The Exponential-Time Hypothesis and the Relative Complexity of Optimization and Logical Reasoning Problems

Peter Jonsson[a], Victor Lagerkvist[a,*], Johannes Schmidt[b], Hannes Uppman[c]

[a]*Department of Computer and Information Science, Linköping University, Sweden*
{peter.jonsson, victor.lagerkvist}@liu.se
[b]*Department of Computer Science and Informatics, Jönköping University, Sweden*
johannes.schmidt@ju.se
[c]*Linköping, Sweden*
hannes.uppman@gmail.com

Abstract

Obtaining lower bounds for NP-hard problems has for a long time been an active area of research. Algebraic techniques introduced by Jonsson et al. (JCSS 2017) show that the fine-grained time complexity of the parameterized $\mathrm{SAT}(\cdot)$ problem correlates to the lattice of strong partial clones. With this ordering they isolated a relation R such that $\mathrm{SAT}(R)$ can be solved at least as fast as any other NP-hard $\mathrm{SAT}(\cdot)$ problem. In this paper we extend this method and show that such languages also exist for the *surjective SAT problem*, the *max ones problem*, the propositional *abduction problem*, and the *Boolean valued constraint satisfaction problem* over finite-valued constraint languages. These languages may be interesting when investigating the borderline between polynomial time, subexponential time and exponential-time algorithms since they in a precise sense can be regarded as NP-hard problems with minimum time complexity. Indeed, with the help of these languages we relate all of the above problems to the *exponential time hypothesis* (ETH) in several different ways.

Keywords: Constraint satisfaction problems, fine-grained complexity, universal algebra

1. Introduction

In this article we study the *fine-grained complexity* of NP-hard optimization problems and logical reasoning problems, with a particular focus on describing the *relative* complexity of the problems in each class. For each problem class under consideration we are interested in determining an intractable problem which is 'maximally easy', in the sense that there cannot exist any other intractable problem in the class with a strictly lower (exponential) running time. After successfully accomplishing this for the four problems under consideration we then explore the likelihood of obtaining

*Corresponding author

subexponential time algorithms, in light of the *exponential-time hypothesis*, where we obtain several strong equivalent characterizations.

Background

A superficial analysis of the NP-complete problems may lead one to think that they are a highly uniform class of problems: in fact, under polynomial-time reductions, the NP-complete problems may be viewed as a *single* problem. However, there are many indications (both from practical and theoretical viewpoints) that the NP-complete problems are a diverse set of problems with highly varying properties, and this becomes visible as soon as one analyses these problems with more refined methods than polynomial-time reductions. This has inspired a strong line of research on the 'inner structure' of the set of NP-complete problem, sometimes referred to as *fine-grained* complexity, to contrast it against classical *coarse-grained* complexity. Examples include the intensive search for faster algorithms for NP-complete problems [1] and the highly influential work on the *exponential-time hypothesis* (ETH) and its variants [2]. Such research might not directly resolve whether P is equal to NP or not, but rather attempts to explain the seemingly large difference in complexity between NP-hard problems, and what makes one problem seemingly harder than another. Tangentially related research include investigations into more restricted nondeterministic complexity classes than NP, e.g., the complexity class VERTEXNLIN, which is also defined with respect to a fine-grained complexity parameter [3].

Unfortunately, there is a lack of general methods for studying and comparing the complexity of NP-complete problems with more restricted notions of reducibility. Jonsson et al. [4] presented a framework based on *clone theory*, applicable to problems that can be viewed as 'assigning values to variables', such as Boolean *satisfiability* (SAT) problems, *constraint satisfaction problems* (CSPs), the vertex cover problem, and integer programming problems. In short, every instance of a 'variable assignment problem' corresponds to a set of potential models, and with the algebraic approach it is possible to describe the symmetry of this solution space by properties of *partial polymorphisms*, in such a way that computationally hard problems have a small amount of symmetry, while comparably easier problems have a richer amount of symmetry. The corresponding algebraic method for studying classical complexity based on *total polymorphisms*, the so-called *algebraic approach*, turned out to be immensely successful and recently resulted in a complete characterization of the classical complexity of CSPs: the *CSP dichotomy theorem* [5, 6]. For a direct comparison between the classical and fine-grained approach, see e.g. the survey by Couceiro et al. [7]. Hence, while the algebraic framework for studying fine-grained complexity is not as mature as the algebraic approach for studying classical complexity, there is potentially much to gain from expanding this toolbox, and investigating its limits by exploring new classes of problems.

Aims and methods

Inspired by this algebraic framework of Jonsson et al. [4] we study fine-grained complexity aspects for a wide range of 'variable assignment problems' of particular importance in Boolean optimization and propositional logical reasoning. To analyze

and relate the complexity of these problems in greater detail we utilize polynomial-time reductions which increase the number of variables by a constant factor (*linear variable reductions* or *LV-reductions*) and reductions which increase the amount of variables by a constant (*constant variable reductions* or *CV-reductions*). Note the following: (1) if a problem A is solvable in $O(c^n \cdot \text{poly}(m))$ time (where n denotes the number of variables and m the size of the instance) for all $c > 1$ and if problem B is LV-reducible to A then B is also solvable in $O(c^n \cdot \text{poly}(m))$ time for all $c > 1$ and (2) if A is solvable in time $O(c^n \cdot \text{poly}(m))$ and if B is CV-reducible to A then B is also solvable in time $O(c^n \cdot \text{poly}(m))$. Thus, LV-reductions preserve *subexponential* complexity while CV-reductions preserve the exact constant in an exponential running time of the form $O(c^n \cdot \text{poly}(m))$.

While there exist NP-hard problems solvable in subexponential time, e.g., the FEEDBACK ARC SET problem restricted to tournaments [8], it is widely believed that this is not the case for all NP-hard problems. The particular conjecture that the 3-SAT problem is not solvable in subexponential time is known as the *exponential-time hypothesis* (ETH) [9]. Thus, the ETH can be seen as a stronger, more fine-grained variant of P $\neq$ NP, which has proven to be an immensely useful tool for proving superpolynomial lower bounds for many different types of problems [10]. In this vein, and with the aforementioned algebraic approach, Jonsson et al. [4] studied the Boolean satisfiability $\text{SAT}(\cdot)$ problem and identified an NP-hard $\text{SAT}(\{R\})$ problem CV-reducible to all other NP-hard $\text{SAT}(\cdot)$ problems. Hence $\text{SAT}(\{R\})$ is, in a sense, the *easiest* NP-complete $\text{SAT}(\cdot)$ problem since if any other NP-hard $\text{SAT}(\Gamma)$ can be solved in $O(c^n)$ time, then this holds for $\text{SAT}(\{R\})$, too. The existence of an 'easiest problem' of this form is not only an interesting theoretical curiosity, but has important consequences. For example: if there exists *any* NP-hard $\text{SAT}(\Gamma)$ problem solvable in subexponential time, then the easiest problem $\text{SAT}(\{R\})$ must be solvable in subexponential time, too. Hence, if we can prove that 3-SAT is no harder than $\text{SAT}(\{R\})$, via a suitable LV-reduction, then *no* NP-hard $\text{SAT}(\Gamma)$ problem is solvable in subexponential time without violating the ETH. The advantage is then that it is significantly easier to show the existence of an LV-reduction of this form for a concrete language $\{R\}$, than to consider arbitrary intractable problems $\text{SAT}(\Gamma)$. With the aid of this result, Jonsson et al. [4] also analyzed the applicability of the *sparsification lemma* [11] for arbitrary $\text{SAT}(\cdot)$ problem, and proved that sparsification of $\text{SAT}(\Gamma)$ is always possible when Γ is finite. This should not be taken for granted since Santhanam and Srinivasan [12] have proven that sparsification is not always possible for infinite constraint languages. This study was later generalised to a broad class of finite-domain constraint satisfaction problems where it was proven that one can find an 'easiest NP-hard CSP' for every fixed, finite, domain [13]. Curiously, this sequence of problems was shown to decrease in complexity, in the sense that one for every $c > 1$ can find an NP-hard finite-domain CSP problem solvable in $O(c^n)$ time, even though none of these problems are solvable in subexponential time without contradicting the ETH. However, if one steps into the realm of *infinite-domain CSPs*, then it is known that there cannot exist an 'easiest NP-hard infinite-domain CSP', unless the ETH is false [14]. Thus, while easiest problems of this form exist for finite-domain CSPs and SAT problems, they should in general not be taken for granted. We believe that the existence and construction of such easiest languages forms an important puzzle piece in the quest of relating the complexity of NP-hard problems with each other, since

it effectively gives a lower bound on the time complexity of a given problem with respect to constraint language restrictions.

Our results

As a logical continuation on the work on $\mathrm{SAT}(\cdot)$ and $\mathrm{CSP}(\cdot)$ we pursue the study of CV- and LV-reducibility in the context of Boolean optimization problems and logical reasoning problems. In particular, we investigate the complexity of the *maximum ones problem* (MAX-ONES$(\cdot)$), the *surjective satisfiability problem* ($\mathrm{SSAT}(\cdot)$), the *propositional abduction problem* ($\mathrm{ABD}(\cdot)$) and the *Boolean valued constraint satisfaction problem* ($\mathrm{VCSP}(\cdot)$). These problems are presented in greater detail in Section 2, but for the moment they can be summarised as follows.

- The MAX-ONES$(\cdot)$ problem [15] is a variant of $\mathrm{SAT}(\cdot)$ where the goal is to find a satisfying assignment which maximizes the number of variables assigned the value 1. This problem is closely related to the 0/1 LINEAR PROGRAMMING problem.
- The $\mathrm{SSAT}(\Gamma)$ problem is another twist on the classical $\mathrm{SAT}(\Gamma)$ problem where the goal is to find a surjective solution, which in the Boolean domain simply means that the presented solution is not constantly 0 or constantly 1 [16]. The $\mathrm{SSAT}(\cdot)$ problem is not only of theoretical interest and its complexity classification has been used to simplify complexity classifications of other problems of practical interest, e.g., *enumeration problems* [17].
- The $\mathrm{ABD}(\Gamma)$ problem is a well-known problem within artificial intelligence where the goal is to find an explanation of a manifestation which is consistent with a given Γ-formula. A complexity *trichotomy* is known for every finite, Boolean language Γ [18, 19], and we concentrate on the case when $\mathrm{ABD}(\Gamma)$ is included in NP.
- The $\mathrm{VCSP}(\cdot)$ problem is a function minimization problem that generalizes the MAX-CSP and MIN-CSP problems [15].

We treat both the unweighted and weighted versions of MAX-ONES$(\cdot)$ and $\mathrm{VCSP}(\cdot)$ and use the prefix U to denote the unweighted problem and W to denote the weighted version. All of these problems are well-studied with respect to separating tractable cases from NP-hard cases [15, 20] but much less is known when considering the weaker schemes of LV-reductions and CV-reductions. We begin (in Section 3) by identifying the easiest languages for $\mathrm{SSAT}(\cdot)$, W-MAX-ONES$(\cdot)$, and $\mathrm{ABD}(\cdot)$. The idea behind these proofs is to first perform a 'coarse-grained' complexity analysis, based on existing classical complexity classifications, and for each such case determine the easiest problem by identifying a constraint language with the richest set of partial polymorphisms. However, this may still result in a large number of cases that needs to be compared, and to identify an easiest problem one needs to prove that one such problem is CV-reducible to every other problem, which in general is highly non-trivial. To accomplish this for the W-MAX-ONES$(\cdot)$ problem we investigate a novel reduction technique based on *weighted primitive positive implementations* [21].

For VCSP$(\cdot)$ the situation differs even more since the algebraic techniques developed for CSP$(\cdot)$ are not applicable — instead we use *multimorphisms* [22] when considering the complexity of VCSP$(\cdot)$ in Section 3.4. We prove that the binary function $f_{\neq}$, which returns 0 if its two arguments are different and 1 otherwise, results in the easiest NP-hard VCSP$(\cdot)$ problem. This problem is very familiar since it is the MAX CUT problem slightly disguised. The complexity landscape surrounding these problems is outlined in Section 3.5. Interestingly, it turns out that none of the identified problems, with the possible exception of the easiest ABD$(\cdot)$ problem, is easier than the easiest SAT$(\cdot)$ problem from Jonsson et al. [4].

With the aid of the languages identified in Section 3, we continue (in Section 4) by relating the problems to subexponential complexity, using the ETH as a guiding star. This is accomplished via the aforementioned LV-reductions since they preserve subexponential complexity. For each such problem there are then two main questions with respect to subexponential complexity: is it possible to find an LV-reduction *from* 3-SAT, and is every such problem LV-reducible *to* 3-SAT? Since LV-reductions preserve subexponential complexity this means that we (1) rule out the possibility of subexponential algorithms if the ETH is *true*, and (2) show that every such problem is solvable in subexponential time if the ETH is *false*. If both of these conditions are proven to hold for a problem one therefore obtains a complete understanding of subexponential complexity (under the ETH). In particular, our results imply (1) if the ETH is true then no NP-complete U-MAX-ONES(Γ), W-MAX-ONES(Γ), SSAT(Γ), ABD(Γ), or VCSP(Δ) problem is solvable in subexponential time and (2) if the ETH is false then U-MAX-ONES(Γ), SSAT(Γ), and U-VCSP$_d(\Delta)$ are solvable in subexponential time for every choice of Γ and Δ and $d \geq 0$. Here, U-VCSP$_d(\Delta)$ is the U-VCSP(Δ) problem restricted to instances where the sum to minimize contains at most dn terms. Thus, to disprove the ETH, our result implies that it is sufficient to find a single language Γ or a set of cost functions Δ such that U-MAX-ONES(Γ), W-MAX-ONES(Γ), SSAT(Γ), ABD(Γ), or VCSP(Δ) is NP-hard and solvable in subexponential time.

2. Preliminaries

Let Γ denote a finite set of finitary relations over $\mathbb{B} = \{0,1\}$. We call Γ a *Boolean constraint language*, and when there is no risk for confusion, simply a *constraint language*. Given $R \subseteq \mathbb{B}^k$ we let $\mathrm{ar}(R) = k$ denote its arity, and similarly for functions. When $\Gamma = \{R\}$ we typically omit the set notation and treat R as a constraint language. We write $\mathbb{Q}_{\geq 0}$ for the set of all rational numbers larger than or equal to 0.

2.1. Problem Definitions

Let us now properly define the problems under consideration in this article. We begin with the *constraint satisfaction problem* over a set of relations Γ over a domain D (CSP(Γ)), which is defined as follows.

CSP(Γ)
INSTANCE: A set V of variables and a set C of constraint applications $R(v_1,\ldots,v_k)$ where $R \in \Gamma$, $k = \mathrm{ar}(R)$, and $v_1,\ldots,v_k \in V$.
QUESTION: Is there a function $f : V \to D$ such that $(f(v_1),\ldots,f(v_k)) \in R$ for each $R(v_1,\ldots,v_k)$ in C?

For the Boolean domain this problem is typically denoted as SAT(Γ), and in addition we write k-SAT to denote the variant of the satisfiability problem where each input clause is of length k. One may also remark that k-SAT can be formulated as a SAT(Γ) problem by letting the constraint language Γ consist of relations corresponding to the set of models of k-ary clauses. By SAT(Γ)-B we mean the SAT(Γ) problem restricted to instances where each variable can occur in at most B constraints. Similarly, we write k-SAT-B for the variant of k-SAT where each variable can occur in at most B constraints. These restricted problems are occasionally useful since each instance contains at most Bn constraints. We now define the variants of SAT(Γ) that are considered in this article. The *weighted maximum ones problem* over Γ (W-MAX-ONES(Γ)) is defined as follows.

W-MAX-ONES(Γ)
INSTANCE: A SAT(Γ) instance $(\{x_1,\ldots,x_n\},C)$ where each variable x_i has an associated weight $w_i \in \mathbb{Q}_{\geq 0}$
OBJECTIVE: Find a satisfying assignment h to $(\{x_1,\ldots,x_n\},C)$ which maximises the sum $\sum_{i=1}^{n} w_i h(x_i)$.

Example 1. *Naturally, every NP-hard* SAT(Γ) *problem results in a natural optimisation variant* W-MAX-ONES(Γ) *where one wishes to find a solution maximising the number of variables assigned 1. However,* W-MAX-ONES($\cdot$) *also includes many problems without a clear link to the standard satisfiability problem. For example, consider the relation* $\mathrm{NAND}^2 = \{(0,0),(0,1),(1,0)\}$*, and take an instance of* W-MAX-ONES(NAND^2)*, interpreted as the* complement *of a graph (i.e., a constraint* $\mathrm{NAND}^2(x,y)$ *means that there is no edge between x and y). The resulting problem is then nothing else than a reformulation of the (weighted variant of) the* MAX INDEPENDENT SET *problem, which is well known to be NP-hard. In contrast,* SAT(NAND^2) *is a special case of 2-SAT, and is thus solvable in polynomial time.*

The *unweighted maximum ones problem* (U-MAX-ONES(Γ)) is the W-MAX-ONES(Γ) problem where all weights have the value 1. Occasionally, it does not matter whether the problem is weighted or unweighted, and in that case we simply write MAX-ONES(Γ).

SSAT(Γ)
INSTANCE: A SAT(Γ) instance I.
QUESTION: Does there exist a satisfying assignment to I which is surjective?

Note that, since we operate in the Boolean domain, an assignment is surjective if and only if it is non-constant, i.e., does not assign the same value to each variable.

A *finite-valued cost function* on $\mathbb{B}$ is a function $f : \mathbb{B}^k \to \mathbb{Q}_{\geq 0}$. The *valued constraint satisfaction problem* over a finite set of finite-valued cost functions Δ (VCSP(Δ)) is defined as follows.

VCSP(Δ)
INSTANCE: A set $V = \{x_1, \ldots, x_n\}$ of variables and an objective function

$$f_I(x_1, \ldots, x_n) = \sum_{i=1}^{m} w_i f_i(\mathbf{x}_i),$$

where f_i is a cost function in Δ, $\mathbf{x}_i$ is a tuple over the variables V, and w_i is a non-negative rational weight.
OBJECTIVE: Find an assignment $h : V \to \mathbb{B}$ such that $f_I(h(x_1), \ldots, h(x_n))$ is minimised.

