Fine-Grained Time Complexity of Constraint Satisfaction Problems[*]

Peter Jonsson[†1], Victor Lagerkvist[‡1], and Biman Roy[§2]

[1]Department of Computer and Information Science, Linköping University, Linköping, Sweden

[2]Independent researcher, Drabantgatan 5, Linköping, 58214, Östergötland, Sweden

Abstract

We study the constraint satisfaction problem (CSP) parameterized by a constraint language Γ (CSP(Γ)) and how the choice of Γ affects its worst-case time complexity. Under the exponential-time hypothesis (ETH), we rule out the existence of subexponential algorithms for finite-domain NP-complete CSP(Γ) problems. This extends to certain infinite-domain CSPs and structurally restricted problems. For CSPs with finite domain D and where all unary relations are available, we identify a relation S_D such that the time complexity of the NP-complete problem CSP($\{S_D\}$) is a lower bound for all NP-complete CSPs of this kind. We also prove that the time complexity of CSP($\{S_D\}$) strictly decreases when $|D|$ increases (unless the ETH is false), and provide stronger complexity results in the special case when $|D| = 3$.

1 Introduction

The *constraint satisfaction problem* over a constraint language Γ (CSP(Γ)) is the computational decision problem of verifying whether a set of constraints over Γ is satisfiable or not. The principal aim of this article is to investigate the seemingly large discrepancy in time complexity of NP-complete CSPs, using methods from universal algebra. This study can be seen as a continuation of Jonsson et al. [35] who studied related questions for Boolean Γ. However, constraint languages over arbitrary finite domains introduce a large array of

[*]A preliminary version of this article appeared in Proceedings of the 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS 2017).

[†]peter.jonsson@liu.se

[‡]victor.lagerkvist@liu.se

[§]cse.biman@gmail.com

complicating factors, requiring us to use and develop several new algebraic techniques to accomplish our goal.

This introduction is split into three sections which describes these goals in greater detail. We begin in Section 1.1 by highlighting similar research and describe the overarching research field, continue in Section 1.2 by explaining our algebraic approach, and in Section 1.3 we conclude the introduction by outlining the main results established in the article.

1.1 Background and Motivation

The constraint satisfaction problem is widely studied from both a theoretical and a practical standpoint. From a practical point of view this problem can be used to model many natural problems occurring in real-world applications. From a more theoretical point of view the CSP problem is (among several other things) of great interest due to its connections with *universal algebra*. The details of the so-called *algebraic approach* will be expanded in Section 1.2, but at the moment it is sufficient to know that there exists an algebraic correspondence between constraint languages and certain closed sets of operations, *polymorphisms*, determining the complexity of a CSP up to polynomial-time reductions [32]. Most recently, the algebraic approach culminated into a dichotomy theorem separating tractable from NP-complete CSPs [15, 47].

However, the mere fact that two CSPs are polynomial-time interreducible does not offer much insight into their relative worst-case time complexity. For example, on the one hand, it has been conjectured that the Boolean satisfiability problem with unrestricted clause length, CNF-SAT, is not solvable strictly faster than $O(2^n)$, where n denotes the number of variables [30]. On the other hand, k-SAT is known to be solvable strictly faster than $O(2^n)$ for every $k \geq 1$ [29], and even more efficient algorithms are known for severely restricted satisfiability problems such as 1-in-3-SAT [46]. This discrepancy in complexity stems from the fact that a polynomial-time reduction can change the structure of an instance and e.g. introduce a large number of fresh variables. Hence, it is worthwhile to study the complexity of NP-complete CSPs using more fine-grained notions of reductions. To make this a bit more precise, given a constraint language Γ we let

$$\mathsf{T}(\Gamma) = \inf\{c \mid \mathrm{CSP}(\Gamma) \text{ is solvable in time } 2^{cn}\}$$

where n denotes the number of variables. For each k let Γ^k_{SAT} denote the constraint language where each relation is the set of models of a k-ary clause. If $\mathsf{T}(\Gamma) = 0$ then $\mathrm{CSP}(\Gamma)$ is said to be solvable in *subexponential time*, and the conjecture that $\mathrm{CSP}(\Gamma^3_{\mathrm{SAT}})$ (i.e., 3-SAT) is not solvable in subexponential time is known as the *exponential-time hypothesis* (ETH) [30]. The stronger conjecture that the limit of the sequence $\mathsf{T}(\Gamma^3_{\mathrm{SAT}})$, $\mathsf{T}(\Gamma^4_{\mathrm{SAT}}), \mathsf{T}(\Gamma^5_{\mathrm{SAT}}), \ldots$ tends to 1 is known as the *strong exponential-time hypothesis* [30, 18].

It is worth remarking that no concrete values of $\mathsf{T}(\Gamma)$ are known when $\mathrm{CSP}(\Gamma)$ is NP-complete. Despite this, studying properties of the function T can still be of great interest since such properties can be used to compare and relate the worst-case running times of NP-complete CSP problems. Moreover, for Boolean constraint languages, several properties of the function T are known. For example, it is known that there exists a finite Boolean constraint language Γ such that $\mathrm{CSP}(\Gamma)$ is NP-complete and $\mathsf{T}(\Gamma) = 0$ if and only if $\mathsf{T}(\Delta) = 0$ for *every* Boolean constraint language Δ [35]. Hence, even though the status of the ETH is unclear at the moment, finding a subexponential time algorithm for one NP-complete Boolean CSP problem is tantamount to being able to solve every Boolean CSP problem in subexponential time. It is also known that there exists a Boolean relation $R^{\neq\neq\neq 01}_{1/3}$ such that $\mathrm{CSP}(\{R^{\neq\neq\neq 01}_{1/3}\})$ is NP-complete but $\mathsf{T}(\{R^{\neq\neq\neq 01}_{1/3}\}) \leq \mathsf{T}(\Gamma)$ for *every* Boolean constraint language Γ such that $\mathrm{CSP}(\Gamma)$ is NP-complete. In Jonsson et al. [35] this problem is referred to as the *easiest NP-complete Boolean CSP problem.* The existence of this relation e.g. rules out the possibility that for each Boolean constraint language Γ there exists Δ such that $\mathsf{T}(\Delta) < \mathsf{T}(\Gamma)$ — a scenario which otherwise would have been compatible with the ETH. Hence, even though no concrete values are known for $\mathsf{T}(\Gamma)$ when $\mathrm{CSP}(\Gamma)$ is NP-complete, quite a lot is known concerning the relationship between $\mathsf{T}(\Gamma)$ and $\mathsf{T}(\Delta)$ for Boolean Γ and Δ.

In this article we are interested in studying properties of the function T for constraint languages Γ over arbitrary finite domains. In particular we are interested in relating $\mathsf{T}(\Gamma)$ to the ETH, and the possibility of implicitly bounding T from below by finding an "easiest NP-complete $\mathrm{CSP}(\Gamma)$ problem". However, in contrast to the Boolean domain where the algebraic landscape is relatively simple, arbitrary finite domains pose significant obstacles to overcome, which is the topic of the forthcoming section.

1.2 The Algebraic Approach

In this section we describe the algebraic approach in greater detail and outline how it is used in this article. A k-ary operation $f\colon D^k \to D$ over a universe, or domain, D is said to be a *polymorphism* of a constraint language Γ if, for every relation $R \in \Gamma$, $f(t_1, \ldots, t_k) \in R$ for every $t_1, \ldots, t_k \in R$, where f is applied componentwise to the tuples $t_1, \ldots, t_k$. Another way of viewing this is that the tuples $t_1, \ldots, t_k$ form the rows in a matrix, and that one then applies the function f to the columns of this matrix. Hence, a polymorphism may simply be viewed as a homomorphism from the kth power of R to R itself, and we write $\mathrm{Pol}(\Gamma)$ for the set of operations preserving all relations in the constraint language Γ. Sets of operations of the form $\mathrm{Pol}(\Gamma)$ are known as *clones* and form a lattice for every domain when ordered by set inclusion. The importance of polymorphisms in the context of CSPs stems from their correspondence to sets of relations closed under logical formulas consisting of existential quantification, conjunction, and equality constraints, so-called *primitive positive definitions* (pp-definitions). With this correspondence Jeavons [32] proved that $\mathrm{CSP}(\Gamma)$ is polynomial-time many-one reducible to $\mathrm{CSP}(\Delta)$ if $\mathrm{Pol}(\Delta) \subseteq \mathrm{Pol}(\Gamma)$.

One shortcoming of this approach is that the lattice of clones, despite being well understood in the Boolean domain due to Post's lattice [38], are not nearly as well described for arbitrary finite domains. Hence, it was realized early that more powerful methods for characterizing polymorphisms was needed [17], and the majority of research concentrated on classifying polymorphisms according to *identities*. In simple terms, this approach offers a simplification in the sense that instead of studying properties of specific polymorphisms, it is sufficient to study classes of algebras satisfying certain identities, so-called *varieties*. In this article we will concentrate on the relational counterpart to such classes of algebras, *primitive positive interpretations*, which is a generalization of pp-definitions which can be used to compare the expressive strength of constraint languages defined over different finite domains (see Section 2.4 for a formal definition). Similar to pp-definitions it is then possible to use pp-interpretations in order to obtain polynomial-time many-one reductions between CSPs (cf. Theorem 5.5.6 in Bodirsky [5]). With the notion of pp-interpretations the CSP dichotomy theorem can then simply be stated as follows [17, 15, 47].

Theorem 1. *Let Γ be an idempotent constraint language over a finite domain. Then* $\mathrm{CSP}(\Gamma)$ *is NP-complete if* Γ *pp-interprets 3-SAT and is tractable otherwise.*

By idempotent we here mean that the constraint language contains all possible constants over the domain. However, it is a priori not obvious how the assumption that Γ pp-interprets 3-SAT can be used to characterise the behaviour of $\mathsf{T}(\Gamma)$. In the Boolean domain Jonsson et al. [35] handled this difficulty by considering more refined algebras than polymorphisms, so-called *partial polymorphisms*. These operations are defined similarly to total polymorphisms with the exception that they are allowed to be undefined, and we write $\mathrm{pPol}(\Gamma)$ for the set of partial polymorphisms of the constraint language Γ. Sets of the form $\mathrm{pPol}(\Gamma)$ are sometimes referred to as *strong partial clones* and it is known that $\mathsf{T}(\Gamma) \leq \mathsf{T}(\Delta)$ if $\mathrm{pPol}(\Delta) \subseteq \mathrm{pPol}(\Gamma)$ [35]. Hence, partial polymorphisms carry sufficient information to study the worst-case time complexity of NP-complete CSPs. It is also worth remarking that partial polymorphisms are not only useful when studying CSPs with this very fine-grained notion of complexity, and have been used to study the classical complexity of many different computational problems where polymorphisms are not directly applicable [4, 12, 16, 28].

