sparsification for some binary CSPs using low-degree polynomials. In *Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS-2016)*, volume 58, pages 71:1–71:14, Oktavie-Allee, 66687 Wadern, Germany, 2016. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

[25] B. M. P. Jansen and A. Pieterse. Optimal data reduction for graph coloring using low-degree polynomials. In *IPEC*, volume 89 of *LIPIcs*, pages 22:1–22:12, Oktavie-Allee, 66687 Wadern, Germany, 2017. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

[26] B. M. P. Jansen and A. Pieterse. Sparsification upper and lower bounds for graph problems and not-all-equal SAT. *Algorithmica*, 79(1):3–28, 2017.

[27] P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

[28] P. Jeavons, D. Cohen, and M. Gyssens. A unifying framework for tractable constraints. In *Proceedings of the First International Conference in Principles and Practice of Constraint Programming (CP-1995)*, pages 276–291, Berlin, Heidelberg, 1995. Springer-Verlag.

[29] P. Jonsson, V. Lagerkvist, G. Nordh, and B. Zanuttini. Strong partial clones and the time complexity of SAT problems. *Journal of Computer and System Sciences*, 84:52 – 78, 2017.

[30] S. Kratsch, D. Marx, and M. Wahlström. Parameterized complexity and kernelizability of max ones and exact ones problems. *ACM Transactions on Computation Theory*, 8(1):1, 2016.

[31] S. Kratsch and M. Wahlström. Preprocessing of min ones problems: A dichotomy. In *Proceedings of the 7th International Colloquium on Automata, Languages and Programming (ICALP-2010)*, volume 6198 of *Lecture Notes in Computer Science*, pages 653–665, Berlin, Heidelberg, 2010. Springer-Verlag.

[32] A. A. Krokhin and D. Marx. On the hardness of losing weight. *ACM Transactions on Algorithms*, 8(2):19, 2012.

[33] V. Lagerkvist and M. Wahlström. The power of primitive positive definitions with polynomially many variables. *Journal of Logic and Computation*, 27(5):1465–1488, 2017.

[34] V. Lagerkvist and M. Wahlström. Which np-hard SAT and CSP problems admit exponentially improved algorithms? *CoRR*, abs/1801.09488, 2018.

[35] V. Lagerkvist, M. Wahlström, and B. Zanuttini. Bounded bases of strong partial clones. In *Proceedings of the 45th International Symposium on Multiple-Valued Logic (ISMVL-2015)*, pages 189–194, Washington, DC, USA, 2015. IEEE Computer Society.

[36] D. Marx. Parameterized complexity of constraint satisfaction problems. *Computational Complexity*, 14(2):153–183, 2005.

[37] G. L. Nemhauser and L. E. Trotter. Vertex packings: Structural properties and algorithms. *Mathematical Programming*, 8(1):232–248, 1975.

[38] B.A. Romov. The algebras of partial functions and their invariants. *Cybernetics and Systems Analysis*, 17(2):157–167, 1981.

[39] T. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory Of Computing (STOC-78)*, pages 216–226, New York, NY, USA, 1978. ACM.

[40] G. S. Tseitin. *Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970*, chapter On the Complexity of Derivation in Propositional Calculus, pages 466–483. Springer Berlin Heidelberg, Berlin, Heidelberg, 1983.

[41] D. Zhuk. The proof of CSP dichotomy conjecture. In *Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS-2017)*, pages 331–342, Washington, DC, USA, 2017. IEEE Computer Society.