When the set of cost functions is singleton $\{f\}$ we write VCSP(f) for the VCSP($\cdot$) problem over $\{f\}$. We let U-VCSP be the VCSP problem without weights and U-VCSP$_d$ (for $d \geq 0$) denote the U-VCSP problem restricted to instances containing at most $d\,|\mathrm{Var}(I)|$ constraints.

Example 2. *Many optimization problems can be viewed as* VCSP(Δ) *problems for suitable* Δ*; well-known examples are the* MAX-CSP(Γ) *and* MIN-CSP(Γ) *problems where the number of satisfied constraints in a* CSP *instance are maximized or minimized. For each* Γ*, there obviously exists sets of cost functions* $\Delta_{\min}, \Delta_{\max}$ *such that* MIN-CSP(Γ) *is polynomial-time equivalent to* VCSP($\Delta_{\min}$) *and* MAX-CSP(Γ) *is polynomial-time equivalent to* VCSP($\Delta_{\max}$).

For a second, concrete example, recall that $f_{\neq}$ *is the Boolean cost function which returns 0 if and only if its two arguments are unequal, and consider the problem* VCSP($f_{\neq}$). *Then note that a* VCSP($f_{\neq}$) *instance can be viewed as a graph, and under this interpretation the task is thus to minimise the (sum of the weight of the) edges outside the cut, i.e., maximising the edges crossing the cut. Hence, this problem is a reformulation of the well-known* MAX-CUT *problem.*

We have defined U-VCSP, VCSP, U-MAX-ONES and W-MAX-ONES as optimization problems, but to obtain a more uniform treatment we often view them as decision problems, i.e. given k we ask if there is a solution with objective value k or better. We note that the representation of k has a size bounded by those of the representations of the weights. Thus, the introduction of the parameter k does not fundamentally change the problems.

We close this section by introducing the *propositional abduction problem* over a constraint language Γ (ABD(Γ)) (denoted V-ABD(Γ, PosLits) in Nordh & Zanuttini [19] and PQ-ABDUCTION(Γ) in Creignou & Zanuttini [18]). Given a set of variables A we let $\mathrm{Lits}(A) = \{x, \neg x \mid x \in A\}$ be the set of all (positive and negative) literals obtainable from A.

ABD(Γ)
INSTANCE: (φ, A, q), where φ is a Γ-formula, $A \subseteq \mathrm{Var}(\varphi)$, $q \in \mathrm{Var}(\varphi)$
QUESTION: Does there exist an $E \subseteq \mathrm{Lits}(A)$ such that

1. $\varphi \wedge \bigwedge E$ is satisfiable, and
2. $\varphi \wedge \bigwedge E \wedge \neg q$ is unsatisfiable.

Thus, the task is to find an explanation, which is a subset of literals, which is (1) consistent with the input formula, and (2) together with the input formula logically implies the manifestation q.

2.2. Size-Preserving Reductions and Subexponential Time

If A is a computational problem, then we let $I(A)$ be the set of problem instances and $\|I\|$ be the size of any $I \in I(A)$, i.e. the number of bits required to represent I. Many problems can in a natural way be viewed as problems of assigning values from a fixed finite set to a collection of variables. This is certainly the case for the problems under consideration in this article but it is also the case for various graph problems such as MAX-CUT and MAX INDEPENDENT SET. We call problems of this kind *variable problems* and let $\mathrm{Var}(I)$ denote the set of variables of an instance I.

Definition 1. *Let A_1 and A_2 be variable problems in NP. The function f from $I(A_1)$ to $I(A_2)$ is a* many-one linear variable reduction *(LV-reduction) with parameter $C \geq 0$ if*

1. *I is a yes-instance of A_1 if and only if $f(I)$ is a yes-instance of A_2,*
2. *$|\mathrm{Var}(f(I))| = C \cdot |\mathrm{Var}(I)| + O(1)$, and*
3. *$f(I)$ can be computed in time $O(\mathrm{poly}(\|I\|))$.*

LV-reductions can be seen as a restricted form of SERF-reductions [11]. The term CV-reduction is used to denote LV-reductions with parameter 1, and we write $A_1 \leq^{\mathrm{CV}} A_2$ to denote that the problem A_1 has an CV-reduction to A_2. If A_1 and A_2 are two NP-hard problems we say that A_1 is *at least as easy* as (or *not harder than*) A_2 if A_1 is solvable in $O(c^{|\mathrm{Var}(I)|} \cdot \mathrm{poly}(||I||))$ time whenever A_1 is solvable in $O(c^{|\mathrm{Var}(I)|} \cdot \mathrm{poly}(||I||))$ time. By definition, if $A_1 \leq^{\mathrm{CV}} A_2$ then A_1 is not harder than A_2 but the converse is not true in general. If A_1 and A_2 are two problems mutually CV-reducible to each other, i.e., $A_1 \leq^{\mathrm{CV}} A_2$ and $A_2 \leq^{\mathrm{CV}} A_1$, then we write $A_1 =^{\mathrm{CV}} A_2$.

A problem solvable in time $O(2^{c|\mathrm{Var}(I)|} \cdot \mathrm{poly}(||I||))$ for all $c > 0$ is a *subexponential problem*, and we let SE denote the class of all variable problems solvable in subexponential time. It is straightforward to prove that LV-reductions preserve subexponential complexity for all problems A and B considered in this article, in the sense that if A is LV-reducible to B then $A \in \mathrm{SE}$ if $B \in \mathrm{SE}$. Naturally, SE can be defined using other complexity parameters than $|\mathrm{Var}(I)|$ [11]. The conjecture that 3-SAT is not solvable in subexponential time is then known as the *exponential-time hypothesis* (ETH) [9].

2.3. Operations and Relations

We now define the most important classes of operations and relations. An operation f is called *arithmetical* if $f(y,x,x) = f(y,x,y) = f(x,x,y) = y$ for every $x,y \in \mathbb{B}$. The max function is defined as $\max(x,y) = 0$ if $x = y = 0$ and 1 otherwise. We often express a Boolean relation R as a logical formula whose satisfying assignment corresponds to the tuples of R, often using the notation $R(x_1,\ldots,x_n) \equiv \varphi(x_1,\ldots,x_n)$, where n is the arity of R and where $\varphi(x_1,\ldots,x_n)$ is a first-order formula with free variables $x_1,\ldots,x_n$. F and T are the two constant relations $\{(0)\}$ and $\{(1)\}$ while Neq denotes inequality, i.e. the relation $\{(0,1),(1,0)\}$. The relation n-EVEN is defined as $\{(x_1,\ldots,x_n) \in \mathbb{B}^n \mid \sum_{i=1}^{n} x_i \text{ is even}\}$. The relation n-ODD is defined dually. The relations OR^n and NAND^n are the relations corresponding to the clauses $(x_1 \vee \ldots \vee x_n)$ and $(\overline{x_1} \vee \ldots \vee \overline{x_n})$. For any n-ary relation R we let $R^{m\neq}$, $1 \leq m \leq n$, denote the $(n+m)$-ary relation defined as $R^{m\neq}(x_1,\ldots,x_{n+m}) \equiv R(x_1,\ldots,x_n) \wedge \text{Neq}(x_1,x_{n+1}) \wedge \ldots \wedge \text{Neq}(x_n,x_{n+m})$. We let $R_{1/3} = \{(0,0,1),(0,1,0),(1,0,0)\}$. Variables are typically named $x_1,\ldots,x_n$ or x, and as a convention we typically order the arguments of relations in such a way that any constant arguments occur as the last (one or two) arguments.

2.4. Clone Theory

An operation $f : \mathbb{B}^k \to \mathbb{B}$ is a *polymorphism* of a relation R if for every $\mathbf{t}_1,\ldots,\mathbf{t}_k \in R$ it holds that $f(\mathbf{t}_1,\ldots,\mathbf{t}_k) \in R$, where f is applied element-wise. In this case R is *closed*, or *invariant*, under f. For a set of functions F we define $\text{Inv}(\mathsf{F})$ (often abbreviated as IF) to be the set of all relations invariant under all functions in F. Dually, for a set of relations Γ, $\text{Pol}(\Gamma)$ is defined to be the set of polymorphisms of Γ. Sets of the form $\text{Pol}(\Gamma)$ are known as *clones* and sets of the form $\text{Inv}(\mathsf{F})$ are known as *co-clones*. A clone $\text{Pol}(\Gamma)$ can equivalently well be described as a set of functions which (1) contains every function which returns a fixed argument (*projections*), and (2) is closed under functional composition. These two conditions can also be combined to form a closure operator over functions, and a generating set of a clone is called a *base*. In the Boolean domain clones are fully determined due to Post [23]. See Table 2 for a comprehensive list of Boolean clones, and Figure 1 for a visualization of their inclusion structure. It is then known that the relationship between clones and co-clones constitute a *Galois connection* [24].

Theorem 1. *Let* Γ, Γ' *be sets of relations. Then* $\text{Inv}(\text{Pol}(\Gamma')) \subseteq \text{Inv}(\text{Pol}(\Gamma))$ *if and only if* $\text{Pol}(\Gamma) \subseteq \text{Pol}(\Gamma')$.

Co-clones can equivalently be described as sets containing all relations R definable through *primitive positive implementations* (pp-implementations) over a constraint language Γ, i.e. definitions of the form

$$R(x_1,\ldots,x_n) \equiv \exists y_1,\ldots,y_m . R_1(\mathbf{x}_1) \wedge \ldots \wedge R_k(\mathbf{x}_k),$$

where each $R_i \in \Gamma \cup \{\text{Eq}\}$ and each $\mathbf{x}_i$ is a tuple over $x_1,\ldots,x_n$, $y_1,\ldots,y_m$ and where $\text{Eq} = \{(0,0),(1,1)\}$. We typically use the expressions 'pp-implementations' and 'pp-definitions' interchangeably. As a shorthand we let $\langle\Gamma\rangle = \text{Inv}(\text{Pol}(\Gamma))$ for a constraint language Γ, and as can be verified this is the smallest set of relations closed under pp-implementations over Γ. In this case Γ is said to be a *base* of $\langle\Gamma\rangle$, and if a co-clone

admits a finite base it is said to be *finitely generated*; otherwise it is said to be *infinitely generated*.

It is known that if Γ' is finite and $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma')$ then $\mathrm{CSP}(\Gamma')$ is polynomial-time reducible to $\mathrm{CSP}(\Gamma)$ [25]. With this fact and Post's classification of all Boolean clones [23] Schaefer's dichotomy theorem [26] for $\mathrm{SAT}(\cdot)$ follows almost immediately. The complexity of MAX-ONES(Γ) is also preserved under finite expansions with relations pp-definable in Γ, and hence follow the standard Galois connection [15]. Note, however, that $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Gamma')$ does not imply that $\mathrm{CSP}(\Gamma') \leq^{\mathrm{CV}} \mathrm{CSP}(\Gamma)$ or that $\mathrm{CSP}(\Gamma')$ LV-reduces to $\mathrm{CSP}(\Gamma)$ since the number of constraints is not necessarily linearly bounded by the number of variables.

To study these restricted classes of reductions we are therefore in need of Galois connections with increased granularity. In Jonsson et al. [4] the $\mathrm{SAT}(\cdot)$ problem is studied with the Galois connection between closure under pp-definitions without existential quantification and *strong partial clones*. Here, we concentrate on the relational description and instead refer the reader to Schnoor & Schnoor [17], and Couceiro et al. [7], for the corresponding definitions on the functional side. If R is an n-ary Boolean relation and Γ a constraint language then R has a *quantifier-free primitive positive implementation* (qfpp-implementation) in Γ if

$$R(x_1,\ldots,x_n) \equiv R_1(\mathbf{x}_1) \wedge \ldots \wedge R_k(\mathbf{x}_k),$$

where each $R_i \in \Gamma \cup \{\mathrm{Eq}\}$ and each $\mathbf{x}_i$ is a tuple over $x_1,\ldots,x_n$. We use $\langle\Gamma\rangle_{\not\exists}$ to denote the smallest set of relations closed under qfpp-definability over Γ. If $\mathsf{IC} = \langle\mathsf{IC}\rangle_{\not\exists}$ then IC is sometimes called a *weak system*, or a *weak co-clone*. In Jonsson et al. [4] it is proven that if $\Gamma' \subseteq \langle\Gamma\rangle_{\not\exists}$ and if Γ and Γ' are both finite constraint languages then $\mathrm{SAT}(\Gamma') \leq^{\mathrm{CV}} \mathrm{SAT}(\Gamma)$. It is not hard to extend this result to the weighted W-MAX-ONES$(\cdot)$ problem since it follows the standard Galois connection, and therefore we use this fact without explicit proof. The only minor complication is that one has to ensure that the resulting variable is given a weight which matches the sum of the weights of the variables which it has replaced when equality constraints are removed, and variables are identified. Similarly, it is very straightforward to show that the same reduction technique works for $\mathrm{SSAT}(\cdot)$ and $\mathrm{ABD}(\cdot)$, and we thus have the following theorem.

Theorem 2. *Let Γ and Δ be finite Boolean constraint languages. If $\Gamma \subseteq \langle\Delta\rangle_{\not\exists}$, then*

1. $\mathrm{SAT}(\Gamma) \leq^{\mathrm{CV}} \mathrm{SAT}(\Delta)$,
2. W-MAX-ONES$(\Gamma) \leq^{\mathrm{CV}}$ W-MAX-ONES(Δ),
3. $\mathrm{SSAT}(\Gamma) \leq^{\mathrm{CV}} \mathrm{SSAT}(\Delta)$, *and*
4. $\mathrm{ABD}(\Gamma) \leq^{\mathrm{CV}} \mathrm{ABD}(\Delta)$.

Example 3. *We consider a simple example to highlight the type of reduction one obtains via Theorem 2. Recall that $R_{1/3}$ is the ternary relation $\{(0,0,1),(0,1,0),(1,0,0)\}$, define the relation $R_{1/3}^{\neq 01}$ as $\{(0,0,1,1,0,1),(0,1,0,1,0,1),(1,0,0,0,0,1)\}$, and observe that this relation is qfpp-definable over $R_{1/3}$ since $R_{1/3}^{\neq 01}(x_1,x_2,x_3,x_4,x_5,x_6) = R_{1/3}(x_1,x_2,x_3) \wedge R_{1/3}(x_5,x_5,x_6) \wedge R_{1/3}(x_1,x_4,x_5)$. Next, consider an instance (V,C) of (e.g.)* W-MAX-ONES$(R_{1/3})$. *Any constraint $R_{1/3}^{\neq 01}(x_i^1,x_i^2,x_i^3,x_i^4,x_i^5,x_i^6)$ is then replaced*

by the constraints prescribed by the above qfpp-definition, i.e., we introduce the constraints $R_{1/3}(x_i^1, x_i^2, x_i^3) \wedge R_{1/3}(x_i^5, x_i^5, x_i^6) \wedge R_{1/3}(x_i^1, x_i^4, x_i^5)$. *Crucially, while the number of constraints increases by a constant factor, the number of variables does not change at all, and it follows that the original instance has a solution of cost m if and only if the new instance has a solution of cost m.*

However, this does *not* hold (a priori) for the unweighted U-MAX-ONES$(\cdot)$ problem since one cannot safely remove equality constraints without using weights. We will circumvent this problem by introducing a special class of relations which are qfpp-definable without using equality.

Definition 2. *(Schnoor & Schnoor [17]) A* weak base Γ_w *of a co-clone* IC *is a base of* IC *with the property that for any base* Γ *of* IC *it holds that* $\Gamma_w \subseteq \langle \Gamma \rangle_{\not\equiv}$.

Weak bases for Boolean co-clones are well understood due to Lagerkvist [27], and Lagerkvist & Wahlström [28]. See Table 1 for a comprehensive list of weak bases. As a convention, we write R_{IC} for the weak base of the co-clone IC in Table 1. In addition, these weak bases satisfy an additional minimality condition which implies that they can be qfpp-defined without using equality. Hence, the aforementioned problem for U-MAX-ONES$(\cdot)$ does not occur, and we obtain the following theorem.

Theorem 3. *Let* IC *be a finitely generated Boolean co-clone and let* R_{IC} *be the weak base of* IC *from Table 1. Let* Γ *be an* arbitrary *finite base of* IC. *Then:*

1. $\mathrm{CSP}(R_{\mathsf{IC}}) \leq^{\mathrm{CV}} \mathrm{CSP}(\Gamma)$,
2. W-MAX-ONES$(R_{\mathsf{IC}}) \leq^{\mathrm{CV}}$ W-MAX-ONES(Γ),
3. U-MAX-ONES$(R_{\mathsf{IC}}) \leq^{\mathrm{CV}}$ U-MAX-ONES(Γ),
4. $\mathrm{SSAT}(R_{\mathsf{IC}}) \leq^{\mathrm{CV}} \mathrm{SSAT}(\Gamma)$, *and*
5. $\mathrm{ABD}(R_{\mathsf{IC}}) \leq^{\mathrm{CV}} \mathrm{ABD}(\Gamma)$.

Naturally, this is only impactful if the problem in question is intractable, since two polynomially solvable problems are trivially CV-reducible to each other.