However, it is not straightforward to see how Theorem 1 together with partial polymorphisms can be used to say anything non-trivial concerning the function T for arbitrary finite domains. In the Boolean case Jonsson et al. [35] essentially circumvented this difficulty by utilising Schaefer's dichotomy theorem [43] together with Post's lattice [38] of Boolean clones. This approach is not possible for larger domains since little is known about the set $\{\mathrm{Pol}(\Gamma) \mid \Gamma$ pp-interprets 3-SAT$\}$. We describe how to overcome this difficulty in Section 1.3 where the results of the article are described in greater detail.

1.3 Our Results

As a starting point we begin in Section 3 by investigating the possibility of finding NP-complete CSPs solvable in subexponential time. That is, assuming the ETH, can there exist CSP(Γ) where Γ pp-interprets 3-SAT and where $\mathsf{T}(\Gamma) = 0$? For this question we develop a complete understanding and prove that (1) the ETH is false if and only if (2) there exists a finite constraint language Γ over a finite domain such that CSP(Γ) is NP-complete and $\mathsf{T}(\Gamma) = 0$, if and only if (3) $\mathsf{T}(\Gamma) = 0$ for every finite constraint language Γ defined over a finite domain. In other words, finding a subexponential time algorithm for a single NP-complete, finite-domain CSP problem is tantamount to being able to solve all CSP problems in subexponential time. We also study structurally restricted CSPs where the maximum number of constraints a variable may appear in is bounded by a constant B (CSP(Γ)-B). For problems of this form our results are not as sharp, but we prove that, if CSP(Γ) is NP-complete and Γ satisfies an additional algebraic condition, then there exists a constant B such that CSP(Γ)-B is not solvable in subexponential time (unless the ETH is false). We also remark that our proof extends to certain constraint languages defined over infinite domain, and give several examples of infinite-domain NP-complete CSP problems that are not solvable in subexponential time, unless the ETH is false. These results may be interesting to compare to those of De Haan et al. [23], who study subexponential algorithms for structurally restricted CSPs. One crucial difference to our results is that De Haan et al. do not consider constraint language restrictions. For example, it is proved that CSP(Δ)-2, where Δ is the set of all finite-domain relations, is not solvable in subexponential time unless the ETH is false. However, a result of this form tells us very little about the complexity of CSP(Γ)-2 for specific constraint languages, since it does not imply that CSP(Γ)-2 is not solvable in subexponential time for every Γ such that CSP(Γ)-2 is NP-complete.

We have thus established that $\mathsf{T}(\Gamma) > 0$ for every NP-complete, finite-domain CSP(Γ), assuming only the ETH. This immediately raises the question of which further insights can be gained concerning the behaviour of the function T. Is the easiest NP-complete SAT problem also the easiest NP-complete CSP problem (for each fixed domain)? If not, is it possible to find an easiest CSP problem over the domain, or can one for every $c > 0$ construct an NP-complete CSP solvable in $O(2^{cn})$ time? Or, formulated using T, for some fixed finite domain, is it possible to construct an infinite chain of NP-complete CSPs with strictly decreasing complexity, such that T tends to 0? This would not directly contradict the ETH, but would certainly make finite-domain CSPs fundamentally different to SAT problems. We study such questions in Section 4 for CSPs where in an instance one is allowed to restrict the values of individual variables arbitrarily. This restricted CSP problem is particularly well-studied, and it is used as *the* definition of CSPs in many cases: see, for instance, the textbook by Russell and Norvig [42, Section 3.7] and the handbook by Rossi et al. [41, Section 2]. This may be viewed as restricting oneself to constraint languages that contain all unary relations. A closely related restriction (that is typically used when studying CSPs from the algebraic viewpoint) is that every unary relation is primitively positively definable in Γ (see Section 2). Such constraint languages are known as *conservative*. These two restrictions are computationally equivalent up to polynomial-time many-one reductions, but it is not known whether they are equivalent with respect to the function T. Thus, we need to separate them, so we say that a constraint language that contains all unary relations is *ultraconservative*. Clearly, the CSP dichotomy theorem is valid also for conservative and ultraconservative constraint languages, but the conservative case was known to hold well before Theorem 1 [14].

We then show that for every finite domain D there exists a relation S_D such that CSP($\{S_D\}$) is NP-complete and $\mathsf{T}(\{S_D\}) = \mathsf{T}(\{S_D\} \cup 2^D) \leq \mathsf{T}(\Gamma)$ for every ultraconservative and NP-complete CSP(Γ) over D (2^D is the set of all unary relations over D). This relation will be formally defined in Section 4.1, but it is worth pointing out that S_D contains only three tuples and that CSP($\{S_D\}$) can be viewed as a higher-domain variant of the monotone 1-in-3-SAT problem. We refer to CSP($\{S_D\} \cup 2^D$) as the *easiest NP-complete ultraconservative CSP problem over D*. Note that the properties of the relation S_D rule out the possibility of an infinite sequence of ultraconservative languages $\Gamma_1, \Gamma_2, \ldots$ such that each CSP(Γ_i) is NP-complete and $\mathsf{T}(\Gamma_i)$ tends to 0, but also have stronger implications, since the value $\mathsf{T}(\{S_D\})$ is a conditional lower bound for the complexity of *all* NP-complete, ultraconservative CSPs over D.

To prove the existence of the relation S_D we begin in Section 4.1 by first proving that there for each finite D and NP-hard and conservative CSP(Γ) exists a relation R over D of cardinality 3 such that (1) CSP($\{R\}$) is NP-complete and (2) $\mathsf{T}(\{R\}) \leq \mathsf{T}(\Gamma)$. However, this is not enough in order to find an "easiest problem", since for every finite domain there exists a large number of such relations. In Section 4.2 we show that $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\{R\})$ for every such relation R of cardinality 3. We then analyse the time complexity of the problem CSP($\{S_D\}$) and prove that $\mathsf{T}(\{S_D\})$ tends to 0 for increasing values of $|D|$. This also shows, despite the fact that no finite-domain NP-complete CSP(Γ) is solvable in subexponential time (if the ETH is true), that for every $c > 0$ one can find Γ over a finite domain such that CSP(Γ) is NP-complete and solvable in $O(2^{cn})$ time. Based on an existing reduction [36], we also prove that the CSP problem over D where every possible constraint over D may appear in an instance, is not solvable strictly faster than $O(|D|^n)$ time (unless the SETH is false). Colloquially speaking, one interpretation of these results is that easy CSPs become easier, while the hard CSPs become harder, when the domain increases in size.

Having identified an easiest ultraconservative NP-complete CSP for every finite domain, it is tempting to prove similar results for arbitrary NP-complete CSPs. As a reasonable starting point we first investigate the case when Γ is conservative but not necessary ultraconservative (in Section 5). Surprisingly, even the conservative case is far from straightforward, and to make progress we have to develop new algebraic constructions. Furthermore, in contrast to the ultraconservative case, these constructions explicitly require the usage of partial polymorphisms, giving the impression that further progress could hinge on stronger algebraic techniques. For ternary domains D we then manage to prove that $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\Gamma)$ for every conservative Γ over D such that CSP(Γ) is NP-complete, but it is not obvious that similar proof techniques are applicable to larger domains.

When adjoined, our results in this article clearly demonstrate that the function T can be analysed without an extensive knowledge of the polymorphisms related to a constraint language. Hence, fine-grained complexity analyses are in no way limited to Boolean CSPs but can be pursued for much richer classes of problems. We have also established that lower bounds based on the ETH and the SETH can be obtained for both finite-domain CSPs and many classes of infinite-domain CSPs.

2 Preliminaries

In this section we introduce and define the technical notions needed for the remainder of the article.

2.1 Relations and Constraint Languages

A k-ary *relation* R over a set D is a subset of D^k, and we write $\mathrm{ar}(R) = k$ to denote its arity. A finite set of relations Γ over a set D is called a *constraint language*. We will typically refer to the set D as a *domain*, and will always assume that $|D| > 1$.

Given two tuples s and t we let $s^\frown t$ denote the concatenation of s and t, i.e., if $s = (s_1, \ldots, s_{k_1})$ and $t = (t_1, \ldots, t_{k_2})$ then $s^\frown t = (s_1, \ldots, s_{k_1}, t_1, \ldots, t_{k_2})$. If t is an n-ary tuple we let $t[i]$ denote its ith element and $\mathrm{Proj}_{i_1,\ldots,i_{n'}}(t) = (t[i_1], \ldots, t[i_{n'}])$, $n' \leq n$, denote the *projection* of t on the coordinates $i_1, \ldots, i_{n'} \in \{1, \ldots, n\}$. Similarly, if R is an n-ary relation we let

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

We write Eq_D for the equality relation $\{(x, x) \mid x \in D\}$. If there is no risk for confusion we omit the subscript and simply write Eq. For each $d \in D$ we write R^d for the unary, constant relation $\{(d)\}$. We will occasionally represent relations by first-order formulas, and if $\varphi(x_1, \ldots, x_k)$ is a first-order formula with free variables $x_1, \ldots, x_k$ then we write $R(x_1, \ldots, x_k) \equiv \varphi(x_1, \ldots, x_k)$ to define the relation $R = \{(f(x_1), \ldots, f(x_k)) \mid f \text{ is a model of } \varphi(x_1, \ldots, x_k)\}$.