Example 4. *Let us take a concrete example from Table 1. Recall the relational definitions from Section 2.3. We now see that* $R_{\mathsf{II}_2}(x_1, \ldots, x_6, x_7, x_8) \equiv R_{1/3}^{\neq\neq\neq}(x_1, \ldots, x_6) \wedge \mathrm{F}(x_7) \wedge \mathrm{T}(x_8)$ *and* $R_{\mathsf{IN}_2}(x_1, \ldots, x_8) \equiv \text{4-EVEN}^{4\neq}(x_1, \ldots, x_8) \wedge (x_1 x_4 \leftrightarrow x_2 x_3)$ *from Table 1 are the two relations (where the tuples in the relations are listed as rows)*

$$R_{\mathsf{II}_2} = \left\{ \begin{smallmatrix} 0&0&1&1&1&0&0&1 \\ 0&1&0&1&0&1&0&1 \\ 1&0&0&0&1&1&0&1 \end{smallmatrix} \right\} \quad \textit{and} \quad R_{\mathsf{IN}_2} = \left\{ \begin{smallmatrix} 0&0&0&0&1&1&1&1 \\ 0&0&1&1&1&1&0&0 \\ 0&1&0&1&1&0&1&0 \\ 1&1&1&1&0&0&0&0 \\ 1&1&0&0&0&0&1&1 \\ 1&0&1&0&0&1&0&1 \end{smallmatrix} \right\}.$$

3. The Easiest NP-Hard SSAT, ABD, MAX-ONES, and VCSP Problems

We will now study the complexity of SSAT, ABD, W-MAX-ONES, and VCSP with respect to CV-reductions (we return to the complexity of U-MAX-ONES$(\cdot)$ in Section 4). We remind the reader that constraint languages Γ and sets of cost functions

Table 1: Weak bases for all Boolean co-clones from Lagerkvist [27].

Co-clone	Weak base
IBF	$\{\mathrm{Eq}(x_1,x_2)\}$
IR_0	$\{\mathrm{F}(x_1)\}$
IR_1	$\{\mathrm{T}(x_1)\}$
IR_2	$\{\mathrm{F}(x_1)\wedge\mathrm{T}(x_2)\}$
IM	$\{(x_1\to x_2)\}$
IM_0	$\{(x_1\to x_2)\wedge\mathrm{F}(x_3)\}$
IM_1	$\{(x_1\to x_2)\wedge\mathrm{T}(x_3)\}$
IM_2	$\{(x_1\to x_2)\wedge\mathrm{F}(x_3)\wedge\mathrm{T}(x_4)\}$
$\mathsf{IS}_0^n, n\geq 2$	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge\mathrm{T}(x_{n+1})\}$
IS_0	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge\mathrm{T}(x_{n+1}) \mid n\geq 2\}$
$\mathsf{IS}_{02}^n, n\geq 2$	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge\mathrm{F}(x_{n+1})\wedge\mathrm{T}(x_{n+2})\}$
IS_{02}	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge\mathrm{F}(x_{n+1})\wedge\mathrm{T}(x_{n+2}) \mid n\geq 2\}$
$\mathsf{IS}_{01}^n, n\geq 2$	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge(x_{n+1}\to x_1\cdots x_n)\wedge\mathrm{T}(x_{n+2})\}$
IS_{01}	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge(x_{n+1}\to x_1\cdots x_n)\wedge\mathrm{T}(x_{n+2}) \mid n\geq 2\}$
$\mathsf{IS}_{00}^n, n\geq 2$	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge(x_{n+1}\to x_1\cdots x_n)\wedge\mathrm{F}(x_{n+2})\wedge\mathrm{T}(x_{n+3})\}$
IS_{00}	$\{\mathrm{OR}^n(x_1,\ldots,x_n)\wedge(x_{n+1}\to x_1\cdots x_n)\wedge\mathrm{F}(x_{n+2})\wedge\mathrm{T}(x_{n+3}) \mid n\geq 2\}$
$\mathsf{IS}_1^n, n\geq 2$	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge\mathrm{F}(x_{n+1})\}$
IS_1	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge\mathrm{F}(x_{n+1}) \mid n\geq 2\}$
$\mathsf{IS}_{12}^n, n\geq 2$	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge\mathrm{F}(x_{n+1})\wedge\mathrm{T}(x_{n+2})\}$
IS_{12}	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge\mathrm{F}(x_{n+1})\wedge\mathrm{T}(x_{n+2}) \mid n\geq 2\}$
$\mathsf{IS}_{11}^n, n\geq 2$	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge(x_1\to x_{n+1})\wedge\ldots\wedge(x_n\to x_{n+1})\wedge\mathrm{F}(x_{n+2})\}$
IS_{11}	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge(x_1\to x_{n+1})\wedge\ldots\wedge(x_n\to x_{n+1})\wedge\mathrm{F}(x_{n+2}) \mid n\geq 2\}$
$\mathsf{IS}_{10}^n, n\geq 2$	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge(x_1\to x_{n+1})\wedge\ldots\wedge(x_n\to x_{n+1})\wedge\mathrm{F}(x_{n+2})\wedge\mathrm{T}(x_{n+3})\}$
IS_{10}	$\{\mathrm{NAND}^n(x_1,\ldots,x_n)\wedge(x_1\to x_{n+1})\wedge\ldots\wedge(x_n\to x_{n+1})\wedge\mathrm{F}(x_{n+2})\wedge\mathrm{T}(x_{n+3}) \mid n\geq 2\}$
ID	$\{\mathrm{Neq}(x_1,x_2)\}$
ID_1	$\{\mathrm{Neq}(x_1,x_2)\wedge\mathrm{F}(x_3)\wedge\mathrm{T}(x_4)\}$
ID_2	$\{\mathrm{OR}^2(x_1,x_2)\wedge\mathrm{Neq}(x_1,x_3)\wedge\mathrm{Neq}(x_2,x_4)\wedge\mathrm{F}(x_5)\wedge\mathrm{T}(x_6)\}$
IL	$\{4\text{-EVEN}(x_1,x_2,x_3,x_4)\}$
IL_0	$\{3\text{-EVEN}(x_1,x_2,x_3)\wedge\mathrm{F}(x_4)\}$
IL_1	$\{3\text{-ODD}(x_1,x_2,x_3)\wedge\mathrm{T}(x_4)\}$
IL_2	$\{3\text{-EVEN}^{3\neq}(x_1,\ldots,x_6)\wedge\mathrm{F}(x_7)\wedge\mathrm{T}(x_8)\}$
IL_3	$\{4\text{-EVEN}^{4\neq}(x_1,\ldots,x_8)\}$
IV	$\{(\overline{x_1}\leftrightarrow\overline{x_2}\,\overline{x_3})\wedge(\overline{x_2}\vee\overline{x_3}\to\overline{x_4})\}$
IV_0	$\{(\overline{x_1}\leftrightarrow\overline{x_2}\,\overline{x_3})\wedge\mathrm{F}(x_4)\}$
IV_1	$\{(\overline{x_1}\leftrightarrow\overline{x_2}\,\overline{x_3})\wedge(\overline{x_2}\vee\overline{x_3}\to\overline{x_4})\wedge\mathrm{T}(x_5)\}$
IV_2	$\{(\overline{x_1}\leftrightarrow\overline{x_2}\,\overline{x_3})\wedge\mathrm{F}(x_4)\wedge\mathrm{T}(x_5)\}$
IE	$\{(x_1\leftrightarrow x_2x_3)\wedge(x_2\vee x_3\to x_4)\}$
IE_0	$\{(x_1\leftrightarrow x_2x_3)\wedge(x_2\vee x_3\to x_4)\wedge\mathrm{F}(x_5)\}$
IE_1	$\{(x_1\leftrightarrow x_2x_3)\wedge\mathrm{T}(x_4)\}$
IE_2	$\{(x_1\leftrightarrow x_2x_3)\wedge\mathrm{F}(x_4)\wedge\mathrm{T}(x_5)\}$
IN	$\{4\text{-EVEN}(x_1,x_2,x_3,x_4)\wedge x_1x_4\leftrightarrow x_2x_3\}$
IN_2	$\{4\text{-EVEN}^{4\neq}(x_1,\ldots,x_8)\wedge x_1x_4\leftrightarrow x_2x_3\}$
II	$\{(x_1\leftrightarrow x_2x_3)\wedge(\overline{x_4}\leftrightarrow\overline{x_2}\,\overline{x_3})\}$
II_0	$\{(\overline{x_1}\vee\overline{x_2})\wedge(\overline{x_1}\,\overline{x_2}\leftrightarrow\overline{x_3})\wedge\mathrm{F}(x_4)\}$
II_1	$\{(x_1\vee x_2)\wedge(x_1x_2\leftrightarrow x_3)\wedge\mathrm{T}(x_4)\}$
II_2	$\{R_{1/3}^{\neq\neq\neq}(x_1,\ldots,x_6)\wedge\mathrm{F}(x_7)\wedge\mathrm{T}(x_8)\}$

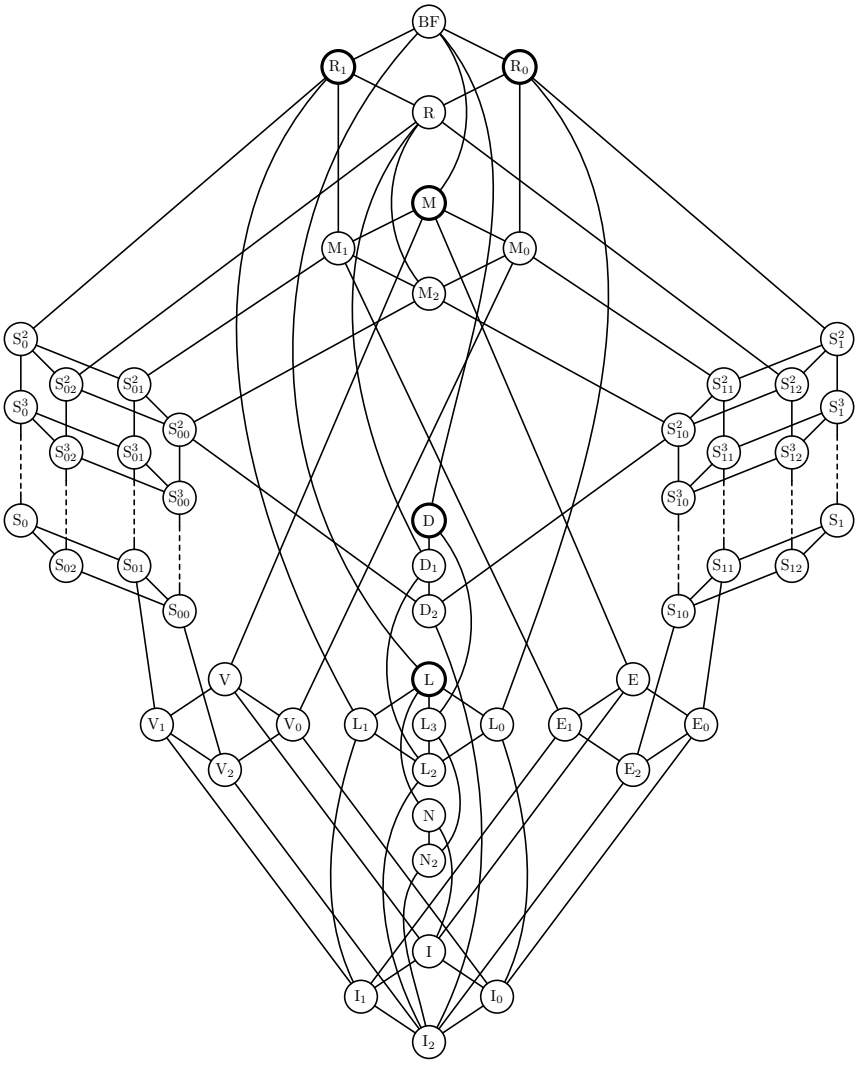

Figure 1: The lattice of Boolean clones.

Table 2: List of all Boolean clones with definitions and bases, where $\mathrm{id}(x) = x$ and $h_n(x_1,\ldots,x_{n+1}) = \bigvee_{i=1}^{n+1} x_1\cdots x_{i-1}x_{i+1}\cdots x_{n+1}$, $\mathrm{dual}(f)(a_1,\ldots,a_n) = 1 - f(\overline{a_1},\ldots,\overline{a_n})$. See e.g. [29].

Clone	Definition	Base
BF	All Boolean functions	$\{x\wedge y, \neg x\}$
R_0	$\{f \mid f$ is 0-reproducing$\}$	$\{x\wedge y, x\oplus y\}$
R_1	$\{f \mid f$ is 1-reproducing$\}$	$\{x\vee y, x\oplus y\oplus 1\}$
R_2	$\mathsf{R}_0\cap\mathsf{R}_1$	$\{x\vee y, x\wedge(y\oplus z\oplus 1)\}$
M	$\{f \mid f$ is monotonic$\}$	$\{x\vee y, x\wedge y, 0, 1\}$
M_1	$\mathsf{M}\cap\mathsf{R}_1$	$\{x\vee y, x\wedge y, 1\}$
M_0	$\mathsf{M}\cap\mathsf{R}_0$	$\{x\vee y, x\wedge y, 0\}$
M_2	$\mathsf{M}\cap\mathsf{R}_2$	$\{x\vee y, x\wedge y\}$
S_0^n	$\{f \mid f$ is 0-separating of degree $n\}$	$\{x\to y, \mathrm{dual}(h_n)\}$
S_0	$\{f \mid f$ is 0-separating$\}$	$\{x\to y\}$
S_1^n	$\{f \mid f$ is 1-separating of degree $n\}$	$\{x\wedge\neg y, h_n\}$
S_1	$\{f \mid f$ is 1-separating$\}$	$\{x\wedge\neg y\}$
S_{02}^n	$\mathsf{S}_0^n\cap\mathsf{R}_2$	$\{x\vee(y\wedge\neg z), \mathrm{dual}(h_n)\}$
S_{02}	$\mathsf{S}_0\cap\mathsf{R}_2$	$\{x\vee(y\wedge\neg z)\}$
S_{01}^n	$\mathsf{S}_0^n\cap\mathsf{M}$	$\{\mathrm{dual}(h_n), 1\}$
S_{01}	$\mathsf{S}_0\cap\mathsf{M}$	$\{x\vee(y\wedge z), 1\}$
S_{00}^n	$\mathsf{S}_0^n\cap\mathsf{R}_2\cap\mathsf{M}$	$\{x\vee(y\wedge z), \mathrm{dual}(h_n)\}$
S_{00}	$\mathsf{S}_0\cap\mathsf{R}_2\cap\mathsf{M}$	$\{x\vee(y\wedge z)\}$
S_{12}^n	$\mathsf{S}_1^n\cap\mathsf{R}_2$	$\{x\wedge(y\vee\neg z), h_n\}$
S_{12}	$\mathsf{S}_1\cap\mathsf{R}_2$	$\{x\wedge(y\vee\neg z)\}$
S_{11}^n	$\mathsf{S}_1^n\cap\mathsf{M}$	$\{h_n, 0\}$
S_{11}	$\mathsf{S}_1\cap\mathsf{M}$	$\{x\wedge(y\vee z), 0\}$
S_{10}^n	$\mathsf{S}_1^n\cap\mathsf{R}_2\cap\mathsf{M}$	$\{x\wedge(y\vee z), h_n\}$
S_{10}	$\mathsf{S}_1\cap\mathsf{R}_2\cap\mathsf{M}$	$\{x\wedge(y\vee z)\}$
D	$\{f \mid f$ is self-dual$\}$	$\{(x\wedge\neg y)\vee(x\wedge\neg z)\vee(\neg y\wedge\neg z)\}$
D_1	$\mathsf{D}\cap\mathsf{R}_2$	$\{(x\wedge y)\vee(x\wedge\neg z)\vee(y\wedge\neg z)\}$
D_2	$\mathsf{D}\cap\mathsf{M}$	$\{h_2\}$
L	$\{f \mid f$ is affine$\}$	$\{x\oplus y, 1\}$
L_0	$\mathsf{L}\cap\mathsf{R}_0$	$\{x\oplus y\}$
L_1	$\mathsf{L}\cap\mathsf{R}_1$	$\{x\oplus y\oplus 1\}$
L_2	$\mathsf{L}\cap\mathsf{R}_2$	$\{x\oplus y\oplus z\}$
L_3	$\mathsf{L}\cap\mathsf{D}$	$\{x\oplus y\oplus z\oplus 1\}$
V	$\{f \mid f$ is a disjunction or constants$\}$	$\{x\vee y, 0, 1\}$
V_0	$\mathsf{V}\cap\mathsf{R}_0$	$\{x\vee y, 0\}$
V_1	$\mathsf{V}\cap\mathsf{R}_1$	$\{x\vee y, 1\}$
V_2	$\mathsf{V}\cap\mathsf{R}_2$	$\{x\vee y\}$
E	$\{f \mid f$ is a conjunction or constants$\}$	$\{x\wedge y, 0, 1\}$
E_0	$\mathsf{E}\cap\mathsf{R}_0$	$\{x\wedge y, 0\}$
E_1	$\mathsf{E}\cap\mathsf{R}_1$	$\{x\wedge y, 1\}$
E_2	$\mathsf{E}\cap\mathsf{R}_2$	$\{x\wedge y\}$
N	$\{f \mid f$ depends on at most one variable$\}$	$\{\neg x, 0, 1\}$
N_2	$\mathsf{N}\cap\mathsf{R}_2$	$\{\neg x\}$
I	$\{f \mid f$ is a projection or a constant$\}$	$\{\mathrm{id}, 0, 1\}$
I_0	$\mathsf{I}\cap\mathsf{R}_0$	$\{\mathrm{id}, 0\}$
I_1	$\mathsf{I}\cap\mathsf{R}_1$	$\{\mathrm{id}, 1\}$
I_2	$\mathsf{I}\cap\mathsf{R}_2$	$\{\mathrm{id}\}$

Δ are always finite. We prove that for all of these problems we can identify identify a single language which is CV-reducible to every other NP-hard language. Hence, each such problem can be regarded as being 'maximally easy', in the sense that there cannot exist any other NP-hard problem within the class solvable within a strictly smaller running time. We will typically refer to each such problem as the 'easiest NP-hard problem' within the class, but remind the reader that we by this phrase do not claim that there cannot exist other problems with the same complexity (such a claim would be much too strong since it might even be the case that all problems are solvable in polynomial time, and would thus trivially be CV-reducible to each other).

Out of the infinite number of candidate languages generating different co-clones, we then prove that the language $\{R_{\mathsf{II}_2}\}$ defines the easiest $\mathrm{SSAT}(\cdot)$ problem and the easiest W-MAX-ONES$(\cdot)$ problem, while the language $\{R_{\mathsf{IV}_2}\}$ results in the easiest NP-complete $\mathrm{ABD}(\cdot)$ problem. Recall that we write R_{IC} for the weak base of the co-clone IC from Table 1. It might be interesting to note that $\{R_{\mathsf{II}_2}\}$ is the same language as for $\mathrm{SAT}(\cdot)$ [4], which might be contrary to intuition since one could be led to believe that the co-clones in the lower parts of the co-clone lattice, generated by very simple languages where the corresponding $\mathrm{SAT}(\cdot)$ problem is in P, would result in even easier problems. For example, $\mathrm{SAT}(\Gamma)$ is trivially in P for $\langle\Gamma\rangle \in \{\mathsf{II}_0, \mathsf{II}_1, \mathsf{II}, \mathsf{IN}\}$ since Γ is then closed under a constant operation, while $\mathrm{SSAT}(\Gamma)$ is NP-hard for these cases, and a priori there is no clear reason why the problem $\mathrm{SSAT}(R_{\mathsf{II}_2})$ should be easier.

For the $\mathrm{VCSP}(\cdot)$ problem the situation is a bit different since it is defined in the setting of cost functions, rather than constraint languages, and here we instead prove that the familiar problem MAX CUT results in the easiest $\mathrm{VCSP}(\cdot)$ problem. In addition, we summarize and relate the results together by illustrating the complexity landscape of these problems in Section 3.5.

3.1. The Surjective SAT *Problem*

Recall that the $\mathrm{SSAT}(\Gamma)$ problem is the problem of determining whether a set of constraints over a Boolean Γ admits a surjective solution, which in the Boolean domain simply implies that the solution in question is not constant. While very little is known known regarding the complexity of this problem for arbitrary finite domains [30], a complete complexity dichotomy has been established for $\mathrm{SSAT}(\Gamma)$. Say that a Boolean operation $f : \{0,1\}^k \to \{0,1\}$ is *essentially unary* if there exists a unary function g and $i \in \{1,\ldots,k\}$ such that $f(x_1,\ldots,x_i,\ldots,x_n) = g(x_i)$ for all $x_1,\ldots,x_i,\ldots,x_n \in \{0,1\}$. We then have the following result from Creignou et al. [16].

Theorem 4. $\mathrm{SSAT}(\Gamma)$ *is NP-complete if* $\mathrm{Pol}(\Gamma)$ *consists of essentially unary operations and is in P otherwise.*

See Figure 2 for a visualization of this dichotomy, and observe that the problem is NP-complete for exactly six clones. We will now prove that $\mathrm{SSAT}(R_{\mathsf{II}_2})$ results in the easiest NP-complete $\mathrm{SSAT}(\Gamma)$ problem.

Theorem 5. *Let* Γ *be a finite constraint language such that* $\mathrm{SSAT}(\Gamma)$ *is NP-hard. Then* $\mathrm{SSAT}(R_{\mathsf{II}_2}) \leq^{\mathrm{CV}} \mathrm{SSAT}(\Gamma)$.

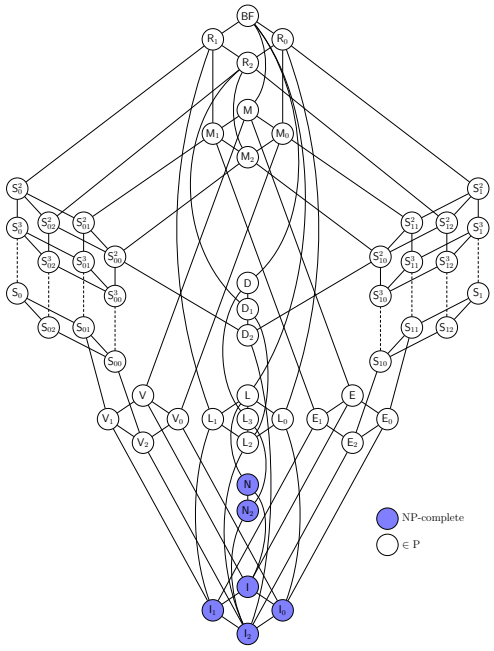

Figure 2: The complexity of $\mathrm{SSAT}(S)$.

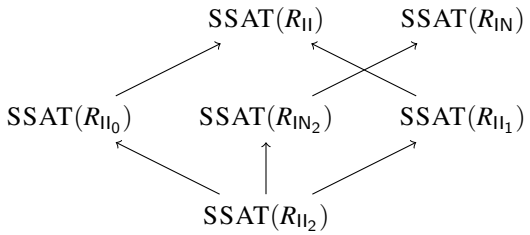

Figure 3: The size preserving reduction sequences for SSAT.

Proof. Recall from Theorem 3 that each weak base R_{IC} results in the easiest NP-hard $\mathrm{SSAT}(\cdot)$ problem for the co-clone IC. Hence, all we have to show is that $\mathrm{SSAT}(R_{\mathsf{II}_2}) \leq^{\mathrm{CV}} \mathrm{SSAT}(R_{\mathsf{IC}})$ for every IC where $\mathrm{SSAT}(R_{\mathsf{IC}})$ is NP-hard.

We first observe that given an instance of $\mathrm{SSAT}(R_{\mathsf{II}_2})$, we may assume (1) that $|V| \geq 2$, and (2) that every variable occurs in at least one constraint. The other cases are easy. Indeed, if $|V| \leq 1$ we simply output an arbitrary unsatisfiable instance. Otherwise, if there exists some variable that does *not* occur in any constraint, then surjectivity is not an issue. In this case we introduce a fresh variable z and add the constraint $R_{\mathsf{II}_2}(z,z,x,x,x,z,z,x)$ for every unconstrained variable x. We can assume the properties (1) and (2) also when given an instance of $\mathrm{SSAT}(R_{\mathsf{II}_0})$, $\mathrm{SSAT}(R_{\mathsf{II}_1})$ or $\mathrm{SSAT}(R_{\mathsf{IN}_2})$, for very similar reasons.

We give the reductions in terms of local constraint transformations and note that

in total, they introduce a constant number of global variables and run in polynomial time since all involved constraint languages are finite. Let Y_1, Y_2 denote two fresh variables. Each implementation is written as $R(x_1,\ldots,x_n) \mapsto \phi$, where $R(x_1,\ldots,x_n)$ is the relation in question and ϕ is a qfpp-definition (without equality) over the variables $\{x_1,\ldots,x_n\} \cup \{Y_1, Y_2\}$. Using these implementations we can then obtain a CV-reduction by (1) introducing the two fresh variables Y_1 and Y_2 and (2) replacing each constraint in the input instance by the constraints prescribed by the qfpp-definition. The reduction sequences are summarized in Figure 3.

With R_{IN_2} we use the following implementation:

$$R_{\mathsf{II}_2}(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8) \mapsto R_{\mathsf{IN}_2}(x_7,x_1,x_2,x_6,x_8,x_4,x_5,x_3) \\ \wedge R_{\mathsf{IN}_2}(Y_1,x_7,x_7,Y_1,Y_2,x_8,x_8,Y_2).$$

The second constraint ensures that the constraints $\mathrm{Eq}(x_7,Y_1)$, $\mathrm{Eq}(x_8,Y_2)$ and $\mathrm{Neq}(Y_1,Y_2)$ hold. If f is a satisfying assignment where $f(Y_2) = 0$ then we may use the assignment $f'(x) = 1 - f(x)$ instead since the complement of a valid assignment is also a valid assignment for languages closed under complement.

To implement R_{IN_2} with R_{IN} use

$$R_{\mathsf{IN}_2}(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8) \mapsto R_{\mathsf{IN}}(x_1,x_2,x_3,x_4) \wedge R_{\mathsf{IN}}(Y_1,x_1,x_5,Y_2) \\ \wedge R_{\mathsf{IN}}(Y_1,x_2,x_6,Y_2) \wedge R_{\mathsf{IN}}(Y_1,x_3,x_7,Y_2) \wedge R_{\mathsf{IN}}(Y_1,x_4,x_8,Y_2).$$

Note that $R_{\mathsf{IN}}(0,x,y,1) \Leftrightarrow R_{\mathsf{IN}}(1,x,y,0) \Leftrightarrow \mathrm{Neq}(x,y)$, $R_{\mathsf{IN}}(0,x,y,0) \Leftrightarrow (\overline{x} \wedge \overline{y})$, and $R_{\mathsf{IN}}(1,x,y,1) \Leftrightarrow (x \wedge y)$. Hence, any solution f where $f(Y_1) = f(Y_2) = a$, assigns all variables a and is therefore trivial.

The reductions from R_{II_2} to R_{II_0} and R_{II_1} are very similar, hence we only include the latter. The implementation is

$$R_{\mathsf{II}_2}(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8) \mapsto R_{\mathsf{II}_1}(x_6,x_5,x_1,x_8) \wedge R_{\mathsf{II}_1}(Y_1,Y_2,x_7,x_8) \\ \wedge R_{\mathsf{II}_1}(x_1,x_4,Y_1,Y_2) \wedge R_{\mathsf{II}_1}(x_2,x_5,Y_1,Y_2) \wedge R_{\mathsf{II}_1}(x_3,x_6,Y_1,Y_2).$$

One verifies that $R_{\mathsf{II}_1}(x,y,0,1) \Leftrightarrow \mathrm{Neq}(x,y)$ and $R_{\mathsf{II}_1}(1,1,x,y) \Leftrightarrow R_{\mathsf{II}_1}(x,y,1,1) \Leftrightarrow (x \wedge y)$. Therefore, any solution f where $f(Y_1) = f(Y_2) = 1$ assigns all variables 1 and is therefore trivial.

For the last step it is easy to verify that R_{II} can implement both R_{II_0} and R_{II_1} with one global variable in each case. □

Interestingly, $\mathrm{SSAT}(R_{\mathsf{II}_2})$ is CV-interreducible to the easiest NP-hard SAT problem, $\mathrm{SAT}(R_{\mathsf{II}_2})$, since every instance of the latter admits a solution if and only if it admits a surjective solution (provided that every variable appears in at least one constraint). To see this, simply observe that a R_{II_2}-constraint which is not trivially unsatisfiable, can only be satisfied by a surjective assignment.

3.2. The Propositional Abduction Problem

The complexity of propositional abduction is well-studied, and for constraint languages this problem enjoys a trichotomy between Σ_2^{p}-complete, NP-complete, and tractable cases (recall that we by constraint language always mean a *finite* set of relations).

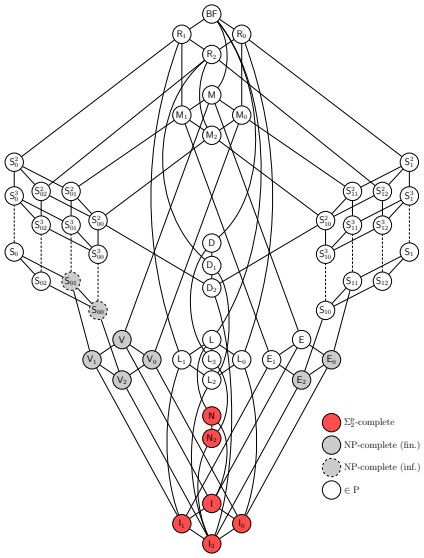

Figure 4: The complexity of $\mathrm{ABD}(\Gamma)$: the Σ_2^{p}-complete cases are colored in red, the NP-complete cases for finitely generated co-clones are colored in gray, the NP-complete cases for infinitely generated co-clones are drawn as dashed, gray circles, and the tractable cases are drawn in white.

Theorem 6 (Creignou & Zanuttini [19])**.** *Let* Γ *be a Boolean (finite) constraint language. Then* $\mathrm{ABD}(\Gamma)$ *is NP-complete if* $\langle\Gamma\rangle \in \{\mathsf{IV}_2, \mathsf{IV}_0, \mathsf{IV}_1, \mathsf{IV}, \mathsf{IE}_2, \mathsf{IE}_0\}$*,* Σ_2^{p}*-complete if* $\langle\Gamma\rangle \in \{\mathsf{II}_2, \mathsf{II}_0, \mathsf{II}_1, \mathsf{II}, \mathsf{IN}_2, \mathsf{IN}\}$*, and in* P *otherwise.*

Here, finiteness is not merely a simplifying assumption, but absolutely crucial, since it is known that there exists an infinite set of relations Γ such that $\mathrm{ABD}(\Gamma)$ is NP-intermediate [31]. In the sequel, we thus concentrate on the case when $\mathrm{ABD}(\Gamma)$ is NP-complete and when Γ is finite.

Theorem 7. *If* Γ *is a finite constraint language such that* $\mathrm{ABD}(\Gamma)$ *is NP-complete, then* $\mathrm{ABD}(R_{\mathsf{IE}_2}) \leq^{\mathrm{CV}} \mathrm{ABD}(\Gamma)$.

Our proof makes use of the following two lemmas.

Lemma 1. *Suppose* $R(x_1,\ldots,x_n) = R'(x_{\pi(1)},\ldots,x_{\pi(n)}) \wedge x_j$ *for some* j *and some function* $\pi : \{1,\ldots,n\} \to \{1,\ldots,m\}$*, and that the binary relation* $x_1 \vee \neg x_2$ *is pp-definable from* R'*. Then* $\mathrm{ABD}(R) \leq^{\mathrm{CV}} \mathrm{ABD}(R')$.

Proof. Let (φ, A, q) be an instance of $\mathrm{ABD}(R)$. We construct a R'-formula φ' over the variables $\mathrm{Var}(\varphi) \cup \{Y\}$, where Y is a fresh variable, as follows: For every constraint $R(x_1,\ldots,x_n)$ in φ, create the constraints $R'(x_{\pi(1)},\ldots,x_{\pi(n)}) \wedge (x_j = Y)$, and finally add the constraint $(Y \vee \neg q)$. The very last constraint is over a relation that is pp-definable from R', and can therefore be eliminated while only introducing a constant number of additional variables. The equality constraints can all be eliminated by identification of variables.

We claim that (φ, A, q) is a Yes-instance of $\mathrm{ABD}(R)$ if and only if $(\varphi', A \cup \{Y\}, q)$ is a Yes-instance of $\mathrm{ABD}(R')$. For $(\Rightarrow)$, if $\varphi \wedge \bigwedge E$ is satisfiable and $\varphi \wedge \bigwedge E \wedge \neg q$ is unsatisfiable, then $\varphi' \wedge \bigwedge(E \cup \{Y\})$ is satisfiable and $\varphi' \wedge \bigwedge(E \cup \{Y\}) \wedge \neg q$ is unsatisfiable. For $(\Leftarrow)$, assume $\varphi' \wedge \bigwedge E'$ is satisfiable and $\varphi' \wedge \bigwedge E' \wedge \neg q$ is unsatisfiable. Then $\varphi' \wedge \bigwedge E' \wedge Y$ must also be satisfiable (since $(Y \vee \neg q)$ is part of φ'), and $\varphi' \wedge \bigwedge E' \wedge Y \wedge \neg q$ must be unsatisfiable (since it is a restriction). From this follows that $\varphi \wedge \bigwedge(E' \setminus \{Y, \neg Y\})$ is satisfiable and $\varphi \wedge \bigwedge(E' \setminus \{Y, \neg Y\}) \wedge \neg q$ is unsatisfiable. □

Let $\mathrm{flip} : \{0,1\} \to \{0,1\}$ be the function that maps 0 to 1 and 1 to 0. We define this function also on relations through $\mathrm{flip}(R) = \{(\mathrm{flip}(x_1), \ldots, \mathrm{flip}(x_n)) : (x_1, \ldots, x_n) \in R\}$.

Lemma 2. *Suppose $R(x_1, \ldots, x_n) = \mathrm{flip}(R')(x_{\pi(1)}, \ldots, x_{\pi(n)})$ for some permutation π, and that the binary relation $x_1 \vee x_2$ is pp-definable from R'. Then* $\mathrm{ABD}(R) \leq^{\mathrm{CV}} \mathrm{ABD}(R')$.

Proof. Let (φ, A, q) be an instance of $\mathrm{ABD}(R)$. We construct a R'-formula φ' on variables $\mathrm{Var}(\varphi) \cup \{q'\}$, where q' is a fresh variable, as follows. For each constraint $R(x_1, \ldots, x_n)$ in φ, create the constraint $R'(x_{\pi(1)}, \ldots, x_{\pi(n)})$. Finally, add the constraint $(q \vee q')$. The last constraint can be eliminated at the cost of only a constant number of additional variables since is over a relation that is pp-definable from R' (possibly introduced equality constraints can be eliminated by identification of variables).

We claim that (φ, A, q) is a Yes-instance of $\mathrm{ABD}(R)$ if and only if (φ', A, q') is a Yes-instance of $\mathrm{ABD}(R')$. For $(\Rightarrow)$, let E be an arbitrary set of literals and define $\mathrm{flip}(E) = \{\neg \ell : \ell \in E\}$. If $\varphi \wedge \bigwedge E$ is satisfiable and $\varphi \wedge \bigwedge E \wedge \neg q$ is unsatisfiable, then $\varphi' \wedge \bigwedge \mathrm{flip}(E)$ is satisfiable and $\varphi' \wedge \bigwedge \mathrm{flip}(E) \wedge q$ is unsatisfiable. This means, since $(q \vee q')$ is part of φ', also that $\varphi' \wedge \bigwedge \mathrm{flip}(E) \wedge \neg q'$ is unsatisfiable. For $(\Leftarrow)$, again let E' be an arbitrary set of literals and define $\mathrm{flip}(E') = \{\neg \ell : \ell \in E'\}$. Assume $\varphi' \wedge \bigwedge E'$ is satisfiable and $\varphi' \wedge \bigwedge E' \wedge \neg q'$ is unsatisfiable. Since the only constraint in φ' in which q' occurs is $(q \vee q')$, this means that also $\varphi' \wedge \bigwedge E' \wedge q$ is unsatisfiable. Hence, $\varphi \wedge \bigwedge \mathrm{flip}(E')$ is satisfiable and $\varphi \wedge \bigwedge \mathrm{flip}(E') \wedge \neg q$ is unsatisfiable. □

We can now prove the theorem.