As a graphical representation, we will sometimes view a k-ary relation $R = \{t_1, \ldots, t_m\}$ as an $m \times k$ matrix where the columns of the matrix enumerate the arguments of the relation (in some fixed ordering). For example,

$$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

represents the relation $\{(0, 0, 1, 1), (0, 1, 0, 1)\}$.

2.2 The Constraint Satisfaction Problem

The *constraint satisfaction problem* over a constraint language Γ over D ($\mathrm{CSP}(\Gamma)$) is the computational decision problem defined as follows.

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

If $\Gamma = \{R\}$ is singleton then we write $\mathrm{CSP}(R)$ instead of $\mathrm{CSP}(\{R\})$, and if Γ is Boolean we typically write $\mathrm{SAT}(\Gamma)$ instead of $\mathrm{CSP}(\Gamma)$. We let $\mathbb{B} = \{0, 1\}$. For example, if $R_{1/3} = \{(0, 0, 1), (0, 1, 0), (1, 0, 0)\}$, then $\mathrm{SAT}(R_{1/3})$ is an alternative formulation of 1-in-3-SAT without negation. Also, for each $k \geq 3$ let Γ^k_{SAT} be the constraint language which for every $t \in \mathbb{B}^k$ contains the relation $\mathbb{B}^k \setminus \{t\}$. Then, $\mathrm{SAT}(\Gamma^k_{\mathrm{SAT}})$ can be viewed as an alternative formulation of k-SAT. Last, let us define the 8-ary relation $R^{\neq\neq\neq 01}_{1/3}$ from Jonsson et al [35] as

$$R^{\neq\neq\neq 01}_{1/3} = \{(0, 0, 1, 1, 1, 0, 0, 1), (0, 1, 0, 1, 0, 1, 0, 1), (1, 0, 0, 0, 1, 1, 0, 1)\}.$$

Then $\mathrm{SAT}(R^{\neq\neq\neq 01}_{1/3})$ can be seen as a variant of $\mathrm{SAT}(R_{1/3})$ where each variable in each constraint is constant (0 or 1) or has a complementary variable (explaining the three inequalities in the notation $R^{\neq\neq\neq 01}_{1/3}$). We will return to this SAT problem several times in the sequel.

2.3 Representing Constraints

We have defined a constraint language Γ as a finite set of relations, implying that constraints in instances of $\mathrm{CSP}(\Gamma)$ can be represented in a multitude of equivalent ways. Since our main interest in this article lies in studying the time complexity of CSPs over finite constraint languages, such a relaxed notion of representation is typically sufficient.

One exception to this finitary stance is when discussing the so-called *uniform* CSP problem over a finite domain D where every possible relation over D may appear in an instance. We let D-CSP denote this problem and adopt a simple representation of constraints: each involved relation is explicitly represented as a list of tuples. This is the predominant method of representing constraints in the majority of theoretical CSP research, but is in stark contrast to CNF-SAT where a clause of arbitrary length is usually encoded by a list of literals. Hence, the only difference between $\{0, 1\}$-CSP and CNF-SAT is the preferred choice of representation.

2.4 Primitive Positive Definitions and Interpretations

Let Γ be a constraint language. A k-ary relation R is said to have a *primitive positive definition* (pp-definition) over Γ if

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

where each $R_i \in \Gamma \cup \{\mathrm{Eq}\}$ and each $\mathbf{x_i}$ is an $\mathrm{ar}(R_i)$-ary tuple of variables over $x_1, \ldots, x_k$, $y_1, \ldots, y_{k'}$. In addition, if the primitive positive formula does not contain any existentially quantified variables, we say that it is a *quantifier-free primitive positive formula* (qfpp), and if each $R_i \in \Gamma$, we say that it is a *equality-free primitive positive formula* (efpp). Hence, an efpp-formula can only utilise equality constraints if they are already included in Γ.

For example, the reader can verify that the standard reduction from k-SAT to $(k-1)$-SAT, where a clause of length k is replaced by clauses of length $k-1$ making use of one fresh variable, can be formulated as a pp-definition but not as a qfpp-definition. We write $\langle \Gamma \rangle$, $\langle \Gamma \rangle_{\nexists}$, and $\langle \Gamma \rangle_{\neq}$ to denote the smallest set of relations containing Γ and which are closed under pp-definitions, qfpp-definitions, and efpp-definitions, respectively. The intended mnemonic for the latter two cases is thus that existential quantification, respectively equality, is not allowed. If $\Gamma = \{R\}$ is singleton then we instead write $\langle R \rangle$, $\langle R \rangle_{\nexists}$, and $\langle R \rangle_{\neq}$.

Note that $\langle \Gamma \rangle$ is closed under projections, in the sense that if $R \in \langle \Gamma \rangle$ then $\mathrm{Proj}_{i_1,\ldots,i_n}(R) \in \langle \Gamma \rangle$ for all $i_1, \ldots, i_n \in \{1, \ldots, \mathrm{ar}(R)\}$. The same is true for $\langle \Gamma \rangle_{\neq}$, but it does not necessarily hold for $\langle \Gamma \rangle_{\nexists}$. Jeavons [32] proved the following important result.

Theorem 2. *Let Γ and Δ be constraint languages. If $\Delta \subseteq \langle \Gamma \rangle$ then CSP(Δ) is polynomial-time reducible to CSP(Γ).*

Theorem 2 naturally holds also for relations defined by qfpp- or efpp-formulas. However, there are additional advantages of these more restricted ways of defining relations and we will return to them later on. We are now ready to define the concept of primitive positive interpretations.

Definition 3. *Let D and E be two domains and let Γ and Δ be two constraint languages over D and E, respectively. A* primitive positive interpretation *(pp-interpretation) of Δ over Γ consists of a d-ary relation $F \subseteq D^d$ and a surjective function $f : F \to E$ such that $F, f^{-1}(\mathrm{Eq}_E) \in \langle \Gamma \rangle$ and $f^{-1}(R) \in \langle \Gamma \rangle$ for every $R \in \Delta$, where $f^{-1}(R)$, $\mathrm{ar}(R) = k$, denotes the $(k \cdot d)$-ary relation*

$$\{(x_1^1, \ldots, x_1^d, \ldots, x_k^1, \ldots, x_k^d) \in D^{k \cdot d} \mid$$
$$(f(x_1^1, \ldots, x_1^d), \ldots, f(x_k^1, \ldots, x_k^d)) \in R\}.$$

The main purpose of pp-interpretations is to relate constraint languages which might be incomparable with respect to pp-definitions. Let us consider a concrete example.

Example 1. *Recall that $R_{1/3} = \{(0,0,1),(0,1,0),(1,0,0)\}$, and let $R = \{(0,0,2),(0,2,0),(2,0,0)\}$. Then neither $R_{1/3} \in \langle R \rangle$ nor $R \in \langle R_{1/3} \rangle$, even though $\mathrm{CSP}(R_{1/3})$ and $\mathrm{CSP}(R)$ are trivially reducible to each other. However, it is easy to show that R can pp-interpret $R_{1/3}$, and vice versa: simply choose $d = 1$, $F = \{(0),(2)\}$ and the parameter $f(0) = 0$, $f(2) = 1$. Then $F \in \langle R \rangle$ since $F(x) \equiv \exists y, z. R(x,y,z)$, $f^{-1}(R_{1/3}) = R$ and $f^{-1}(\mathrm{Eq}_{\mathbb{B}}) = \mathrm{Eq}_{\{0,2\}}$, from which it follows that R can pp-interpret $R_{1/3}$.*

Second, consider the relation $R_{\neq} = \{(x,y) \in \{0,1,2\}^2 \mid x \neq y\}$, and observe that $\mathrm{CSP}(R_{\neq})$ corresponds to the 3-coloring problem. We invite the reader to verify that the standard reduction from 3-coloring to 3-SAT can be phrased as a pp-interpretation of $R_{\neq}$ over Γ^3_{SAT}, but that this reduction cannot be expressed via pp-definitions due to the different domains.

Hence, pp-interpretations are generalisations of pp-definitions, and can be used to obtain polynomial-time reductions between CSPs.

Theorem 4 (cf. Theorem 5.5.6 in Bodirsky [5]). *If Γ, Δ are constraint languages and there is a pp-interpretation of Δ over Γ, then CSP(Δ) is polynomial-time reducible to CSP(Γ).*

2.5 Polymorphisms and Partial Polymorphisms

Let f be a k-ary function over a domain D. We write $\mathrm{ar}(f) = k$ to denote the arity of f and will occasionally also refer to f as an *operation.* For a set D we write OP_D for the set of all (finitary) operations over D. An operation $f\colon D^k \to D$ is said to be *idempotent* if $f(x, \dots, x) = x$ for each $x \in D$. A k-ary *partial function*, or a *partial operation*, is a map $X \to D$ where $X \subseteq D^k$, and we write pOP_D for the set of all (finitary) partial operations over D. If f is a k-ary partial operation we let $\mathrm{ar}(f) = k$ denote its arity and $\mathrm{dom}(f) = X$ denote its domain, i.e., the set of values where it is defined.

If f is a k-ary operation and R an n-ary relation, both defined over the same domain, it is easy to extend f to tuples $t_1, \dots, t_k$ from R as follows:

$$f(t_1, \dots, t_k) = (f(t_1[1], \dots, t_k[1]), \dots, f(t_1[n], \dots, t_k[n])).$$

This immediately leads to the following important definition.

Definition 5. *Let $f : D^k \to D$ be a k-ary operation and $R \subseteq D^n$ an n-ary relation. The operation f is a* polymorphism *of R if $f(t_1, \dots, t_k) \in R$ for each sequence of tuples $t_1, \dots, t_k \in R$.*

This notion is easy to generalise to partial operations as follows.

Definition 6. *Let f be a k-ary partial operation over D and $R \subseteq D^n$ an n-ary relation. The partial operation f is a* partial polymorphism *of R if $f(t_1, \dots, t_k) \in R$ for each sequence of tuples $t_1, \dots, t_k \in R$ such that $(t_1[i], \dots, t_n[i]) \in \mathrm{dom}(f)$ for each $1 \leq i \leq n$.*

If f is a polymorphism or a partial polymorphism of a relation R then we occasionally also say that R is *invariant* under f, or that R is *closed* under f.

Definition 7. *Let R be a relation over D and Γ a constraint language over D. We make the following definitions.*

1. $\mathrm{Pol}(R) = \{f \in \mathrm{OP}_D \mid f$ *is a polymorphism of* $R\}$.
2. $\mathrm{pPol}(R) = \{f \in \mathrm{pOP}_D \mid f$ *is a partial polymorphism of* $R\}$.
3. $\mathrm{Pol}(\Gamma) = \bigcap_{R\in\Gamma} \mathrm{Pol}(R)$.
4. $\mathrm{pPol}(\Gamma) = \bigcap_{R\in\Gamma} \mathrm{pPol}(R)$.

Sets of the form $\mathrm{Pol}(\Gamma)$ are in the literature usually known as *clones* and sets of the form $\mathrm{pPol}(\Gamma)$ known as *strong partial clones.* Both clones and strong partial clones form a lattice when ordered by set inclusion. For the majority of applications in this paper we will not need any sophisticated algebraic machinery, but we remark that the lattice of Boolean clones, *Post's lattice*, is countably infinite and completely described [38], but that each larger domain contains a continuum of clones [25] with a mostly unknown structure. For strong partial clones the situation is even more severe as there exists a continuum of strong partial clones already in the Boolean domain [1], and only small fragments of this lattice have been determined [45], while the remaining parts are largely unexplored.

We write $\mathrm{Inv}(F)$ to denote the set of all relations invariant under the set of total or partial functions F. It is known that $\mathrm{Inv}(\mathrm{Pol}(\Gamma)) = \langle\Gamma\rangle$ and that $\mathrm{Inv}(\mathrm{pPol}(\Gamma)) = \langle\Gamma\rangle_{\nexists}$, giving rise to the following *Galois connections*.

Theorem 8 ([10, 11, 26, 40])**.** *Let Γ and Γ' be two constraint languages. Then $\Gamma \subseteq \langle\Gamma'\rangle$ if and only if $\mathrm{Pol}(\Gamma') \subseteq \mathrm{Pol}(\Gamma)$ and $\Gamma \subseteq \langle\Gamma'\rangle_{\nexists}$ if and only if $\mathrm{pPol}(\Gamma') \subseteq \mathrm{pPol}(\Gamma)$.*

2.6 Time complexity and size-preserving reductions

Given a constraint language Γ we let

$$\mathsf{T}(\Gamma) = \inf\{c \mid \mathrm{CSP}(\Gamma) \text{ is solvable in time } 2^{cn}\}$$

where n denotes the number of variables in a given instance. If $\mathsf{T}(\Gamma) = 0$ then $\mathrm{CSP}(\Gamma)$ is said to be *subexponential*, or solvable in *subexponential time*. The conjecture that $\mathsf{T}(\Gamma^{3}_{\mathrm{SAT}}) > 0$ is known as the *exponential-time hypothesis* (ETH) [30], and the conjecture that $\lim_{k\to\infty} \mathsf{T}(\Gamma^{k}_{\mathrm{SAT}}) = 1$ is known as the *strong exponential- time hypothesis* (SETH). Hence, the ETH states that 3-SAT is not solvable in subexponential time, and the SETH that the complexity of k-SAT tends to 2^n for increasing values of k. Note in particular that the SETH implies that CNF-SAT is not solvable in $O(c^n)$ time for any $c < 2$.

We now introduce a type of reduction useful for studying the complexity of CSPs with respect to the function T.

Definition 9. *Let Γ and Δ be two constraint languages. The function f from the instances of $\mathrm{CSP}(\Gamma)$ to the instances of $\mathrm{CSP}(\Delta)$ is a* many-one linear variable reduction *(LV-reduction) with parameter $d \geq 0$ if (1) f is a polynomial-time many-one reduction from $\mathrm{CSP}(\Gamma)$ to $\mathrm{CSP}(\Delta)$ and (2) $|V'| = d\cdot|V| + O(1)$ where V, V' are the set of variables in I and $f(I)$, respectively.*

The term CV-reduction, short for *constant variable reduction*, is used to denote LV-reductions with parameter 1, and we write $\mathrm{CSP}(\Gamma) \leq^{\mathrm{CV}} \mathrm{CSP}(\Delta)$ when $\mathrm{CSP}(\Gamma)$ has a CV-reduction to $\mathrm{CSP}(\Delta)$. It follows that if $\mathrm{CSP}(\Gamma) \leq^{\mathrm{CV}} \mathrm{CSP}(\Delta)$ then $\mathsf{T}(\Gamma) \leq \mathsf{T}(\Delta)$, and if $\mathrm{CSP}(\Gamma)$ LV-reduces to $\mathrm{CSP}(\Delta)$ then $\mathsf{T}(\Gamma) = 0$ if $\mathsf{T}(\Delta) = 0$. Note that $\mathsf{T}(\Gamma) \leq \mathsf{T}(\Delta)$ also holds if there exists a polynomial-time many-one reduction from $\mathrm{CSP}(\Gamma)$ to $\mathrm{CSP}(\Delta)$ which introduces $o(|V|)$ fresh variables, but for our purposes CV-reductions are sufficient.

We then have the following theorem from Jonsson et al. [35], relating the partial polymorphisms of constraint languages with the existence of CV-reductions.

Theorem 10 ([35]). *Let D be a finite domain and let Γ and Δ be two constraint languages over D. If* $\mathrm{pPol}(\Delta) \subseteq \mathrm{pPol}(\Gamma)$ *then* $\mathrm{CSP}(\Gamma) \leq^{\mathrm{CV}} \mathrm{CSP}(\Delta)$.

We remark that the original proof only concerned Boolean constraint languages but that the same proof also works for arbitrary finite domains. Using Theorem 10 and algebraic techniques from Schnoor and Schnoor [44], Jonsson et al. [35] proved that $\mathsf{T}(\{R^{\neq\neq\neq01}_{1/3}\}) \leq \mathsf{T}(\Gamma)$ for any finite Γ such that $\mathrm{SAT}(\Gamma)$ is NP-complete. The problem $\mathrm{SAT}(R^{\neq\neq\neq01}_{1/3})$ was referred to as the *easiest NP-complete* SAT *problem*. We will not go into the details but remark that the proof idea does not work for arbitrary finite domains since it requires a characterisation of every $\mathrm{Pol}(\Gamma)$ such that $\mathrm{CSP}(\Gamma)$ is NP-complete. Such a list is known for the Boolean domain due to Post [38] and Schaefer [43], but not for larger domains.

2.7 Complexity of CSP

Let Γ be a constraint language over a finite domain D. We say that Γ is *idempotent* if $R^d \in \langle\Gamma\rangle$ for every $d \in D$, *conservative* if $2^D \subseteq \langle\Gamma\rangle$, and *ultra-conservative* if $2^D \subseteq \Gamma$. A unary function $f \in \mathrm{Pol}(\Gamma)$ is said to be an *endomorphism*, and if f in addition is bijective it is said to be an *automorphism*. A constraint language Γ is a *core* if every endomorphism is an automorphism. Every finite-domain Γ admits a core which is unique up to isomorphism, and we will therefore sometimes speak of *the core* of Γ. The following theorem is well-known, see e.g., Theorem 3.6 in Barto [2], but is usually expressed in term of polynomial-time many-one reductions instead of CV-reductions.

Theorem 11. *Let Γ be a core constraint language over the domain $\{d_0, \ldots, d_{k-1}\}$. Then* $\mathrm{CSP}(\Gamma \cup \{R^{d_0}, \ldots, R^{d_{k-1}}\}) \leq^{\mathrm{CV}} \mathrm{CSP}(\Gamma)$.

If Γ is a constraint language over $D = \{d_0, \ldots, d_{k-1}\}$, then $\Gamma \cup \{R^{d_0}, \ldots, R^{d_{k-1}}\}$ is both idempotent and a core since its only endomorphism is the identity function on D. The question of whether $\mathrm{CSP}(\Gamma)$ is always tractable or NP-complete has traditionally been referred to as the *CSP dichotomy conjecture* and was first posed by Feder and Vardi [24]. The conjecture was later refined by Bulatov et al. [17] to also induce a sharp characterisation of the tractable and intractable cases, expressed in terms of algebraic properties of the constraint language, and was only recently proved by Bulatov [15] and Zhuk [47]. We will use the following variant of the CSP dichotomy theorem expressed in terms of pp-interpretations.

Theorem 12. *Let Γ be an idempotent constraint language over a finite domain. Then* $\mathrm{CSP}(\Gamma)$ *is NP-complete if Γ pp-interprets Γ^3_{SAT} and tractable otherwise.*

It is worth remarking that if Γ pp-interprets Γ^3_{SAT} then Γ can pp-interpret every finite-domain relation [5, Theorem 5.5.17].

3 Subexponential Time Complexity

For Boolean constraint languages it has been proved that $\mathrm{SAT}(\Gamma^3_{\mathrm{SAT}})$ is solvable in subexponential time if and only if there exists a finite Boolean constraint language Γ such that $\mathrm{SAT}(\Gamma)$ is NP-complete and solvable in subexponential time [35]. In this section we strengthen this result to arbitrary domains and prove that $\mathrm{CSP}(\Gamma)$ is never solvable in subexponential time if Γ can pp-interpret Γ^3_{SAT}, unless the ETH is false.

3.1 Complexity of CSPs in Light of the ETH

Before we present the main result of this section we define a class of structurally restricted CSPs which are relevant when pursuing lower bounds based on the ETH. The *degree* of a variable $x \in V$ of an instance (V, C) of $\mathrm{CSP}(\Gamma)$ is the number of constraints in C containing x. We let $\mathrm{CSP}(\Gamma)$-B, $B \geq 1$, denote the restricted $\mathrm{CSP}(\Gamma)$ problem where each variable occurring in an instance has degree at most B. Note that the number of constraints in an instance (V, C) of $\mathrm{CSP}(\Gamma)$-B is bounded by $|V| \cdot B$.