Proof of Theorem 7. According to Theorem 6 (see also Figure 4) there are six distinct cases to consider when the propositional abduction problem is NP-complete. These are the co-clones IV, IV_0, IV_1, IV_2, IE_0, and IE_2. For each such co-clone IC let R_{IC} denote the weak base from Table 1, and recall from Theorem 3 that $\mathrm{ABD}(R_{\mathsf{IC}}) \leq^{\mathrm{CV}} \mathrm{ABD}(\Gamma)$ when Γ is a finite base of IC.

Note that

$$
\begin{aligned}
R_{\mathsf{IE}_2}(x_1, x_2, x_3, x_4, x_5) &= R_{\mathsf{IE}_0}(x_1, x_2, x_3, x_5, x_4) \wedge x_5, \\
R_{\mathsf{IV}_1}(x_1, x_2, x_3, x_4, x_5) &= R_{\mathsf{IV}}(x_1, x_2, x_3, x_4) \wedge x_5, \\
R_{\mathsf{IV}_2}(x_1, x_2, x_3, x_4, x_5) &= R_{\mathsf{IV}_0}(x_1, x_2, x_3, x_4) \wedge x_5.
\end{aligned}
$$

One easily verifies that the binary relation $x_1 \vee \neg x_2$ is pp-definable from each of R_{IE_0}, R_{IV} and R_{IV_0}. From Lemma 1 it follows that $\mathrm{ABD}(R_{\mathsf{IE}_2}) \leq^{\mathrm{CV}} \mathrm{ABD}(R_{\mathsf{IE}_0})$, $\mathrm{ABD}(R_{\mathsf{IV}_1}) \leq^{\mathrm{CV}} \mathrm{ABD}(R_{\mathsf{IV}})$, and $\mathrm{ABD}(R_{\mathsf{IV}_2}) \leq^{\mathrm{CV}} \mathrm{ABD}(R_{\mathsf{IV}_0})$.

Note also that

$$R_{\mathsf{IE}_2}(x_1,x_2,x_3,x_4,x_5) = \mathrm{flip}(R_{\mathsf{IV}_2})(x_1,x_2,x_3,x_5,x_4),$$
$$R_{\mathsf{IE}_0}(x_1,x_2,x_3,x_4,x_5) = \mathrm{flip}(R_{\mathsf{IV}_1})(x_1,x_2,x_3,x_4,x_5).$$

It is easy to check that the binary relation $x_1 \vee x_2$ is pp-definable both from R_{IV_2} and from R_{IV_1}. Hence, by Lemma 2 it follows that $\mathrm{ABD}(R_{\mathsf{IE}_2}) \leq^{\mathrm{CV}} \mathrm{ABD}(R_{\mathsf{IV}_2})$ and $\mathrm{ABD}(R_{\mathsf{IE}_0}) \leq^{\mathrm{CV}} \mathrm{ABD}(R_{\mathsf{IV}_1})$.

Figure 5 summarises the reductions. □

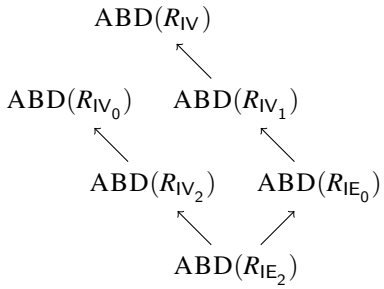

Figure 5: Size preserving reductions for the NP-complete cases of $\mathrm{ABD}(\Gamma)$.

3.3. The MAX-ONES *Problem*

Here we use a slight reformulation of Khanna et al.'s [15] complexity classification of the MAX-ONES problem expressed in terms of polymorphisms. Say that a constraint language is *1-closed* if it is preserved by the constant Boolean function 1.

Theorem 8 ([15])**.** *Let* Γ *be a finite Boolean constraint language.* $\textsc{Max-Ones}(\Gamma)$ *is solvable in polynomial time if* Γ *is* 1*-closed, closed under* max*, or closed under an arithmetical operation, and is NP-hard otherwise.*

See Figure 6 for a visualization of this dichotomy theorem. The theorem holds for both the weighted and the unweighted version of the problem and showcases the strength of the algebraic method since it tells us exactly which cases are intractable, and which properties they satisfy.

Theorem 9. *Let* Γ *be a finite constraint language such that* $\textsc{Max-Ones}(\Gamma)$ *is NP-complete. Then there exists* $R \in \{R_{\mathsf{IS}_1^2}, R_{\mathsf{II}_2}, R_{\mathsf{IN}_2}, R_{\mathsf{IL}_0}, R_{\mathsf{IL}_2}, R_{\mathsf{IL}_3}, R_{\mathsf{ID}_2}\}$ *(relations defined in Table 2) such that* $\textsc{Max-Ones}(\{R\}) \leq^{\mathrm{CV}} \textsc{Max-Ones}(\Gamma)$*. This holds both for the weighted and the unweighted version of the problem.*

We will prove the theorem with the help of the following lemmas, all of which holds both in the weighted and unweighted case.

Lemma 3. *If* Γ *is a constraint language and R is a relation that is qfpp-definable without equality in* Γ*, then* $\textsc{Max-Ones}(\{R\}) \leq^{\mathrm{CV}} \textsc{Max-Ones}(\Gamma)$.

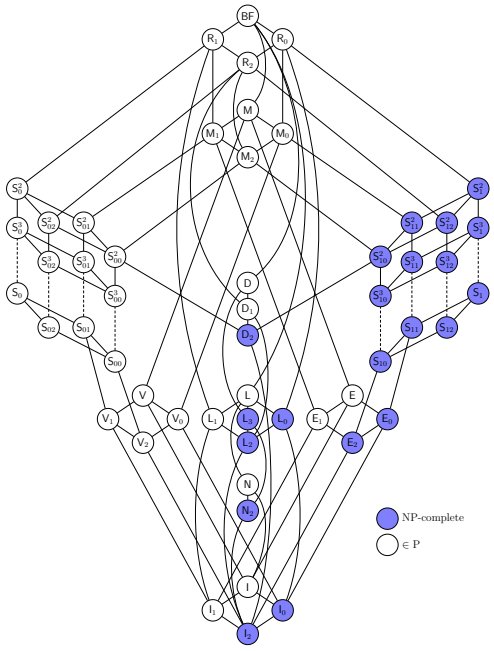

Figure 6: The complexity of MAX-ONES(S).

Proof. Since R is qfpp-definable without equality in Γ, every constraint over R can be replaced with a finite set of constraints over Γ. □

Note that if equality relations were allowed in the qfpp-formula for R, then these would have to be eliminated. The natural way to do this is by identification of variables, but for this approach to work we need general weights for the variables.

Lemma 4. *If R and R' are relations that satisfy*

$$R(x_1,\ldots,x_n) \Rightarrow R'(x_1,\ldots,x_n,0,1), \tag{1}$$

$$R'(x_1,\ldots,x_n,y_0,y_1) \Rightarrow R(x_1,\ldots,x_n) \wedge F(y_0) \wedge T(y_1) \vee \bigwedge_{i=1}^{n} F(x_i) \wedge F(y_0) \wedge F(y_1), \tag{2}$$

then MAX-ONES($\{R\}$) $\leq^{\mathrm{CV}}$ MAX-ONES($\{R'\}$).

Proof. To see why this is true, consider the following reduction. Given an instance of MAX-ONES($\{R\}$) over n variables, we construct an instance of MAX-ONES($\{R'\}$) over the same set of variables (preserving variable weights), extended with two fresh, unit-weight, variables y_0 and y_1. Let w_{free} be the summed weights of every variable in the given instance that does not occur in any constraint. For every constraint $R(x_1,\ldots,x_\ell)$ in the given instance we add the constraint $R'(x_1,\ldots,x_\ell,y_0,y_1)$ to the new instance. From (1) and the fact that a solution always can be changed to map each unconstrained variable to the value 1, it follows that if the original instance has a solution with objective

value $\geq k$, then the new instance has a solution with objective value $\geq k+1$, and hence $\geq \max(k, w_{\text{free}})+1$ since $k \geq w_{\text{free}}$ must hold in the original instance (since setting all unconstrained variables to 1 is a solution).

For the other direction, consider a solution to the new instance with objective value $\geq \max(k, w_{\text{free}})+1$. Assume first that y_1 is assigned the value 0. By (2) it follows that every variable that occurs in a constraint must be assigned the value 0, so the objective is at most w_{free}. This is a contradiction, so y_1 must be assigned the value 1. It then follows from (2) that there is a solution the the original instance with objective value $\geq \max(k, w_{\text{free}})$. Hence, the original instance has a solution with objective value $\geq k$ if and only if the new instance has a solution with objective value $\geq \max(w_{\text{free}}, k)+1$. □

We now have all the results in place that we need to prove the theorem.

Proof of Theorem 9. By Theorem 8 in combination with Table 2 and Figure 1 it follows that MAX-ONES(Γ) is NP-complete if and only if $\langle\Gamma\rangle \supseteq \mathsf{IS}_1^2$ or if $\langle\Gamma\rangle \in \{\mathsf{IL}_0, \mathsf{IL}_3, \mathsf{IL}_2, \mathsf{IN}_2\}$. Then, in principle, for every co-clone we have to decide which language is CV-reducible to every other base of the co-clone, but since a weak base always has this property, we can eliminate a lot of tedious work and directly consult the precomputed relations in Table 1. From this we first see that $\langle R_{\mathsf{IS}_1^2}\rangle_{\not\equiv} \subset \langle R_{\mathsf{IS}_1^n}\rangle_{\not\equiv}$, $\langle R_{\mathsf{IS}_{12}^2}\rangle_{\not\equiv} \subset \langle R_{\mathsf{IS}_{12}^n}\rangle_{\not\equiv}$, $\langle R_{\mathsf{IS}_{11}^2}\rangle_{\not\equiv} \subset \langle R_{\mathsf{IS}_{11}^n}\rangle_{\not\equiv}$ and $\langle R_{\mathsf{IS}_{10}^2}\rangle_{\not\equiv} \subset \langle R_{\mathsf{IS}_{10}^n}\rangle_{\not\equiv}$ for every $n \geq 3$. Hence, in the four infinite chains IS_1^n, IS_{12}^n, IS_{11}^n, IS_{10}^n we only have to consider the bottom-most co-clones IS_1^2, IS_{12}^2, IS_{11}^2, IS_{10}^2.

For $R_{\mathsf{IS}_1^2}(x_1,x_2,x_3)$ we can define relations $R'_{\mathsf{IS}_1^2}(x_1,x_2,x_3,y_0,y_1)$ with $R_{\mathsf{IS}_{12}^2}$, $R_{\mathsf{IS}_{11}^2}$, $R_{\mathsf{IS}_{10}^2}$, R_{IE_2}, and R_{IE_0}, satisfying the requirements of Lemma 4 as follows

$$
\begin{aligned}
R'_{\mathsf{IS}_1^2}(x_1,x_2,x_3,y_0,y_1) &\equiv R_{\mathsf{IS}_{12}^2}(x_1,x_2,x_3,y_1) \wedge R_{\mathsf{IS}_{12}^2}(x_1,x_2,y_0,y_1),\\
R'_{\mathsf{IS}_1^2}(x_1,x_2,x_3,y_0,y_1) &\equiv R_{\mathsf{IS}_{11}^2}(x_1,x_2,y_1,x_3) \wedge R_{\mathsf{IS}_{11}^2}(x_1,x_2,y_1,y_0),\\
R'_{\mathsf{IS}_1^2}(x_1,x_2,x_3,y_0,y_1) &\equiv R_{\mathsf{IS}_{10}^2}(x_1,x_2,y_1,x_3,y_1) \wedge R_{\mathsf{IS}_{10}^2}(x_1,x_2,y_1,y_0,y_1),\\
R'_{\mathsf{IS}_1^2}(x_1,x_2,x_3,y_0,y_1) &\equiv R_{\mathsf{IE}_2}(x_3,x_1,x_2,x_3,y_1) \wedge R_{\mathsf{IE}_2}(x_3,x_1,x_2,y_0,y_1),\\
R'_{\mathsf{IS}_1^2}(x_1,x_2,x_3,y_0,y_1) &\equiv R_{\mathsf{IE}_0}(x_3,x_1,x_2,y_1,x_3) \wedge R_{\mathsf{IE}_0}(y_0,x_1,x_2,y_1,y_0),
\end{aligned}
$$

and similarly a relation R'_{II_2} using R_{II_0}, also satisfying the requirements of Lemma 4, as follows

$$
\begin{aligned}
R'_{\mathsf{II}_2}(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,y_0,y_1) \equiv{}& R_{\mathsf{II}_0}(x_1,x_2,x_6,x_7) \wedge R_{\mathsf{II}_0}(x_7,x_8,y_1,y_0)\\
&\wedge R_{\mathsf{II}_0}(x_1,x_4,y_1,y_0) \wedge R_{\mathsf{II}_0}(x_2,x_5,y_1,y_0) \wedge R_{\mathsf{II}_0}(x_3,x_6,y_1,y_0).
\end{aligned}
$$

Since these are qfpp-definitions without equality, the corresponding CV-reductions to the bases from Table 2 follows from Lemma 3.

Using Figure 1 we then see that the only remaining cases for Γ when $\langle\Gamma\rangle \supseteq \mathsf{IS}_1^2$ is when $\langle\Gamma\rangle = \mathsf{II}_2$ or when $\langle\Gamma\rangle = \mathsf{ID}_2$. This concludes the proof. □

Using qfpp-implementations to further decrease the set of relations in Theorem 9 appears difficult and we therefore make use of more powerful implementations. Let

Table 3: arg max definitions for W-MAX-ONES($\cdot$).

R_1	R_2	arg max definition of R_1 with R_2
R_{II_2}	R_{IN_2}	$\arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_7,x_1,x_2,x_6,x_8,x_4,x_5,x_3)\in R_{\mathsf{IN}_2}} x_8$
R_{II_2}	R_{ID_2}	$\arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_5,x_4,x_2,x_1,x_7,x_8),(x_6,x_4,x_3,x_1,x_7,x_8),(x_6,x_5,x_3,x_2,x_7,x_8)\in R_{\mathsf{ID}_2}} (x_1+x_2+x_3)$
R_{II_2}	R_{IL_2}	$\arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_4,x_5,x_6,x_1,x_2,x_3,x_7,x_8)\in R_{\mathsf{IL}_2}} (x_4+x_5+x_6)$
R_{IL_2}	R_{IL_3}	$\arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_7,x_1,x_2,x_3,x_8,x_4,x_5,x_6)\in R_{\mathsf{IL}_3}} x_8$
R_{IL_2}	R_{IL_0}	$\arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_1,x_2,x_3,x_7),(x_8,x_1,x_4,x_7),(x_8,x_2,x_5,x_7),(x_8,x_3,x_6,x_7)\in R_{\mathsf{IL}_0}} x_8$
R_{II_2}	$R_{\mathsf{IS}_1^2}$	$\arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_1,x_2,x_7),(x_1,x_3,x_7),(x_2,x_3,x_7),(x_1,x_4,x_7),(x_2,x_5,x_7),(x_3,x_6,x_7)\in R_{\mathsf{IS}_1^2}} \left(\sum_{i=1}^{8} x_i+\sum_{i=1}^{3} x_i\right)$

Optsol(I) be the set of all optimal solutions of a W-MAX-ONES(Γ) instance I. A relation R has a *weighted pp definition* (wpp-definition) [21] in Γ if there exists an instance I of W-MAX-ONES(Γ) on variables V such that

$$R = \{(\phi(v_1),\ldots,\phi(v_m)) \mid \phi \in \text{Optsol}(I)\}$$

for some $v_1,\ldots,v_m \in V$. The set of all relations wpp-definable in Γ is denoted $\langle\Gamma\rangle_w$ and we furthermore have that if $\Gamma' \subseteq \langle\Gamma\rangle_w$ is finite then W-MAX-ONES(Γ') is polynomial-time reducible to W-MAX-ONES(Γ) [21]. If there is a W-MAX-ONES(Γ) instance I on V such that

$$R = \{(\phi(v_1),\ldots,\phi(v_m)) \mid \phi \in \text{Optsol}(I)\}$$

for $v_1,\ldots,v_m \in V$ satisfying $\{v_1,\ldots,v_m\} = V$, then we say that R is qfwpp-definable in Γ. We use $\langle\Gamma\rangle_{\not\equiv,w}$ for set of all relations qfwpp-definable in Γ. It is not hard to check that if $\Gamma' \subseteq \langle\Gamma\rangle_{\not\equiv,w}$, then every instance is mapped to an instance of equally many variables — hence W-MAX-ONES(Γ') is CV-reducible to W-MAX-ONES(Γ) whenever Γ' is finite.

Theorem 10. *Let Γ be a constraint language such that* W-MAX-ONES(Γ) *is NP-hard. Then it holds that* W-MAX-ONES(R_{II_2}) $\leq^{\text{CV}}$ W-MAX-ONES(Γ).

Proof. We need to prove that $R_{\mathsf{II}_2} \in \langle R\rangle_{\not\equiv,w}$ for every $R \in \{R_{\mathsf{IS}_1^2}, R_{\mathsf{IN}_2}, R_{\mathsf{IL}_0}, R_{\mathsf{IL}_2}, R_{\mathsf{IL}_3}, R_{\mathsf{ID}_2}\}$. Let us first consider the case of $R_{\mathsf{II}_2} \in \langle R_{\mathsf{IN}_2}\rangle_{\not\equiv,w}$. First, note that

$$R_{\mathsf{II}_2} = \arg\max_{(x_1,\ldots,x_8)\in\mathbb{B}^8:(x_7,x_1,x_2,x_6,x_8,x_4,x_5,x_3)\in R_{\mathsf{IN}_2}} x_8. \qquad (\dagger)$$

We will see that it is not difficult to obtain a qfwpp-definition of R_{II_2} with R_{IN_2}, armed with this information. Let I be the W-MAX-ONES($\{R_{\mathsf{IN}_2}\}$) instance over the variables $\{x_1,\ldots,x_8\}$ with the single constraint $R_{\mathsf{IN}_2}(x_7,x_1,x_2,x_6,x_8,x_4,x_5,x_3)$, where x_8 have the weight 1 and all other variables the weight 0. It is then easy to see that I has exactly three optimal solutions ϕ_1, ϕ_2, ϕ_3, and that

$$((\phi_1(x_1),\ldots,\phi_1(x_8))) = (0,0,1,1,1,0,0,1) \in R_{\mathsf{II}_2},$$

$$((\phi_2(x_1),\ldots,\phi_2(x_8))) = (0,1,0,1,0,1,0,1) \in R_{\mathsf{II}_2},$$

and

$$((\phi_3(x_1),\ldots,\phi_3(x_8))) = (1,0,0,0,1,1,0,1) \in R_{\mathsf{II}_2},$$

which implies that $R_{\mathsf{II}_2} \in \langle R\rangle_{\not\equiv,w}$. Hence, it is rather easy, albeit tedious, to prove that a relation is qfwpp-definable over another relation when given an equality akin to (†). To avoid needless repetition, we have prepared these arg max definitions in Table 3, from which it is easy to derive the necessary qfwpp-definitions. □

3.4. The VCSP *Problem*

Let us first remark that the VCSP$(\cdot)$ problem does not follow the standard Galois connection in Theorem 1. For a concrete counter example, recall the two relations $\mathrm{F} = \{(0)\}$ and $\mathrm{T} = \{(1)\}$, and consider the problem MAX-CSP$(\{\mathrm{F},\mathrm{T}\})$ (crucially, remember that every MAX-CSP problem can be seen as a special case of a VCSP problem). On the one hand, it is then known that MAX-CSP$(\{\mathrm{F},\mathrm{T}\})$ is tractable, but on the other hand, the pp-definable relation $R^{01}(x_1,x_2) \equiv \mathrm{F}(x_1) \wedge \mathrm{T}(x_2)$ results in an NP-hard problem (see, e.g., Theorem 2.11 in Khanna et al. [32]).

Thus, the weak base method is not applicable, and alternative methods are required. For this purpose we use *multimorphisms* from Cohen et al. [22]. Let Δ be a set of cost functions on $\mathbb{B}$, let p be a unary operation on $\mathbb{B}$, and let f,g be binary operations on $\mathbb{B}$. We say that Δ admits the binary *multimorphism* (f,g) if it holds that $\nu(f(x,y)) + \nu(g(x,y)) \leq \nu(x) + \nu(y)$ for every $\nu \in \Delta$ and $x,y \in \mathbb{B}^{\mathrm{ar}(\nu)}$. Similarly Δ admits the unary *multimorphism* (p) if it holds that $\nu(p(x)) \leq \nu(x)$ for every $\nu \in \Delta$ and $x \in \mathbb{B}^{\mathrm{ar}(\nu)}$. We have the following result.

Theorem 11 ([22])**.** *Let Δ be a set of finite-valued cost functions on $\mathbb{B}$. If Δ admits the unary (0)-multimorphism, the unary (1)-multimorphism or the binary $(\min,\max)$-multimorphism, then* VCSP(Δ) *is in PO. Otherwise* VCSP(Δ) *is NP-hard.*

Recall that the function $f_{\neq}$ equals $\{(0,0) \mapsto 1, (0,1) \mapsto 0, (1,0) \mapsto 0, (1,1) \mapsto 1\}$ and that the minimisation problem VCSP$(f_{\neq})$ and the maximisation problem MAX CUT are trivially CV-reducible to each other. We will make use of (a variant of) the concept of *expressibility* [22]. We say that a cost function g is $\nexists$*-expressible* in Δ if

$$g(x_1,\ldots,x_n) = \sum_{i=1}^{m} w_i f_i(\mathbf{s}_i) + w$$

for some tuples $\mathbf{s}_i$ over $\{x_1,\ldots,x_n\}$, weights $w_i \in \mathbb{Q}_{\geq 0}$, $w \in \mathbb{Q}$ and cost functions $f_i \in \Delta$. It is not hard to see that if every function in a finite set Δ' is $\nexists$-expressible in Δ, then VCSP$(\Delta') \leq^{\mathrm{CV}}$ VCSP(Δ). Note that if a unary cost function f_c is expressible in Δ with $\arg\min_{x\in\mathbb{B}} f_c(x) = \{c\}$, then we can force a variable to take the constant value c. If we can force variables to the constants 0 and 1, then we can allow tuples $\mathbf{s}_i$ over $\{x_1,\ldots,x_n,0,1\}$ in the $\nexists$-expression, and still obtain a CV-reduction.

Theorem 12. *Let Δ be a set of finite-valued cost functions on $\mathbb{B}$. If the problem* VCSP(Δ) *is NP-hard, then* VCSP$(f_{\neq}) \leq^{\mathrm{CV}}$ VCSP(Δ).

Proof. The proof consists of two parts. Using the assumption that VCSP(Δ) is NP-hard we will (1) prove that we can $\nexists$-express $f_{\neq}$ or force two variables v_0 and v_1 to 0 and 1, respectively, and (2) (in case we did not already $\nexists$-express $f_{\neq}$ in previous step) show that there exists a certain function which is $\nexists$-expressible over Δ, which together with the two constant variables v_0 and v_1 can $\nexists$-express $f_{\neq}$.

Step 1. VCSP(Δ) is NP-hard, so by Theorem 11 we know that Δ does not admit the unary (0)-multimorphism or the unary (1)-multimorphism, unless P=NP. Therefore there are $g,h \in \Delta$ and $\mathbf{u} = (u_1,\ldots,u_k)$, $\mathbf{v} = (v_1,\ldots,v_\ell)$ such that $g(\mathbf{0}) > g(\mathbf{u})$ and $h(\mathbf{1}) > h(\mathbf{v})$. We may assume $g(\mathbf{u}) = \min_{\mathbf{x}} g(\mathbf{x})$ and $h(\mathbf{v}) = \min_{\mathbf{x}} h(\mathbf{x})$.

Define $f_{01}(x,y) = g(z_1,\dots,z_k) + h(w_1,\dots,w_\ell)$ where

$$z_i = \begin{cases} x & \text{if } u_i = 0, \\ y & \text{otherwise,} \end{cases} \quad \text{and} \quad w_i = \begin{cases} x & \text{if } v_i = 0, \\ y & \text{otherwise.} \end{cases}$$

Note that

$$\begin{aligned} &f_{01}(0,1) = g(\mathbf{u}) + h(\mathbf{v}), \\ &f_{01}(0,0) = g(\mathbf{0}) + h(\mathbf{0}) > g(\mathbf{u}) + h(\mathbf{0}) \geq g(\mathbf{u}) + h(\mathbf{v}) = f_{01}(0,1), \\ &f_{01}(1,1) = g(\mathbf{1}) + h(\mathbf{1}) > g(\mathbf{1}) + h(\mathbf{v}) \geq g(\mathbf{u}) + h(\mathbf{v}) = f_{01}(0,1), \\ &f_{01}(1,0) = g(\mathbf{u}') + h(\mathbf{v}') \geq g(\mathbf{u}) + h(\mathbf{v}) = f_{01}(0,1) \quad \text{(for some } \mathbf{u}', \mathbf{v}'). \end{aligned}$$

If $f_{01}(0,1) = f_{01}(1,0)$, then $f_{\neq}(x,y) = \alpha_1(f_{01}(x,y) + f_{01}(y,x)) + \alpha_2$ for suitable $\alpha_1 \in \mathbb{Q}_{\geq 0}$ and $\alpha_2 \in \mathbb{Q}$. This is an $\nexists$-expression, and we are done. Otherwise $f_{01}(0,1) < f_{01}(1,0)$, which means we can force variables v_0 and v_1 to 0 and 1, respectively, with the term $f_{01}(v_0,v_1)$ (given sufficiently high weight).

Step 2. Assume now that $f_{\neq}$ was not $\nexists$-expressed in the previous step and recall that we have access to variables v_0 and v_1 that are forced to take values 0 and 1, respectively. We know that Δ does not admit the $(\min,\max)$-multimorphism (by Theorem 11) since VCSP(Δ) is NP-hard by assumption. Hence, there exists a k-ary function $f \in \Delta$ and $\mathbf{s} = (s_1,\dots,s_k)$, $\mathbf{t} = (t_1,\dots,t_k)$ such that

$$f(\min(\mathbf{s},\mathbf{t})) + f(\max(\mathbf{s},\mathbf{t})) > f(\mathbf{s}) + f(\mathbf{t}).$$

Define $g(x,y) = f(z_1,\dots,z_k)$ where

$$z_i = \begin{cases} v_1 & \text{if } \min(s_i,t_i) = 1, \\ v_0 & \text{if } \max(s_i,t_i) = 0, \\ x & \text{otherwise if } s_i > t_i, \text{ and} \\ y & \text{otherwise,} \end{cases}$$

for $1 \leq i \leq k$. Note the following:

$$\begin{aligned} g(0,0) &= f(\min(\mathbf{s},\mathbf{t})), \\ g(1,1) &= f(\max(\mathbf{s},\mathbf{t})), \\ g(1,0) &= f(\mathbf{s}), \\ g(0,1) &= f(\mathbf{t}). \end{aligned}$$

Set $h(x,y) = g(x,y) + g(y,x)$. Now $h(0,1) = h(1,0)$ and

$$\begin{aligned} h(0,1) = g(0,1) + g(1,0) &= f(\mathbf{s}) + f(\mathbf{t}) \\ &< f(\min(\mathbf{s},\mathbf{t})) + f(\max(\mathbf{s},\mathbf{t})) = \frac{1}{2}(h(0,0) + h(1,1)). \end{aligned} \quad (3)$$

If $h(0,0) = h(1,1)$, then $f_{\neq} = \alpha_1 h + \alpha_2$ for some $\alpha_1 \in \mathbb{Q}_{\geq 0}$ and $\alpha_2 \in \mathbb{Q}$.

Otherwise, assume $h(1,1) > h(0,0)$. We can do this without loss of generality since the other case is symmetric. Furthermore, we can without loss of generality (by scaling and translating the function h) assume that $h(1,1) - h(0,0) = 2$. Let $f_1(x) = \alpha_1 f_{01}(v_0, x) + \alpha_2$ for some $\alpha_1 \in \mathbb{Q}_{\geq 0}$ and $\alpha_2 \in \mathbb{Q}$ such that $f_1(1) = 0$ and $f_1(0) = 1$. Then define $h'(x,y) = f_1(x) + f_1(y) + h(x,y)$. This function satisfies $h'(0,0) = h'(1,1)$ and $h'(0,1) = h'(1,0)$. Moreover, we have

$$h'(0,0) = \frac{1}{2}(h'(0,0) + h'(1,1)) = \frac{1}{2}(2 + h(0,0) + h(1,1)) > 1 + h(0,1) = h'(0,1),$$

where the inequality follows from (3). Hence, we can $\nexists$-express $f_{\neq}$ as $\alpha_1 h' + \alpha_2$ for some $\alpha_1 \in \mathbb{Q}_{\geq 0}$ and $\alpha_2 \in \mathbb{Q}$. □

3.5. The Broader Picture

The results in Section 3 do not describe the *relative* complexity between $\mathrm{SAT}(\cdot)$, $\mathrm{SSAT}(\cdot)$, $\mathrm{ABD}(\cdot)$, MAX-ONES$(\cdot)$ and $\mathrm{VCSP}(\cdot)$. We have thus far not been able to precisely pinpoint the complexity of $\mathrm{ABD}(\cdot)$ to the other problems, but we readily see that

1. $\mathrm{SSAT}(R_{\mathrm{II}_2}) =^{\mathrm{CV}} \mathrm{SAT}(R_{\mathrm{II}_2}) \leq^{\mathrm{CV}}$ W-MAX-ONES(R_{II_2}), and
2. W-MAX-ONES$(R_{\mathrm{II}_2}) \leq^{\mathrm{CV}}$ W-MAX-ONES$(\mathrm{NAND}^2) =^{\mathrm{CV}}$ W-MAX INDEPENDENT SET $\leq^{\mathrm{CV}} \mathrm{VCSP}(\{f_{\mathrm{nand}}, f_0, f_1\})$,

where f_0 and f_1 are the unary cost functions $\{0 \mapsto 0, 1 \mapsto 1\}$ and $\{0 \mapsto 1, 1 \mapsto 0\}$, respectively, and f_{nand} denotes the the binary cost function $\{(0,0) \mapsto 0, (0,1) \mapsto 0, (1,0) \mapsto 0, (1,1) \mapsto 1\}$.

We have the following relation.

Lemma 5. $\mathrm{VCSP}(\{f_{\mathrm{nand}}, f_0, f_1\}) \leq^{\mathrm{CV}} \mathrm{VCSP}(f_{\neq})$.

Proof. Note that f_{nand} is $\nexists$-expressible in $\{f_{\neq}, f_0, f_1\}$ since

$$f_{\mathrm{nand}}(x,y) = \frac{1}{2}(f_{\neq}(x,y) + f_0(x) + f_0(y) - 1).$$

Hence (see Section 3.4), $\mathrm{VCSP}(\{f_{\mathrm{nand}}, f_0, f_1\}) \leq^{\mathrm{CV}} \mathrm{VCSP}(\{f_{\neq}, f_0, f_1\})$.

To complete the proof, we present a CV-reduction from $\mathrm{VCSP}(\{f_{\neq}, f_0, f_1\})$ to $\mathrm{VCSP}(\{f_{\neq}\})$. Let v_0 and v_1 be two fresh variables. We make sure v_0 and v_1 are not mapped to the same value with the term $f_{\neq}(v_0, v_1)$ (with a sufficiently high weight). Let $\sigma = \{0 \mapsto 1, 1 \mapsto 0\}$ and note that $f_{\neq}$ is invariant under σ in the sense that $f_{\neq} = f_{\neq} \circ \sigma$. Hence, if I is an instance of $\mathrm{VCSP}(\{f_{\neq}\})$ and ϕ is a solution to I of cost C, then $\sigma \circ \phi$ is another solution to I of cost C. We can therefore, without loss of generality, assume that every solution maps v_0 to 0 and v_1 to 1. Now, every term $f_0(x)$ can be replaced with $f_{\neq}(x, v_1)$, and every term $f_1(x)$ with $f_{\neq}(x, v_0)$. □

Combining Lemma 5 and Theorem 12, we get $\mathrm{VCSP}(\{f_{\mathrm{nand}}, f_0, f_1\}) =^{\mathrm{CV}} \mathrm{VCSP}(f_{\neq})$. Thus, there is no unique easiest NP-hard set of finite-valued cost functions for VCSP. The complexity results are summarized in Figure 7. Some trivial inclusions are omitted in the figure, for example it holds that $\mathrm{SAT}(\Gamma) \leq^{\mathrm{CV}}$ W-MAX-ONES(Γ) for all Γ.

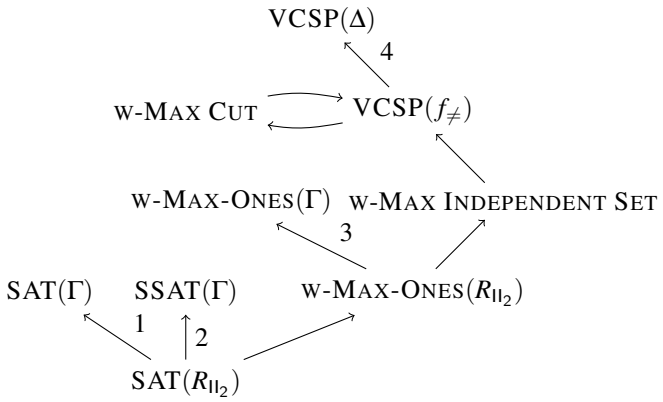

Figure 7: The complexity landscape of some Boolean optimization and satisfiability problems. A directed arrow from one node A to B means that $A \leq^{\mathrm{CV}} B$. Relation 1 holds when $\mathrm{SAT}(\Gamma)$ is NP-hard, relation 2 holds when $\mathrm{SSAT}(\Gamma)$ is NP-hard, relation 3 holds when W-MAX-ONES(Γ) is NP-hard, and relation 4 holds for every finite-valued Δ such that $\mathrm{VCSP}(\Delta)$ is NP-hard.

4. Subexponential Time and the Exponential-Time Hypothesis

Recall that the ETH states that 3-SAT $\notin$ SE [9]. We remind the reader that the ETH can be based on different size parameters (such as the number of variables or the number of clauses) and that these different definitions often coincide [11]. In this section we investigate the consequences of the ETH for the SSAT, ABD, U-MAX-ONES, and U-VCSP problems. A direct consequence of Section 3 is that if there exists any finite constraint language Γ or set of cost functions Δ such that W-MAX-ONES(Γ), $\mathrm{SSAT}(\Gamma)$ or $\mathrm{VCSP}(\Delta)$ is NP-hard and in SE, then $\mathrm{SAT}(R_{\mathrm{II}_2})$ is in SE which implies that the ETH is false [4]. The other direction is interesting too since it highlights the likelihood of subexponential time algorithms for the problems, relative to the ETH. We investigate these questions for $\mathrm{SSAT}(\cdot)$, $\mathrm{ABD}(\cdot)$, U-MAX-ONES$(\cdot)$, and $\mathrm{VCSP}(\cdot)$, in Section 4.1–4.4. In Section 4.5 we summarize our results and bring them together with the help of the ETH.

4.1. Lower Bounds for Surjective Satisfiability

Recall from Figure 7 that if there exists an NP-hard $\mathrm{SSAT}(\Gamma)$ problem in SE then $\mathrm{SSAT}(R_{\mathrm{II}_2}) =^{\mathrm{CV}} \mathrm{SAT}(R_{\mathrm{II}_2}) \in$ SE which contradicts the ETH [4]. Hence, all that remains to show is that every $\mathrm{SSAT}(\Gamma)$ problem is in SE if the ETH is false, which we accomplish with the following lemma.

Lemma 6. *If the ETH is false then for every finite constraint language* Γ *the problem* $\mathrm{SSAT}(\Gamma)$ *is in* SE.

Proof. If the ETH does not hold then $\mathrm{SAT}(\Gamma \cup \{\mathrm{F},\mathrm{T}\})$ is in SE since Γ is finite [4]. We present an SE algorithm for $\mathrm{SSAT}(\Gamma)$ based on an SE algorithm for $\mathrm{SAT}(\Gamma \cup \{\mathrm{F},\mathrm{T}\})$.