Now recall that an equality-free primitive positive definition (efpp-definition) is a primitive positive definition without explicit equality constraints, and that Eq_D denotes the equality relation over a domain D. In particular observe that $\mathsf{T}(\Gamma) = \mathsf{T}(\Gamma \cup \{\mathrm{Eq}_D\})$ but that $\mathrm{CSP}(\Gamma \cup \{\mathrm{Eq}_D\})$-$B$ is not necessarily CV-reducible to $\mathrm{CSP}(\Gamma)$-B for every B.

Theorem 13. *Assume that the ETH is true and let Γ be a finite constraint language over a (possibly infinite) domain D such that Γ pp-interprets Γ^3_{SAT}. Then $\mathrm{CSP}(\Gamma)$ is not solvable in subexponential time, and if Γ efpp-defines Eq_D then there exists a constant B, depending only on Γ, such that $\mathrm{CSP}(\Gamma)$-B is not solvable in subexponential time.*

Proof. Due to the assumption that Γ pp-interprets Γ^3_{SAT}, Γ can pp-interpret any Boolean Δ, as was pointed out in Section 2.7. In particular, Γ can pp-interpret the constraint language $\{R^{\neq\neq\neq}_{1/3}\}$ from Jonsson et al. [35], where $R^{\neq\neq\neq}_{1/3} = \mathrm{Proj}_{1,\ldots,6}(R^{\neq\neq\neq 01}_{1/3})$. It is known that $\mathrm{SAT}(R^{\neq\neq\neq}_{1/3})$-2 is NP-complete and that if it is solvable in subexponential time, then the ETH is false [35]. Hence, we will prove the theorem by giving an LV-reduction from $\mathrm{SAT}(R^{\neq\neq\neq}_{1/3})$-2 to $\mathrm{CSP}(\Gamma)$, respectively to $\mathrm{CSP}(\Gamma)$-B for some $B > 0$.

Let $F \subseteq D^d$ and $f \colon F \to \mathbb{B}$ denote the parameters in the pp-interpretation of $\{R^{\neq\neq\neq}_{1/3}\}$. Note in particular that $d \in \mathbb{N}$ is a fixed constant. For some formulas φ_1 and φ_2 let

$$f^{-1}(R^{\neq\neq\neq}_{1/3})(x_{1,1}, \ldots, x_{1,d}, \ldots, x_{6,1}, \ldots, x_{6,d}) \equiv \\ \exists y_1, \ldots, y_{k_1}.\varphi_1(x_{1,1}, \ldots, x_{1,d}, \ldots, x_{6,1}, \ldots, x_{6,d}, y_1, \ldots, y_{k_1})$$

and

$$F(x_1, \ldots, x_d) \equiv \exists z_1, \ldots, z_{k_2}.\varphi_2(x_1, \ldots, x_d, z_1, \ldots, z_{k_2})$$

denote efpp-definitions of $f^{-1}(R_{1/3}^{\neq\neq\neq})$ and F over Γ if Eq_D is efpp-definable over Γ, and otherwise pp-definitions of $f^{-1}(R_{1/3}^{\neq\neq\neq})$ and F over Γ. Let L denote the maximum degree of any variable occurring in these pp-definitions, and note that L is a fixed constant depending only on Γ.

Let $I = (V, C)$ be an instance of $\mathrm{SAT}(\{R_{1/3}^{\neq\neq\neq}\})$-2. Since each variable may occur in at most 2 constraints it follows that $|C| \leq 2|V|$. For each variable x_i introduce d fresh variables $x_{i,1}, \ldots, x_{i,d}$, k_2 fresh variables $z_{i,1}, \ldots, z_{i,k_2}$, and introduce the constraint corresponding to

$$\varphi_2(x_{i,1}, \ldots, x_{i,d}, z_{i,1}, \ldots, z_{i,k_2}).$$

For each constraint $C_i = R_{1/3}^{\neq\neq\neq}(x_i, y_i, z_i, x'_i, y'_i, z'_i)$ introduce k_1 fresh variables $w_{i,1}, \ldots, w_{i,k_1}$ and replace C_i by

$$\varphi_1(x_{i,1}, \ldots, x_{i,d}, y_{i,1}, \ldots, y_{i,d}, z_{i,1}, \ldots, z_{i,d},\\ x'_{i,1}, \ldots, x'_{i,d}, y'_{i,1}, \ldots, y'_{i,d}, z'_{i,1}, \ldots, z'_{i,d}, w_{i,1}, \ldots, w_{i,k_1}).$$

If Γ cannot efpp-define Eq_D then we in addition identify variables according to their occurrence in equality constraints. This can be accomplished by the following: for each equality constraint $\mathrm{Eq}_D(x, y)$ we repeatedly replace each occurrence of y by x throughout the instance. Let $I' = (V', C')$ denote the resulting instance of $\mathrm{CSP}(\Gamma)$. Clearly, I' can be constructed in polynomial time. We begin by proving that I' has a solution if and only if I has a solution. Let $s' : V' \to D$ be a solution to I'. Recall that every variable x_i in V corresponds to a 'block' of variables $x_{i,1}, \ldots, x_{i,d}$ in V'. Now, consider a subset X of constraints corresponding to

$$\varphi_1(x_{i,1}, \ldots, x_{i,d}, y_{i,1}, \ldots, y_{i,d}, z_{i,1}, \ldots, z_{i,d},\\ x'_{i,1}, \ldots, x'_{i,d}, y'_{i,1}, \ldots, y'_{i,d}, z'_{i,1}, \ldots z'_{i,d}, w_{i,1}, \ldots, w_{i,k_1}).$$

Consider one block of variables $x_{i,1}, \ldots, x_{i,d}$. We know that $(s'(x_{i,1}), \ldots, s'(x_{i,d})) \in F$ due to the constraint $F(x_{i,1}, \ldots, x_{i,d})$ and that s' satisfies X. Since X and the block of variables are arbitrarily chosen, we conclude that the function $s : V \to \mathbb{B}$ defined by

$$s(x) = f(s'(x_1), \ldots, s'(x_d))$$

is a solution to I.

Assume instead that $s : V \to \mathbb{B}$ is a solution to I. Arbitrarily choose $t_0, t_1 \in F$ such that $f(t_0) = 0$ and $f(t_1) = 1$. For each variable $x_i \in V$, let $x_{i,1}, \ldots, x_{i,d}$ denote the corresponding block of variables in V', and let $\hat{V} = \{x_{i,1}, \ldots, x_{i,d} \mid x_i \in V\}$ be the set of all such variables. Define the function $\hat{s} : \hat{V} \to F$ such that $\hat{s}(x_{i,j}) = t_0[j]$ if $s(x_i) = 0$ and $\hat{s}(x_{i,j}) = t_1[j]$ otherwise. The function $\hat{s}$ satisfies every constraint $F(x_{i,1}, \ldots, x_{i,d})$ by definition. Consider a subset X of constraints corresponding to

$$\varphi_1(x_{i,1}, \ldots, x_{i,d}, y_{i,1}, \ldots, y_{i,d}, z_{i,1}, \ldots, z_{i,d},\\ x'_{i,1}, \ldots, x'_{i,d}, y'_{i,1}, \ldots, y'_{i,d}, z'_{i,1}, \ldots, z'_{i,d}, w_{i,1}, \ldots, w_{i,k_1}).$$

Recall that φ_1 is a pp-definition of $f^{-1}(R^{\neq\neq\neq}_{1/3})$. Thus, the variables $w_{i,1}, \ldots, w_{i,k_1}$ can be assigned values that in combination with the values provided by $\hat{s}$ satisfies φ_1 and, consequently, X. This implies that there is a solution to I'.

We continue by analysing this reduction. First, observe that if Γ can efpp-define Eq_D then the maximum degree of any variable is $3L$. This follows from the fact that any original variable occurs in at most 2 constraints, resulting in at most $2L$ occurences in the φ_1-constraints, and that we for each variable in addition introduce a φ_2-constraint, contributing at most L additional occurences. This implies that I' is in fact an instance of $\mathrm{CSP}(\Gamma)$-$3L$. Second, note that $|C| \leq 2|V|$, and that we introduce k_1 fresh variables for every constraint in C. This implies that $|V'| \leq |V|d + 2|V|k_1 + k_2$, and, since k_1, k_2 and d are fixed constants, there exists a constant K such that $|V'| = K|V| + O(1)$. Since this reduction is an LV-reduction from $\mathrm{SAT}(R^{\neq\neq\neq}_{1/3})$-2 to $\mathrm{CSP}(\Gamma)$-$3L$ (or to $\mathrm{CSP}(\Gamma)$ if Γ cannot efpp-define Eq_D), it follows that $\mathrm{SAT}(R^{\neq\neq\neq}_{1/3})$-2 is solvable in subexponential time if $\mathrm{CSP}(\Gamma)$-$3L$ (or $\mathrm{CSP}(\Gamma)$) is solvable in subexponential time. $\square$

We have now obtained a complete understanding of subexponential solvability of finite-domain $\mathrm{CSP}(\Gamma)$ problems (modulo the ETH), and made important progress on the structurally restricted problems $\mathrm{CSP}(\Gamma)$-B.

Theorem 14. *The following statements are equivalent.*

1. *The ETH is false.*
2. $\mathrm{CSP}(\Gamma)$ *is solvable in subexponential time for every finite* Γ *over a finite domain.*
3. $\mathrm{CSP}(\Gamma)$-B *is solvable in subexponential time for every* $B \in \mathbb{N}$ *for every finite* Γ *over a finite domain.*
4. *There exists a finite constraint language* Γ *over a finite domain* D *such that* $\mathrm{CSP}(\Gamma)$ *is NP-complete and subexponential.*
5. *There exists a finite constraint language* Γ *over a finite domain* D *such that* $\mathrm{CSP}(\Gamma)$ *is NP-complete,* $\mathrm{Eq}_D \in \langle\Gamma\rangle_{\neq}$*, and* $\mathrm{CSP}(\Gamma)$-B *is subexponential for every* $B \in \mathbb{N}$.