Arbitrarily choose $\varepsilon > 0$ and let A be an algorithm for $\mathrm{SAT}(\Gamma \cup \{\mathrm{F},\mathrm{T}\})$ that runs in $2^{\varepsilon \cdot n}$ time. Given an instance (V,C) of $\mathrm{SSAT}(\Gamma)$, do the following:

1. *ans* := 'no'
2. for every pair of distinct variables $v, w \in V$, let *ans* := 'yes' if $A((V, C \cup \{\mathrm{F}(v), \mathrm{T}(w)\}))$ or $A((V, C \cup \{\mathrm{F}(w), \mathrm{T}(v)\}))$
3. return *ans*

This algorithm answers 'yes' if and only if (V,C) has a surjective solution. Furthermore, it runs in $poly(||(V,C)||) \cdot 2^{\varepsilon \cdot n}$ time, so $\mathrm{SSAT}(\Gamma)$ is in SE. □

4.2. Lower Bounds for Propositional Abduction

From Theorem 7 we see that if there exists an NP-complete $\mathrm{ABD}(\Gamma)$ problem in SE then $\mathrm{ABD}(R_{\mathsf{IE}_2}) \in \mathrm{SE}$, too. We now show that this problem cannot be solvable in subexponential time without violating the ETH, which implies that no NP-complete $\mathrm{ABD}(\Gamma)$ problem is solvable in subexponential time (under the ETH).

Lemma 7. *If* $\mathrm{ABD}(\Gamma) \in \mathrm{SE}$ *for some finite constraint language* Γ *such that* $\mathrm{ABD}(\Gamma)$ *is NP-complete then the ETH is false.*

Proof. First, assume $\mathrm{ABD}(\Gamma) \in \mathrm{SE}$ and that $\mathrm{ABD}(\Gamma)$ is NP-complete. In particular this implies that the problem $\mathrm{ABD}(\{R_{\mathsf{IE}_2}\}) \in \mathrm{SE}$ via Theorem 7. From Jonsson et al. [4] there exists a $B > 0$ such that 3-SAT-$B \in \mathrm{SE}$ if and only if 3-$\mathrm{SAT} \in \mathrm{SE}$. Hence, to prove the claim we will give an LV-reduction from 3-SAT-B to $\mathrm{ABD}(\{R_{\mathsf{IE}_2}\})$.

In Creignou and Zanuttini [33] it is proven that there exists a reduction which, given a SAT formula φ over n variables and m constraints, produces an instance (φ', L, q) of $\mathrm{ABD}(\mathsf{IE}_2)$ over $3n$ variables where φ' contains $O(m+n)$ binary constraints, and a single constraint of arity $f(n) \in O(n+m)$ of the form $(\ell_1 \vee \ell_2 \ldots \vee \ell_{f(n)})$, where each ℓ_i is either a positive or negative atom of the form x or $\neg x$. Since m is linearly bounded with respect to n it therefore follows that φ' contains $O(n)$ binary constraints and a single constraint of arity $O(n)$, of the aforementioned form. Using standard techniques (see, e.g., Section 5.2 in Creignou and Zanuttini [33]) one can then prove that $(\ell_1 \vee \ell_2 \ldots \vee \ell_{f(n)})$ can be pp-defined by R_{IE_2} requiring only $O(n)$ variables and constraints. Similarly, each binary constraint in φ' can be pp-defined by R_{IE_2} with a constant number of fresh variables, and if we replace each constraint in φ' in this manner we obtain an instance of $\mathrm{ABD}(R_{\mathsf{IE}_2})$ with a linear amount of fresh variables and constraints. □

Hence, NP-complete $\mathrm{ABD}(\Gamma)$ problems are unlikely to be solvable in subexponential time, since this would contradict the ETH. However, we have been unable to strengthen the other direction, showing that every NP-complete $\mathrm{ABD}(\Gamma)$ problem is in SE if the ETH is false, and it is currently unclear whether such a result is feasible. We discuss this in greater detail in Section 5.

4.3. Lower Bounds for Max-Ones

We now turn to the unweighted U-MAX-ONES$(\cdot)$ problem, where we obtain a complete understanding of subexponential complexity with respect to the ETH.

Lemma 8. *If* U-MAX-ONES(Γ) *is in* SE *for some finite constraint language* Γ *such that* U-MAX-ONES(Γ) *is NP-hard, then the ETH is false.*

Proof. From Jonsson et al. [4] it follows that 3-SAT $\in$ SE if and only if SAT(R_{II_2})-2 $\in$ SE. Combining this with Theorem 9 we only have to prove that SAT(R_{II_2})-2 LV-reduces to U-MAX-ONES(R) for $R \in \{R_{\mathsf{IS}_1^2}, R_{\mathsf{IN}_2}, R_{\mathsf{IL}_0}, R_{\mathsf{IL}_2}, R_{\mathsf{IL}_3}, R_{\mathsf{ID}_2}\}$. We provide an illustrative reduction from SAT(R_{II_2})-2 to U-MAX-ONES($R_{\mathsf{IS}_1^2}$); the remaining reductions are presented in Lemmas 10–14 in the end of this section. Since $R_{\mathsf{IS}_1^2}$ is the NAND relation with one additional constant column, the U-MAX-ONES($R_{\mathsf{IS}_1^2}$) problem is basically the maximum independent set problem or, equivalently, the maximum clique problem in the complement graph. Given an instance I of CSP(R_{II_2})-2 we create for every constraint 3 vertices, one corresponding to each feasible assignment of values to the variables occurring in the constraint. We add edges between all pairs of vertices that are not inconsistent and that do not correspond to the same constraint. The instance I is satisfied if and only if there is a clique of size m where m is the number of constraints in I. Since $m \leq 2n$ this implies that the number of vertices is $\leq 6n$. □

Hence, we have ruled out the possibility that, assuming the ETH, there could exist a U-MAX-ONES(Γ) $\in$ SE problem which is NP-complete. It is also not difficult to prove the opposite, i.e., that if the ETH is false, then U-MAX-ONES(Γ) $\in$ SE for every Boolean constraint language Γ.

Lemma 9. *If the ETH is false, then* U-MAX-ONES(Γ) $\in$ SE *for every Boolean constraint language* Γ.

Proof. Define SNP to be the class of properties expressible by formulas of the type $\exists S_1 \ldots \exists S_n \forall x_1 \ldots \forall x_m.F$ where F is a quantifier-free logical formula, $\exists S_1 \ldots \exists S_n$ are second order existential quantifiers, and $\forall x_1 \ldots \forall x_m$ are first-order universal quantifiers. Monadic SNP (MSNP) is the restriction of SNP where all second-order predicates are required to be unary [34]. The associated search problem tries to identify instantiations of $S_1, \ldots, S_n$ that make the resulting first-order formula true. We will be interested in properties that can be expressed by formulas that additionally contain *size-constrained* existential quantifiers. A size-constrained existential quantifier is of the form $\exists S, |S| \bowtie s$, where $|S|$ is the number of inputs where relation S holds, and $\bowtie \in \{=, \leq, \geq\}$. Define size-constrained SNP as the class of properties of relations and numbers that are expressible by formulas $\exists S_1 \ldots \exists S_n \forall x_1 \ldots \forall x_m.F$ where the existential quantifiers are allowed to be size-constrained.

If the ETH is false then 3-SAT is solvable in subexponential time. By Impagliazzo et al. [11] this problem is size-constrained MSNP-complete under size-preserving SERF reductions. Hence, we only have to prove that U-MAX-ONES($\cdot$) is included in size-constrained MSNP for it to be solvable in subexponential time. Impagliazzo et al. [11] shows that k-SAT is in SNP by providing an explicit formula $\exists S.F$ where F is a universal formula and S a unary predicate interpreted such that $x \in S$ if and only if x is true. Let k be the highest arity of any relation in Γ. Since k-SAT can qfpp implement any k-ary Boolean relation it is therefore sufficient to prove that U-MAX-ONES(Γ^k_{SAT}) is in size-constrained MSNP. This is easy to do with the formula

$$\exists S, |S| \geq K.F$$

where K is the parameter corresponding to the number of variables that have to be assigned 1. □

Additional reductions for U-MAX-ONES$(\cdot)$

In this section we provide the reductions for U-MAX-ONES$(\cdot)$ which were missing in Lemma 8.

Lemma 10. SAT(R_{II_2})*-2 LV-reduces to* U-MAX-ONES(R_{IL_2}).

Proof. We reduce an instance I of SAT(R_{II_2})-2 on n variables and m constraints to an instance of U-MAX-ONES(R_{IL_2}) containing at most $2+8n$ variables. Let v_0, v_1 be two fresh global variables constrained as $R_{\mathsf{IL}_2}(v_0,v_0,v_0,v_1,v_1,v_1,v_0,v_1)$. Note that this forces v_0 to 0 and v_1 to 1 in any satisfying assignment. Now, for every variable x in the SAT-instance we create an additional variable x' which we constrain as $R_{\mathsf{IL}_2}(x',x,v_1,x,x',v_0,v_0,v_1)$. This correctly implements $\mathrm{Neq}(x,x')$. For the i-th constraint, $R_{\mathsf{II}_2}(x_1,\ldots,x_8)$, in I we (1) create three variables z_i^1, z_i^2, z_i^3, and (2) add the constraints

$$R_{\mathsf{IL}_2}(z_i^1,z_i^2,z_i^3,x_1,x_2,x_3,x_7,x_8) \wedge R_{\mathsf{IL}_2}(x_4,x_5,x_6,x_1,x_2,x_3,x_7,x_8).$$

Note that this correctly defines R_{II_2} if z_i^1, z_i^2, z_i^3 are not all assigned 0. Since every variable in the SAT-instance I can occur in at most two constraints we have that $m \leq 2n$. Hence, the resulting U-MAX-ONES instance contains at most $2+2n+3\cdot 2n = 2+8n$ variables. Now, importantly, since x and x', and v_0 and v_1, must take different values it holds that the measure of a solution of this new instance is exactly the number of variables z_i^j that are mapped to 1. Thus, the crucial observation is that one cannot have an optimal solution with objective value $\geq n+2m$ unless one for each block of auxiliary variables z_i^1, z_i^2, z_i^3 assign at least one of them a non-zero value. Hence, for an optimal solution the objective value is $\geq n+2m$ if and only if I is satisfiable. □

Lemma 11. U-MAX-ONES(R_{IL_2}) *LV-reduces to* U-MAX-ONES(R_{IL_0}).

Proof. We reduce an instance I of U-MAX-ONES(R_{IL_2}) on n variables to an instance of U-MAX-ONES(R_{IL_0}) on $2+2n$ variables. Let $v_0, v_1, y_1, \ldots, y_n$ be fresh variables and constrain them as $R_{\mathsf{IL}_0}(v_0,v_0,v_0,v_0) \wedge R_{\mathsf{IL}_0}(v_1,v_0,y_1,v_0) \wedge \ldots \wedge R_{\mathsf{IL}_0}(v_1,v_0,y_n,v_0)$. Note that this forces v_0 to 0, and that if v_1 is mapped to 0, then so are the variables $y_1,\ldots,y_n$. If v_1 is mapped to 1 on the other hand, then $y_1,\ldots,y_n$ can be mapped to 1. For every constraint $R_{\mathsf{IL}_2}(x_1,\ldots,x_8)$ we create the constraints

$$\begin{aligned} R_{\mathsf{IL}_0}(x_1,x_2,x_3,v_0) \wedge R_{\mathsf{IL}_0}(v_1,x_1,x_4,v_0) \wedge R_{\mathsf{IL}_0}(v_1,x_2,x_5,v_0) \\ \wedge R_{\mathsf{IL}_0}(v_1,x_3,x_6,v_0) \wedge R_{\mathsf{IL}_0}(v_1,x_7,x_8,v_0) \wedge R_{\mathsf{IL}_0}(x_7,x_7,x_7,x_7). \end{aligned}$$

The resulting U-MAX-ONES(R_{IL_0}) instance has $2+2n$ variables and has a solution with measure $n+1+k$ if and only if I has a solution with measure k. □

Lemma 12. U-MAX-ONES(R_{II_2}) *LV-reduces to* U-MAX-ONES(R_{IN_2}).

Proof. We reduce an instance I of U-MAX-ONES(R_{II_2}) over n variables to an instance of U-MAX-ONES(R_{IN_2}) over $3+2n$ variables. Create two fresh variables v_0, v_1 and constrain them as $R_{\mathsf{IN}_2}(v_0,v_0,v_0,v_0,v_1,v_1,v_1,v_1)$ in order to force v_0 and v_1 to be mapped to different values. We then create the $n+1$ variables $y_1,\ldots,y_{n+1}$ and constrain them as

$\bigwedge_{i=1}^{n+1} R_{\mathsf{IN}_2}(v_0,v_0,v_0,v_0,y_i,y_i,y_i,y_i)$. This forces all of the variables y_i to be mapped to the same value as v_1. We can now express $R_{\mathsf{II}_2}(x_1,\ldots,x_8)$ using the implementation

$$R_{\mathsf{IN}_2}(v_0,x_1,x_2,x_6,v_1,x_4,x_5,x_3) \wedge R_{\mathsf{IN}_2}(v_0,x_7,x_7,v_0,v_1,x_8,x_8,v_1).$$

Note that in any optimal solution of the new instance v_1 will be mapped to 1 which means that the implementation of R_{II_2} given above will be correct. The resulting instance has a solution with measure $2+n+k$ if and only if I has a solution with measure k. □

Lemma 13. U-MAX-ONES($R_{\mathsf{IS}_1^2}$) *LV-reduces to* U-MAX-ONES(R_{ID_2}).

Proof. We reduce an instance of U-MAX-ONES($R_{\mathsf{IS}_1^2}$) on n variables to an instance of U-MAX-ONES(R_{ID_2}) on $2+3n$ variables. Create two new variables v_0 and v_1 and constrain them as $R_{\mathsf{ID}_2}(v_1,v_1,v_0,v_0,v_0,v_1)$. Note that this forces v_0 to 0 and v_1 to 1. For every variable x we introduce two extra variables x' and x'' and constrain them as $R_{\mathsf{ID}_2}(x,x',x',x,v_0,v_1) \wedge R_{\mathsf{ID}_2}(x',x'',x'',x',v_0,v_1)$. Note that this implements the constraints $\mathrm{Neq}(x,x')$ and $\mathrm{Neq}(x',x'')$, and that no matter what x is mapped to exactly one of x' and x'' is mapped to 1. For every constraint $R_{\mathsf{IS}_1^2}(x,y,z)$ we then introduce the constraint $R_{\mathsf{ID}_2}(x',y',x,y,z,v_1)$. The resulting instance has a solution with measure $1+n+k$ if and only if I has a solution with measure k. □

Lemma 14. U-MAX-ONES(R_{IL_2}) *LV-reduces to* U-MAX-ONES(R_{IL_3}).

Proof. We reduce an instance of U-MAX-ONES(R_{IL_2}) on n variables to an instance of U-MAX-ONES(R_{IL_3}) on $2+2n+1$ variables. Create two new variables v_0 and v_1 and constrain them as $R_{\mathsf{IL}_3}(v_0,v_0,v_0,v_0,v_1,v_1,v_1,v_1)$. Note that this forces v_0 and v_1 to be mapped to different values. We then introduce fresh variables $y_1,\ldots,y_{n+1}$ and constrain them as $\bigwedge_{i=1}^{2n} R_{\mathsf{IL}_3}(v_0,v_0,v_0,v_0,y_i,y_i,y_i,y_i)$. This will ensure that every variables y_i is mapped to the same value as v_1 and therefore that in every optimal solution v_0 is mapped to 0 and v_1 is mapped to 1. For every constraint $R_{\mathsf{IL}_2}(x_1,\ldots,x_8)$ we introduce the constraints

$$R_{\mathsf{IL}_3}(x_7,x_1,x_2,x_3,x_8,x_4,x_5,x_6) \wedge R_{\mathsf{IL}_3}(x_7,x_7,x_7,x_7,v_1,v_1,v_1,v_1) \\ \wedge R_{\mathsf{IL}_3}(v_0,v_0,v_0,v_0,x_8,x_8,x_8,x_8).$$

The resulting instance has a solution with measure $2+n+k$ if and only if I has a solution with measure k. □

4.4. Lower Bounds for VCSP

For VCSP we have already established that there cannot exist any NP-hard VCSP(Δ) problem in SE (under the ETH). The other direction appears significantly harder, if true, but we do manage to prove a partial converse for the unweighted VCSP problem with at most $d|\mathrm{Var}(I)|$ constraints for some fixed $d \geq 0$ (U-VCSP$_d(\Delta)$). We state the following results using U-MAX-ONES(Γ) as a starting point, rather than the ETH, since it simplifies the proof of the forthcoming Theorem 13.

Lemma 15. *If* U-MAX-ONES(Γ) $\in$ SE *for every finite Boolean constraint language* Γ *then* U-VCSP$_d(\Delta) \in$ SE *for every finite set of Boolean cost functions* Δ *and* $d \geq 0$.

Proof. We first show that if every U-MAX-ONES$(\Gamma) \in$ SE, then U-MIN-ONES$(\Gamma) \in$ SE for all Γ, too. Here U-MIN-ONES(Γ) denotes the minimisation variant of U-MAX-ONES(Γ) where the goal instead is to minimise the number of variables assigned 1. Arbitrarily choose a finite constraint language Γ over $\mathbb{B}$. We present an LV-reduction from U-MIN-ONES(Γ) to U-MAX-ONES$(\Gamma \cup \{\text{Neq}\})$. Let $(\{v_1,\ldots,v_n\},C)$ be an arbitrary instance of U-MIN-ONES(Γ) with optimal value K. Consider the instance $I = (V',C')$ of U-MAX-ONES$(\Gamma \cup \{\text{Neq}\})$ where:

$$V' = \{v_1, v_1', v_1'', \ldots, v_n, v_n', v_n''\}, \text{ and}$$
$$C' = C \cup \{\text{Neq}(v_1, v_1'), \text{Neq}(v_1, v_1''), \ldots, \text{Neq}(v_n, v_n'), \text{Neq}(v_n, v_n'')\}.$$

For each variable $v_i \in \{v_1,\ldots,v_n\}$ that is assigned 0, the corresponding variables v_i', v_i'' are assigned 1, and vice-versa. It follows that the optimal value of I' is $2n-K$. Hence, U-MIN-ONES$(\Gamma) \in$ SE since U-MAX-ONES$(\Gamma \cup \{\text{Neq}\}) \in$ SE.

Now, arbitrarily choose $d \geq 0$ and a finite set of Boolean cost functions Δ. Since Δ is finite, we may without loss of generality assume that each function $f \in \Delta$ has its range in $\{0,1,2,\ldots\}$.

We show that U-VCSP$_d(\Delta) \in$ SE by exhibiting an LV-reduction from U-VCSP$_d(\Delta)$ to U-MIN-ONES(Γ) where Γ is finite and only depends on Δ. Given a tuple $\mathbf{a} = (a_1,\ldots,a_k) \in \mathbb{B}^k$, let $\text{val}(\mathbf{a}) = 1 + \sum_{j:a_j=1} 2^{j-1}$. Adding 1 to the sum ensures a non-zero value, which is necessary since the value of $\text{val}(\mathbf{a})$ corresponds to an index, which starts with 1. For each $f \in \Delta$ of arity k, define

$$R_f = \left\{ (x_1,\ldots,x_k,y_1,\ldots,y_{2^k}) \in \mathbb{B}^{k+2^k} \middle| \begin{array}{l} f(x_1,\ldots,x_k) > 0, \\ \{i : y_i \neq 0\} = \{\text{val}(x_1,\ldots,x_k)\} \end{array} \right\}$$
$$\cup \{(x_1,\ldots,x_k,0,\ldots,0) \in \mathbb{B}^{k+2^k} \mid f(x_1,\ldots,x_k) = 0\},$$

and let $\Gamma = \{\text{Eq},\text{Neq}\} \cup \{R_f \mid f \in \Delta\}$.

One may interpret R_f as follows: for each $(x_1,\ldots,x_k) \in \mathbb{B}^k$ the relation R_f contains exactly one tuple $(x_1,\ldots,x_k,y_1,\ldots,y_{2^k})$. If $f(x_1,\ldots,x_k) = 0$, then this is the tuple $(x_1,\ldots,x_k,0,\ldots,0)$. If $f(x_1,\ldots,x_k) > 0$, then this is the tuple $(x_1,\ldots,x_k,0,\ldots,1,\ldots,0)$ where the 1 is in position $k+\text{val}(x_1,\ldots,x_k)$. We show below how R_f can be used for 'translating' each $\mathbf{x} \in \mathbb{B}^k$ into its corresponding weight as prescribed by f.

Let $(V, \sum_{i=1}^m f_i(\mathbf{x}_i))$ be an arbitrary instance of U-VCSP$_d(\Delta)$ where $V = \{v_1,\ldots,v_n\}$. For each variable $v_i \in V$ we begin by introducing a fresh variable w_i. Now, assume the instance has an optimal solution with value K. For each term $f_i(v_1,\ldots,v_k)$ in the sum, do the following:

1. introduce 2^k fresh variables $v_1',\ldots,v_{2^k}'$,
2. for each $\mathbf{a} \in \mathbb{B}^k$ such that $f_i(\mathbf{a}) > 1$, introduce $n' = f_i(\mathbf{a})$ fresh variables $u_0,\ldots,u_{n'-1}$,
3. introduce the constraint $R_{f_i}(v_1,\ldots,v_k,v_1',\ldots,v_{2^k}')$,
4. introduce the constraints $\text{Neq}(v_1,w_1),\ldots,\text{Neq}(v_k,w_k)$, and
5. for each $\mathbf{a} \in \mathbb{B}^k$, let $n' = f_i(\mathbf{a})$ and do the following if $n' > 1$: let $p = \text{val}(\mathbf{a})$ and introduce the constraints $\text{Eq}(v_p',u_0), \text{Eq}(u_0,u_1),\ldots,\text{Eq}(u_{n'-2},u_{n'-1})$.

It is not difficult to realise that the resulting instance has optimal value $K+\sum_{i=1}^{m}\mathrm{ar}(f_i)$ given the interpretation of R_f and the following motivation of step 5: the Neq constraints introduced in step 5 ensure that the weight of $(x_1,\ldots,x_k)$ does not influence the weight of the construction and this explains that we need to adjust the optimal value with $\sum_{i=1}^{m}\mathrm{ar}(f_i)$.

Furthermore, the instance contains at most

$$2|V|+|C|\cdot(2^s+t\cdot(2^s+1))$$

variables where $s=\max\{\mathrm{ar}(f)\mid f\in\Delta\}$ and $t=\max\{f(\mathbf{a})\mid f\in\Delta \text{ and } \mathbf{a}\in\mathbb{B}^{\mathrm{ar}(f)}\}$. By noting that $|C|\leq d|V|$ and that s,t are constants that only depend on Δ, it follows that the reduction is an LV-reduction. □

4.5. Wrapping Up

We have seen several consequences of the ETH in the preceding sections. These results can more generally be related together as follows.

Theorem 13. *The following statements are equivalent.*

1. *The exponential-time hypothesis is false.*
2. $\mathrm{SSAT}(\Gamma)\in\mathrm{SE}$ *for every finite* Γ.
3. $\mathrm{SSAT}(\Gamma)\in\mathrm{SE}$ *for some finite* Γ *such that* $\mathrm{SSAT}(\Gamma)$ *is NP-hard.*
4. $\textsc{u-Max-Ones}(\Gamma)\in\mathrm{SE}$ *for every finite* Γ.
5. $\textsc{u-Max-Ones}(\Gamma)\in\mathrm{SE}$ *for some finite* Γ *such that* $\textsc{u-Max-Ones}(\Gamma)$ *is NP-hard.*
6. $\textsc{u-VCSP}(\Delta)_d\in\mathrm{SE}$ *for every finite set of finite-valued cost functions* Δ *and* $d\geq 0$.

Proof. The implication $1\Rightarrow 2$ follows from Lemma 6, and the implication $2\Rightarrow 3$ is trivial. For the implication $3\Rightarrow 4$, first observe from Figure 7 that if there exists an NP-hard $\mathrm{SAT}(\Gamma)$ problem in SE, then $\mathrm{SSAT}(R_{\mathsf{II}_2})\in\mathrm{SE}$, too, which contradicts the ETH [4]. An application of Lemma 9 then gives the implication $3\Rightarrow 4$.

Next, the implication $4\Rightarrow 5$ is trivial, and $5\Rightarrow 1$ follows by Lemma 8. The implication $4\Rightarrow 6$ follows from Lemma 15. We finish the proof by showing $6\Rightarrow 1$. Let $I=(V,C)$ be an instance of $\mathrm{SAT}(R_{\mathsf{II}_2})$-2. Note that I contains at most $2|V|$ constraints. Let f be the function defined by $f(\mathbf{x})=0$ if $\mathbf{x}\in R_{\mathsf{II}_2}$ and $f(\mathbf{x})=1$ otherwise. Create an instance of $\textsc{u-VCSP}_2(f)$ by, for every constraint $C_i=R_{\mathsf{II}_2}(x_1,\ldots,x_8)\in C$, adding to the cost function the term $f(x_1,\ldots,x_8)$. This instance has a solution with objective value 0 if and only if I is satisfiable. Hence, $\mathrm{SAT}(R_{\mathsf{II}_2})$-$2\in\mathrm{SE}$ which contradicts the ETH [4]. □

5. Future Research

We have studied the fine-grained complexity of several variants and extensions of the Boolean satisfiability problems. In many cases we were able to identify an 'easiest NP-hard problem' in each class, which we were able to use in order to relate the complexity of these problems to the ETH. Interestingly, for $\mathrm{SSAT}(\cdot)$ and $\textsc{u-Max-Ones}(\cdot)$ we were able to obtain a *complete* understanding of the possibility of obtaining subexponential

algorithms, under the ETH. While the results for $\mathrm{ABD}(\cdot)$ and $\mathrm{VCSP}(\cdot)$ are not quite as strong, it is worth mentioning that our results, to the best of our knowledge, are still the first of their kind for these problems. Let us now touch upon some directions for future research.

The abduction problem. While we were able to identify an 'easiest NP-complete $\mathrm{ABD}(\cdot)$ problem', several questions remain unanswered. First, it would be interesting to generalise the study to infinite constraint languages, where $\mathrm{ABD}(\cdot)$ is more interesting than the other problems under consideration due to the existence of NP-intermediate problems [31], i.e., there exists an *infinite* Γ_{NPI} such that $\mathrm{ABD}(\Gamma_{\mathrm{NPI}})$ is neither tractable, nor NP-complete, if P $\neq$ NP. Despite this, is it possible to use the algebraic approach to study fine-grained complexity of NP-intermediate problems? Here, the good news are that the algebraic aspect works equivalently well for infinite sets of relations, e.g., the co-clone $\langle\Gamma_{\mathrm{NPI}}\rangle$ admits a weak base [28]. The more challenging aspect is thus to apply the algebraic approach in such a way that it leads to interesting reductions. Is it, for example, possible to find an 'easiest NP-intermediate $\mathrm{ABD}(\cdot)$ problem' with respect to the co-clone $\langle\Gamma_{\mathrm{NPI}}\rangle$? Another interesting direction is to perform a more careful analysis of the easiest NP-complete problem $\mathrm{ABD}(R_{\mathsf{IE}_2})$: is it possible to relate it to the easiest $\mathrm{SAT}(\cdot)$ problem, or are the two problems fundamentally incomparable?

Weighted versus unweighted problems. Theorem 13 only applies to unweighted problems and lifting these results to the weighted case does not appear straightforward. We believe that some of these obstacles could be overcome with generalized sparsification techniques and provide an example proving that if any NP-hard W-MAX-ONES(Γ) problem is in SE, then MAX-CUT can be approximated within a multiplicative error of $(1\pm\varepsilon)$ (for any $\varepsilon>0$) in subexponential time. Assume that W-MAX-ONES$(\Gamma)\in$ SE is NP-hard, and arbitrarily choose $\varepsilon>0$. Let MAX-CUT$_c$ be the MAX-CUT problem restricted to graphs $G=(V,E)$ where $|E|\leq c\cdot|V|$. We first prove that MAX-CUT$_c$ is in SE for arbitrary $c\geq 0$. By Theorem 10, we infer that W-MAX-ONES$(R_{\mathsf{II}_2})\in$ SE. Given an instance (V,E) of MAX-CUT$_c$, one can introduce one fresh variable x_v for each $v\in V$ and one fresh variable x_e for each edge $e\in E$. For each edge $e=(v,w)$, we then constrain the variables x_v,x_w and x_e as $R(x_v,x_w,x_e)$ where $R=\{(0,0,0),(0,1,1),(1,0,1),(1,1,0)\}\in\langle R_{\mathsf{II}_2}\rangle$. It can then be verified that that the maximum value of $\sum_{e\in E}w_e h(x_e)$ for an optimal solution h (where w_e is the weight associated with the edge e) equals the weight of a maximum cut in (V,E). This is an LV-reduction since $|E|=c\cdot|V|$. Now consider an instance (V,E) of the unrestricted MAX-CUT problem. By Batson et al. [35], we can (in polynomial time) compute a *cut sparsifier* (V',E') with only $D_\varepsilon\cdot n/\varepsilon^2$ edges (where D_ε is a constant depending only on ε), which approximately preserves the value of the maximum cut of (V,E) to within a multiplicative error of $(1\pm\varepsilon)$. By using the LV-reduction above from MAX-CUT$_{D_\varepsilon/\varepsilon^2}$ to W-MAX-ONES(Γ), it follows that we can approximate the maximum cut of (V,E) within $(1\pm\varepsilon)$ in subexponential time.

Acknowledgements

The thank the anonymous reviewer for several helpful comments and suggestions which were used to improve the article. The first author is partially supported by the

Swedish Research Council (VR) under grant 2017-04112, and the second author by VR under grant 2019-03690.

References

[1] G. Woeginger, Exact algorithms for NP-hard problems: a survey, in: M. Juenger, G. Reinelt, G. Rinaldi (Eds.), Combinatorial Optimization – Eureka! You Shrink!, 2000, pp. 185–207.

[2] D. Lokshtanov, D. Marx, S. Saurabh, Lower bounds based on the exponential time hypothesis, Bulletin of the EATCS 105 (2011) 41–72.

[3] E. Grandjean, F. Olive, Graph properties checkable in linear time in the number of vertices, Journal of Computer and System Sciences 68 (3) (2004) 546–597.

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

[5] A. Bulatov, A dichotomy theorem for nonuniform CSPs, in: Proceedings of the 58th Annual Symposium on Foundations of Computer Science (FOCS-2017), IEEE Computer Society, 2017.

[6] D. Zhuk, A proof of the CSP dichotomy conjecture, Journal of the ACM 67 (5) (2020) 30:1–30:78.

[7] M. Couceiro, L. Haddad, V. Lagerkvist, A survey on the fine-grained complexity of constraint satisfaction problems based on partial polymorphisms, Journal of Multiple-Valued Logic and Soft Computing. To appear. (2020).

[8] N. Alon, D. Lokshtanov, S. Saurabh, Fast FAST, in: S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. E. Nikoletseas, W. Thomas (Eds.), Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP-2009), Vol. 5555 of Lecture Notes in Computer Science, Springer, 2009, pp. 49–58.

[9] R. Impagliazzo, R. Paturi, On the complexity of k-SAT, Journal of Computer and System Sciences 62 (2) (2001) 367 – 375.

[10] M. Cygan, F. V. Fomin, LOGSPACE. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, S. Saurabh, Lower Bounds Based on the Exponential-Time Hypothesis, Springer International Publishing, Cham, 2015, pp. 467–521.

[11] R. Impagliazzo, R. Paturi, F. Zane, Which problems have strongly exponential complexity?, Journal of Computer and System Sciences 63 (4) (2001) 512 – 530.

[12] R. Santhanam, S. Srinivasan, On the limits of sparsification, in: Proceeding of the 39th International Colloquium on Automata, Languages, and Programming (ICALP-2012), 2012, pp. 774–785.

[13] P. Jonsson, V. Lagerkvist, B. Roy, Fine-grained time complexity of constraint satisfaction problems, ACM Transactions on Computation Theory 13 (1) (2021).

[14] P. Jonsson, V. Lagerkvist, Lower bounds and faster algorithms for equality constraints, in: Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI-2020), 2020, pp. 1784–1790.

[15] S. Khanna, M. Sudan, L. Trevisan, D. Williamson, The approximability of constraint satisfaction problems, SIAM Journal on Computing 30 (6) (2000) 1863–1920.

[16] N. Creignou, S. Khanna, M. Sudan, Complexity classifications of Boolean constraint satisfaction problems, SIAM Monographs on Discrete Mathematics and Applications, 2001.

[17] H. Schnoor, I. Schnoor, Partial polymorphisms and constraint satisfaction problems, in: N. Creignou, P. G. Kolaitis, H. Vollmer (Eds.), Complexity of Constraints, Vol. 5250 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2008, pp. 229–254.

[18] N. Creignou, B. Zanuttini, A complete classification of the complexity of propositional abduction, SIAM Journal on Computing 36 (1) (2006) 207–229.

[19] G. Nordh, B. Zanuttini, What makes propositional abduction tractable, Artificial Intelligence 172 (2008) 1245–1284.

[20] N. Creignou, H. Vollmer, Boolean constraint satisfaction problems: when does Post's lattice help?, in: N. Creignou, P. G. Kolaitis, H. Vollmer (Eds.), Complexity of Constraints, Vol. 5250, Springer Verlag, Berlin Heidelberg, 2008, pp. 3–37.

[21] J. Thapper, Aspects of a constraint optimisation problem, Ph.D. thesis, Linköping University, The Institute of Technology (2010).

[22] D. A. Cohen, M. C. Cooper, P. G. Jeavons, A. A. Krokhin, The complexity of soft constraint satisfaction, Artificial Intelligence 170 (11) (2006) 983–1016.

[23] E. Post, The two-valued iterative systems of mathematical logic, Annals of Mathematical Studies 5 (1941) 1–122.

[24] D. Lau, Function Algebras on Finite Sets: Basic Course on Many-Valued Logic and Clone Theory (Springer Monographs in Mathematics), Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[25] P. Jeavons, On the algebraic structure of combinatorial problems, Theoretical Computer Science 200 (1998) 185–204.

[26] T. J. Schaefer, The complexity of satisfiability problems, in: Proceedings 10th Symposium on Theory of Computing (STOC-1978), ACM Press, 1978, pp. 216–226.

[27] V. Lagerkvist, Weak bases of Boolean co-clones, Information Processing Letters 114 (9) (2014) 462–468.

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

[29] N. Creignou, J. Schmidt, M. Thomas, Complexity classifications for propositional abduction in Post's framework, Journal of Logic and ComputationTo appear (2012).

[30] M. Bodirsky, J. Kára, B. Martin, The complexity of surjective homomorphism problems-a survey, Discrete Appl. Math. 160 (12) (2012) 1680–1690.

[31] P. Jonsson, V. Lagerkvist, G. Nordh, Constructing NP-intermediate problems by blowing holes with parameters of various properties, Theoretical Computer Science 581 (C) (2015) 67–82.

[32] S. Khanna, M. Sudan, L. Trevisan, D. P. Williamson, The approximability of constraint satisfaction problems, SIAM Journal on Computing 30 (6) (2000) 1863–1920. doi:10.1137/S0097539799349948.

[33] N. Creignou, B. Zanuttini, A complete classification of the complexity of propositional abduction, SIAM Journal on Computing 36 (1) (2006) 207–229.

[34] T. Feder, 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) (1998) 57–104.

[35] J. Batson, D. A. Spielman, N. Srivastava, Twice-ramanujan sparsifiers, SIAM Journal on Computing 41 (6) (2012) 1704 – 1721.