Proof. The implication from (1) to (2) follows from Impagliazzo et al. [31, Theorem 3]. The implication (2) to (3) is trivial since $\mathrm{CSP}(\Gamma)$-B is solvable in subexponential time for every B if $\mathrm{CSP}(\Gamma)$ is solvable in subexponential time. For the implication (3) to (4), we may for example pick the problem $\mathrm{SAT}(R^{\neq\neq\neq}_{1/3})$ from Jonsson et al. [35] which is subexponential if and only if $\mathrm{SAT}(R^{\neq\neq\neq}_{1/3})$-2 is subexponential. For (4) to (5), let Γ be a constraint language over D such that $\mathrm{CSP}(\Gamma)$ is NP-complete and subexponential. We then take the constraint language $\Gamma \cup \{\mathrm{Eq}_D\}$ which trivially can efpp-define Eq_D, and observe that $\mathrm{CSP}(\Gamma \cup \{\mathrm{Eq}_D\})$ is NP-complete and subexponential, implying that $\mathrm{CSP}(\Gamma \cup \{\mathrm{Eq}_D\})$-$B$ is subexponential for every $B \in \mathbb{N}$.

Last, for the implication from (5) to (1), we first observe that Theorem 13 is not directly applicable since it might not be the case that Γ pp-interprets Γ^3_{SAT}

(e.g., if Γ is not core). To circumvent this we have to employ a few algebraic tricks. First, let D be the domain of Γ, and note that $\mathrm{CSP}(\Gamma^c) \leq^{\mathrm{CV}} \mathrm{CSP}(\Gamma)$, where Γ^c is the core of Γ [2, Theorem 3.5], and that $\mathrm{CSP}(\Gamma^c)$ is NP-complete, too. It is also straightforward to verify that $\mathrm{CSP}(\Gamma^c)$-B is subexponential for all $B \in \mathbb{N}$, since $\mathrm{CSP}(\Gamma)$-B is subexponential for every $B \in \mathbb{N}$. Now, let $D' = \{d_1, \ldots, d_k\} \subseteq D$ be the domain of Γ^c. It is easy to see that $\mathrm{Eq}_{D'} \in \langle \Gamma^c \rangle_{\neq}$ since we assumed that $\mathrm{Eq}_D \in \langle \Gamma \rangle_{\neq}$. Indeed, we can simply take any efpp-definition of Eq_D over Γ and replace each constraint by the corresponding constraint over Γ^c, and the resulting efpp-definition correctly defines $\mathrm{Eq}_{D'}$. Hence, since $\mathrm{CSP}(\Gamma^c \cup \{R^{d_1}, \ldots, R^{d_k}\})$ is also NP-complete we now only need to prove that $\mathrm{CSP}(\Gamma^c \cup \{R^{d_1}, \ldots, R^{d_k}\})$-$B$ is subexponential for all $B \in \mathbb{N}$. Let $I = (V, C)$ be an instance of $\mathrm{CSP}(\Gamma^c \cup \{R^{d_1}, \ldots, R^{d_k}\})$-$B$. First, define the relation $R = \{(e(d_1), \ldots, e(d_k)) \mid e \text{ is an endomorphism of } \Gamma^c\}$. This relation is known to be qfpp-definable over Γ^c [2, Theorem 3.6] and we let $\varphi(x_1, \ldots, x_k)$ be a qfpp-definition over Γ^c i.e., $R(x_1, \ldots, x_k) \equiv \varphi(x_1, \ldots, x_k)$. Then we introduce k fresh variables $y_1, \ldots, y_k$ and the constraints $\varphi(y_1, \ldots, y_k)$. Let b denote the maximum degree of any variable in $\varphi(y_1, \ldots, y_k)$, and observe that b is a fixed constant. For each $x_i \in V$ introduce k variables $c_i^1, \ldots, c_i^k$. For each $j \in \{1, \ldots, k\}$ let $I_j \subseteq \{1, \ldots, |V|\}$ denote the set of indices such that $R^{d_j}(x_i) \in C$. In order to ensure that the resulting instance has bounded degree we then for every such index set $I_j = \{i_1, \ldots, i_m\}$ introduce the constraints $\mathrm{Eq}_{D'}(y_j, c^j_{i_1}) \wedge \mathrm{Eq}_{D'}(c^j_{i_1}, c^j_{i_2}) \wedge \mathrm{Eq}_{D'}(c^j_{i_2}, c^j_{i_3}) \wedge \ldots \wedge \mathrm{Eq}_{D'}(c^j_{i_{m-1}}, c^j_{i_m})$. Note that each variable occurs in at most two constraints in this sequence, and that the total number of variables in the resulting instance is bounded by $k \cdot |V| + O(1)$. It follows that the maximum degree of any variable is $b + B + 2$, and since $\mathrm{CSP}(\Gamma^c)$-B' is solvable in subexponential time for all B', $\mathrm{CSP}(\Gamma^c \cup \{R^{d_1}, \ldots, R^{d_k}\})$-$B$ is solvable in subexponential time, too. It follows that $\Gamma^c \cup \{R^{d_1}, \ldots, R^{d_k}\}$ can pp-interpret Γ^3_{SAT}, and Theorem 13 then gives the desired implication from (5) to (1). $\square$

Example 2. *For each finite domain D let $R^D_{\neq}$ denote the binary inequality relation over D, i.e., $R^D_{\neq} = \{(x, y) \in D^2 \mid x \neq y\}$. In Example 1 we saw that $\mathrm{CSP}(R^{\{0,1,2\}}_{\neq})$ is an alternative formulation of the 3-coloring problem, and it is straightforward to verify that $\mathrm{CSP}(R^D_{\neq})$ in general corresponds to the $|D|$-coloring problem. Since $|D|$-coloring is NP-complete for $|D| \geq 3$, a quick application of Theorem 13 gives the desired result that $\mathrm{CSP}(R^D_{\neq})$, $|D| \geq 3$, cannot be solved in subexponential time, unless the ETH is false. This was already proved by Impagliazzo et al. [31, Theorem 4], but it might be interesting to note that our result also carries over to any constraint language Γ such that $\mathrm{Pol}(\Gamma) = \mathrm{Pol}(R^D_{\neq})$. More importantly, we are able to prove this using general methods without actually constructing a concrete, size-preserving reduction between 3-SAT and k-coloring.*

It is also straightforward to show that this result carries over to the bounded degree case. We show how $R^{\neq}_{\{0,1,2\}}$ can efpp-define $\mathrm{Eq}_{\{0,1,2\}}$ (larger domains *follow through similar arguments). Define* $\mathrm{Eq}_{\{0,1,2\}}$ *as*

$$\mathrm{Eq}_{\{0,1,2\}}(x,y) \equiv \exists z_1, z_2 . R^{\neq}_{\{0,1,2\}}(x,z_1) \wedge R^{\neq}_{\{0,1,2\}}(y,z_1) \wedge R^{\neq}_{\{0,1,2\}}(x,z_2) \wedge R^{\neq}_{\{0,1,2\}}(y,z_2) \wedge R^{\neq}_{\{0,1,2\}}(z_1,z_2).$$

By Theorem 13 it then follows that for every $|D| \geq 3$ *there exists* B *such that* $\mathrm{CSP}(R^{D}_{\neq})$*-*$B$ *is NP-complete but not solvable in subexponential time, unless the ETH is false. The ETH-hardness for degree-bounded* k*-coloring is typically referred to as folklore (see, e.g., Cygan et al. [22, Lemma 1]), but our results show that this property can be phrased in a purely algebraic manner.*

Theorem 13 also applies to many interesting classes of infinite-domain CSPs. For example, if we consider Γ such that each $R \in \Gamma$ has a first-order definition over the structure $(\mathbb{Q}; <)$, it is known that $\mathrm{CSP}(\Gamma)$ is NP-complete if and only if Γ can pp-interpret Γ^{3}_{SAT} [5, 8]. Hence, Theorem 13 is applicable, implying that if $\mathrm{CSP}(\Gamma)$ is not solvable in subexponential time if it is NP-complete, unless the ETH fails. More examples of infinite-domain CSPs where Theorem 13 is applicable include graph satisfiability problems [9] and phylogeny constraints [6]. We also remark that the intractable cases of the CSP dichotomy conjecture for infinite-domain CSPs over reducts of finitely bounded homogeneous structures are all based on pp-interpretability of Γ^{3}_{SAT} [3]. If this conjecture is correct, Theorem 13 and the ETH imply that none of these problems are solvable in subexponential time.

3.2 Defining Equality

For $\mathrm{CSP}(\Gamma)$-B our results are not as precise since we need the additional assumption that the equality relation is efpp-definable. This is not surprising since dichotomy results for CSPs are usually concerned with either constraint language restrictions [15, 47, 17, 14] or structural restrictions [23, 27], but rarely both simultaneously[1]. In the Boolean domain there are plenty of examples which illustrates how the equality relation may be efpp-defined [35, 21], suggesting that similar techniques may also exist for larger domains. In this section we consider two such techniques.

For finite domains we may in fact give a straightforward condition for when it is possible to efpp-define the equality relation. For this result we make use of a Galois connection between sets $\langle \Gamma \rangle_{\neq}$ closed under efpp-definitions and their dual objects, usually referred to as *hyperpolymorphisms* [13]. Let $\mathbf{P}(D)$ be the powerset of a set D. A k-ary function $f\colon D^k \rightarrow \mathbf{P}(D) \setminus \{\emptyset\}$ is said to be a *hyperoperation*, or *multifunction*, over D. If R is an n-ary relation over D then a k-ary hyperoperation f is said to preserve R if $f(t_1, \ldots, t_k) \subseteq R$ for each sequence of tuples $t_1, \ldots, t_k \in R$ (where $f(t_1, \ldots, t_k) = f(t_1[1], \ldots, t_k[1]) \times \ldots \times f(t_1[n], \ldots, t_k[n])$). Naturally, if the hyperoperation f preserves R then we say that f is a *hyperpolymorphism* of R. Similar to polymorphisms and partial

[1]However, a few examples of such hybrid restrictions exist, and we refer the reader to the surveys [19, 20] for concrete examples.

polymorphisms we then write $\mathrm{hPol}(\Gamma)$ for the set of all hyperpolymorphisms of a constraint language Γ. A hyperoperation $f\colon D^k \to \mathbf{P}(D)$ is *elementary* if $|f(x_1, \ldots, x_k)| = 1$ for all sequences of arguments $x_1, \ldots, x_k \in D$. Hence, an elementary hyperoperation is simply an ordinary operation in slight disguise. Hyperoperations and efpp-definitions can then be related by the following Galois connection.

Theorem 15 ([26, 37])**.** *Let* Γ *and* Δ *be constraint languages. Then* $\Gamma \subseteq \langle\Delta\rangle_{\neq}$ *if and only if* $\mathrm{hPol}(\Delta) \subseteq \mathrm{hPol}(\Gamma)$*.*

With these notions we can then characterise efpp-definitions of the equality relation as follows.

Theorem 16. *Let* Γ *be a constraint language over a finite domain* D*. Then* $\mathrm{Eq}_D \in \langle\Gamma\rangle_{\neq}$ *if and only if every* $f \in \mathrm{hPol}(\Gamma)$ *is elementary.*

Proof. Assume first that $\mathrm{Eq}_D \in \langle\Gamma\rangle_{\neq}$ and that there exists an n-ary $f \in \mathrm{hPol}(\Gamma)$ and $x \in D^n$ such that $f(x) = X$, $|X| > 1$. Let $t_1, \ldots, t_n \in \mathrm{Eq}_D$ such that $(t_1[1], \ldots, t_n[1]) = x$, and observe that this implies that $(t_1[2], \ldots, t_n[2]) = x$. It directly follows that $f(t_1, \ldots, t_n) = f(x) \times f(x) = X \times X$, and since $|X| > 1$, $X \times X \not\subseteq \mathrm{Eq}_D$.

For the other direction, assume that every hyperpolymorphism of Γ is elementary. Each $f \in \mathrm{hPol}(\Gamma)$ can then viewed as a polymorphism of Γ. It follows that each $f \in \mathrm{hPol}(\Gamma)$ preserves Eq_D, and Theorem 15 then implies that $\mathrm{Eq}_D \in \langle\Gamma\rangle_{\neq}$. $\square$

This holds, for example, if $\mathrm{Pol}(\Gamma)$ consists only of projections, since a non-elementary operation which preserves Γ can be used to construct an operation which is not a projection (see Proposition 3.2 in Romov [39]).

Let us consider an additional example showcasing how Eq_D can be efpp-defined for constraint languages over infinite domains D.

Lemma 17. *Assume that the relation* $R \subseteq D^k$ *is pp-definable in* Γ*. If*

$$\mathrm{Proj}_{\{i,j\}}(R) \not\subseteq \mathrm{Eq}_D$$

whenever $1 \leq i \neq j \leq k$*, then* R *is efpp-definable in* Γ*.*

Proof. Assume that R is pp-defined in Γ by

$$R(x_1, \ldots, x_k) \equiv \exists y_1, \ldots, y_m . \phi(x_1, \ldots, x_k, y_1, \ldots, y_m)$$

where ϕ is chosen to contain the minimal number of $\mathrm{Eq}_D(\cdot, \cdot)$ constraints. If $\mathrm{Eq}_D(x_i, x_j)$ is in ϕ for some $1 \leq i \neq j \leq k$, then $\mathrm{Proj}_{\{i,j\}}(R) \subseteq \mathrm{Eq}_D$ which leads to a contradiction. If $\mathrm{Eq}_D(y_i, y_j)$ is in ϕ for some $1 \leq i \neq j \leq m$, then we can replace y_i with y_j, remove $\mathrm{Eq}_D(y_i, y_j)$ and the existential quantification of y_i, and still have a pp-definition of R in Γ. Once again, this leads to a contradiction, since we assumed that ϕ was minimal with respect to the number of equality constraints. Finally, if $\mathrm{Eq}_D(x_i, y_j)$ is in ϕ, then we replace y_j with x_i and do the same modifications to the formula as above. Again, this contradicts the minimality of the number of equality constraints in ϕ, and we conclude that there exists an efpp-definition of R in Γ. $\square$

It is not difficult to find concrete examples where Lemma 17 is applicable. A straightforward class of constraint languages is *equality constraint languages* Γ where each $R \in \Gamma$ is the set of models of a first-order formula over a structure (D, Eq_D). Note that $\mathrm{Eq}_D \in \langle\Gamma\rangle_{\neq}$ does not hold a priori for an equality constraint language Γ over (D, Eq_D). However, as the following theorem shows, this does in fact hold whenever $\mathrm{CSP}(\Gamma)$ is NP-complete.

Theorem 18. *Let Γ be an equality constraint language over an infinite domain D. If Γ pp-interprets Γ^3_{SAT} (and thus, $\mathrm{CSP}(\Gamma)$ is NP-hard) then $\mathrm{Eq}_D \in \langle\Gamma\rangle_{\neq}$.*

Proof. It is known that the relation $\mathrm{Neq}_D = \{(x, y) \mid x, y \in D, x \neq y\}$ is pp-definable in Γ [7, Lemma 11] together with the relation

$$S(x_1, x_2, x_3) \equiv (x_1 = x_2 \land x_2 \neq x_3) \lor (x_1 \neq x_2 \land x_2 = x_3)$$

(see the remark preceding Lemma 14 in Bodirsky and Kára [7]). Clearly, Neq_D is not a subset of Eq_D, so Neq_D has an efpp-definition in Γ by Lemma 17. Arbitrarily choose two distinct elements d_1, d_2 in D. The following tuples are in S: $t_1 = (d_1, d_1, d_2)$ and $t_2 = (d_2, d_1, d_1)$. The tuple t_1 shows that $\mathrm{Proj}_{1,3}(S) \not\subseteq \mathrm{Eq}_D$ and $\mathrm{Proj}_{2,3}(S) \not\subseteq \mathrm{Eq}_D$ while t_2 shows that $\mathrm{Proj}_{1,2}(S) \not\subseteq \mathrm{Eq}_D$. It follows that S has an efpp-definition in Γ by Lemma 17. It is thus possible to efpp-define the relation Eq_D in Γ since $\mathrm{Eq}_D(x, y) \equiv \exists z.S(x, y, z) \land \mathrm{Neq}_D(y, z)$. $\square$

Moreover, for every equality constraint language Γ it is known that $\mathrm{CSP}(\Gamma)$ is NP-complete if and only if Γ pp-interprets Γ^3_{SAT} [7]. Combining Theorem 13 and Theorem 18 we can therefore claim the following corollary.

Corollary 19. *Let Γ be an equality constraint language over an infinite domain D. Then either*

1. $\mathrm{CSP}(\Gamma)$ *is tractable, or*
2. *there exists a B depending only on Γ such that $\mathrm{CSP}(\Gamma)$-B is NP-complete, but not solvable in subexponential time unless the ETH is false.*

4 The Easiest Ultraconservative CSP Problem

The results from Section 3, assuming the ETH, implies that $\mathsf{T}(\Gamma) > 0$ for any finite-domain and NP-complete $\mathrm{CSP}(\Gamma)$. However, it is safe to say that very little is known about the behaviour of the function T in more general terms. For example, if $\mathrm{CSP}(\Gamma)$ is NP-complete, is it always possible to find an NP-complete $\mathrm{CSP}(\Delta)$ such that $\mathsf{T}(\Delta) < \mathsf{T}(\Gamma)$? Such a scenario would be compatible with the consequences of Theorem 13. We will show that this is unlikely, and prove that for every finite domain D there exists a relation S_D such that $\mathrm{CSP}(S_D)$ is NP-complete but $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\Gamma)$ for any ultraconservative Γ over D such that $\mathrm{CSP}(\Gamma)$ is NP-complete. To prove this we have divided this section into two parts. In Section 4.1 we show that if Γ is ultraconservative and CSP(Γ) is NP-complete, then there exists a relation $R \in \langle\Gamma\rangle_{\nexists}$ which shares certain properties with the relation $R^{\neq\neq\neq 01}_{1/3}$. In Section 4.2 we use these relations to prove that for each finite domain D one can find a relation S_D such that CSP(S_D) is CV-reducible to any other NP-complete and ultraconservative CSP(Γ) problem over D. We also show that $\mathsf{T}(\{S_D\})$ tends to 0 for increasing values of $|D|$, and that the uniform D-CSP problem is not solvable strictly faster than $O(c^n)$ for any $c < |D|$ if the SETH is true.

4.1 $S_{\mathbb{B}}$-Extensions

The columns of the matrix representation of the relation $R^{\neq\neq\neq 01}_{1/3}$ from Jonsson et al. [35] (resulting in the easiest NP-complete SAT problem) enumerates all Boolean ternary tuples. We generalise this relation to arbitrary finite domains as follows.

Definition 20. *For each finite D let $S_D = \{t_1, t_2, t_3\}$ denote the $|D|^3$-ary relation such that for every $(d_1, d_2, d_3) \in D^3$ there exists $1 \leq i \leq |D|^3$ such that $(t_1[i], t_2[i], t_3[i]) = (d_1, d_2, d_3)$.*

Hence, similar to $R^{\neq\neq\neq 01}_{1/3}$, the columns of the matrix representation of S_D enumerates all ternary tuples over D. For each D the relation S_D is unique up to permutation of arguments, and although we will usually not be concerned with the exact ordering, we sometimes assume that $S_{\mathbb{B}} = R^{\neq\neq\neq 01}_{1/3}$ and that $\mathrm{Proj}_{1,\ldots,8}(S_D) = S_{\mathbb{B}}$. The notation S_D is a mnemonic for *saturated*, and the reason behind this will become evident in Section 4.2.1. For example, for $\{0, 1, 2\}$ we obtain a relation $\{t_1, t_2, t_3\}$ with 27 distinct arguments such that $(t_1[i], t_2[i], t_3[i]) \in \{0, 1, 2\}^3$ for each $1 \leq i \leq 27$. Jonsson et al. [35] proved that $S_{\mathbb{B}} \in \langle\Gamma\rangle_{\nexists}$ for every Boolean and idempotent constraint language Γ such that SAT(Γ) is NP-complete. This is not true for arbitrary finite domains, and in order to prove an analogous result we will need the following definition.

Definition 21. *Let R be an n-ary relation of cardinality 3 over a domain D, $|D| \geq 2$. If there exists $i_1, \ldots, i_8 \in \{1, \ldots, n\}$ such that*

$$\mathrm{Proj}_{i_1,\ldots,i_8}(R) = \{(a, a, b, b, b, a, a, b), (a, b, a, b, a, b, a, b), (b, a, a, a, b, b, a, b)\}$$

*for two distinct a and b, then we say that R is an $S_{\mathbb{B}}$-*extension*.*

For example, S_D is an $S_{\mathbb{B}}$-extension for every domain D. Note that CSP(R) is always NP-complete when R is an $S_{\mathbb{B}}$-extension. We will now prove that if CSP(Γ) is NP-complete and Γ is conservative, then Γ can pp-define an $S_{\mathbb{B}}$-extension. Note that for this result we do not require the stronger assumption that Γ is ultraconservative, which will be exploited in Section 5 where we study the conservative case in greater detail.

Lemma 22. *Let Γ be a conservative constraint language over a finite domain D such that* CSP(Γ) *is NP-complete. Then there exists a relation $R \in \langle\Gamma\rangle$ which is an $S_{\mathbb{B}}$-extension.*

Proof. Since CSP(Γ) is NP-complete and Γ is conservative, Theorem 12 implies that Γ can pp-interpret Γ^3_{SAT}, which in turn implies that Γ can pp-interpret every finite-domain relation. Therefore, let $f \colon F \to \mathbb{B}$, $F \subseteq D^d$ denote the parameters in the pp-interpretation of $S_{\mathbb{B}}$, and note that $f^{-1}(S_{\mathbb{B}}) \in \langle \Gamma \rangle$, but that $f^{-1}(S_{\mathbb{B}})$ is not necessarily an $S_{\mathbb{B}}$-extension since it could be the case that $|f^{-1}(S_{\mathbb{B}})| > 3$. Pick two tuples s_0 and s_1 in F such that (1) $f(s_0) = 0$ and $f(s_1) = 1$ and (2) s_0 and s_1 differ in the smallest possible set of coordinates. Such tuples must exist since f is surjective. Now consider the relation $F'(x_1, \ldots, x_d) \equiv F(x_1, \ldots, x_d) \wedge \{(s_0[1]), (s_1[1])\}(x_1) \wedge \ldots \wedge \{(s_0[d], s_1[d])\}(x_d)$. This relation is pp-definable over Γ since Γ is conservative and since $F \in \langle \Gamma \rangle$, and we claim that $F' = \{s_0, s_1\}$. Indeed, the existence of $u \in F'$ differing from both s_0 and s_1 would contradict the minimality of s_0 and s_1 since (1) $u[i] \in \{s_0[i], s_1[i]\}$ for each $1 \leq i \leq d$ and (2) if $u[i] \neq s_0[i]$ for some $1 \leq i \leq d$ then $u[i] = s_1[i]$, and u is strictly closer to s_1 than s_0 (the case when $u[i] \neq s_1[i]$ is symmetric).

Using the relation F' we can then pp-define the relation

$$\begin{aligned} R(x_{1,1}, \ldots, x_{1,d}, \ldots, x_{8,1}, \ldots, x_{8,d}) \equiv & f^{-1}(S_{\mathbb{B}})(x_{1,1}, \ldots, x_{1,d}, \ldots, x_{8,1}, \ldots, x_{8,d}) \wedge \\ & F'(x_{1,1}, \ldots, x_{1,d}) \wedge \ldots \wedge F'(x_{8,1}, \ldots, x_{8,d}). \end{aligned}$$

Clearly, if $(a_{1,1}, \ldots, a_{1,d}, \ldots, a_{8,1}, \ldots, a_{8,d}) \in R$, then $(a_{i,1}, \ldots, a_{i,d}) \in \{s_0, s_1\}$ for each $1 \leq i \leq 8$, and $(f(a_{1,1}, \ldots, a_{1,d}), \ldots, f(a_{8,1}, \ldots, a_{8,d})) \in S_{\mathbb{B}}$ if and only if $(a_{1,1}, \ldots, a_{1,d}, \ldots, a_{8,1}, \ldots, a_{8,d}) \in f^{-1}(S_{\mathbb{B}})$. Since $R \subseteq f^{-1}(S_{\mathbb{B}})$, this implies that

$$(f(a_{1,1}, \ldots, a_{1,d}), \ldots, f(a_{8,1}, \ldots, a_{8,d})) \in S_{\mathbb{B}}$$

if and only if $(a_{1,1}, \ldots, a_{1,d}, \ldots, a_{8,1}, \ldots, a_{8,d}) \in R$ and each $(a_{i,1}, \ldots, a_{i,d}) \in \{s_0, s_1\}$. In other words each element $f(a_{i,1}, \ldots, a_{i,d})$ in a tuple of $S_{\mathbb{B}}$ uniquely corresponds to d arguments $a_{i,1}, \ldots, a_{i,d}$ in the corresponding tuple of R, since $(a_{i,1}, \ldots, a_{i,d}) = s_0$ if $f(a_{i,1}, \ldots, a_{i,d}) = 0$, and $(a_{i,1}, \ldots, a_{i,d}) = s_1$ if $f(a_{i,1}, \ldots, a_{i,d}) = 1$. It follows that

$$R = \{s_0 \frown s_0 \frown s_1 \frown s_1 \frown s_1 \frown s_0 \frown s_0 \frown s_1, s_0 \frown s_1 \frown s_0 \frown s_1 \frown s_0 \frown s_1 \frown s_0 \frown s_1, s_1 \frown s_0 \frown s_0 \frown s_0 \frown s_1 \frown s_1 \frown s_0 \frown s_1\},$$

and since s_0 and s_1 differ in at least one position it is possible to find indices satisfying Definition 21, from which we conclude that R is an $S_{\mathbb{B}}$-extension. □

Observe that the existence of an $S_{\mathbb{B}}$-extension $R \in \langle \Gamma \rangle$ does not imply that CSP(R) $\leq^{\text{CV}}$ CSP(Γ). To accomplish this, we need to show that Γ can also qfpp-define an $S_{\mathbb{B}}$-extension. We will need the following lemma before we can present the proof for Lemma 24.

Lemma 23. *Let Γ be an ultraconservative language over a finite domain D and let $R \in \langle \Gamma \rangle$ be an n-ary relation such that $|R| = 2$. Then there exists $R' \in \langle \Gamma \rangle_{\nexists}$ such that (1) $|R'| = 2$ and (2)* $\text{Proj}_{1,\ldots,n}(R') = R$.

Proof. Let $R(x_1, \ldots, x_n) \equiv \exists y_1, y_2, \ldots, y_m . \varphi(x_1, \ldots, x_n, y_1, y_2, \ldots, y_m)$ denote a pp-definition of R over Γ, and let $R = \{t_1, t_2\}$. We will show that it is possible to remove the existentially quantified arguments $y_1, y_2, \ldots, y_m$ in this pp-definition by gradually adding new arguments to R. First consider the relation

$$R_1(x_1, \ldots, x_n, y_1) \equiv \exists y_2 \ldots, y_m.\varphi(x_1, \ldots, x_n, y_1, y_2, \ldots, y_m).$$

If $|R_1| = 2$ then we move on with the remaining arguments, so instead assume that $|R_1| > 2$. Now note that each tuple $t \in R_1$ in a natural way can be associated with either $t_1 \in R$ or $t_2 \in R$, depending on whether $t = t_1^\frown t'$ or $t = t_2^\frown t'$. Hence, let $S_1 = \{t[n+1] \mid t \in R_1, t_1^\frown t' = t\}$, and $S_2 = \{t[n+1] \mid t \in R_1, t_2^\frown t' = t\}$. In other words S_1 is the set of values taken by y_1 in the tuples corresponding to t_1, and S_2 the values taken by y_1 in the tuples corresponding to t_2. We consider two cases.

Case 1: $S_1 \cap S_2 = \emptyset$. Arbitrarily choose $d_1 \in S_1$ and $d_2 \in S_2$. Construct the relation $R_1'(x_1, \ldots, x_n, y_1) \equiv R_1(x_1, \ldots, x_n, y_1) \wedge \{(d_1), (d_2)\}(y_1)$, and note that $\{(d_1), (d_2)\} \in \Gamma$ since Γ is ultraconservative. We see that $R_1' = \{t_1^\frown(d_1), t_2^\frown(d_2)\}$.

Case 2: $S_1 \cap S_2 \neq \emptyset$. Arbitrarily choose $d \in S_1 \cap S_2$ and construct the relation $R_1'(x_1, \ldots, x_n, y_1) \equiv R_1(x_1, \ldots, x_n, y_1) \wedge R^d(y_1)$. We see that $R_1' = \{t_1^\frown(d), t_2^\frown(d)\}$. Note that we cannot choose elements as in Case 1 since if (for instance) one element is inside $S_1 \cap S_2$ and one element is outside $S_1 \cap S_2$, then the resulting relation will contain three tuples.

If we repeat this procedure for the remaining arguments $y_2, \ldots, y_m$ we will obtain a relation R' which is qfpp-definable over Γ such that $|R'| = 2$ and $\text{Proj}_{1,\ldots,n}(R') = R$. □

Lemma 24. *Let Γ be an ultraconservative constraint language over a finite domain D such that* $\text{CSP}(\Gamma)$ *is NP-complete. Then there exists a relation in $\langle\Gamma\rangle_{\nexists}$ which is an $S_{\mathbb{B}}$-extension.*

Proof. By Lemma 22 there exists a relation $R \in \langle\Gamma\rangle$ which is an $S_{\mathbb{B}}$-extension. Let

$$R(x_1, \ldots, x_n) \equiv \exists y_1, y_2, \ldots, y_m.\varphi(x_1, \ldots, x_n, y_1, \ldots, y_m)$$

denote its pp-definition over Γ. Using this pp-definition we will show that Γ can qfpp-define an $S_{\mathbb{B}}$-extension by gradually removing each existentially quantified variable. First consider the relation

$$R_1(x_1, \ldots, x_n, y_1) \equiv \exists y_2, \ldots, y_m.\varphi(x_1, \ldots, x_n, y_1, y_2, \ldots, y_m).$$

Assume that $|R_1| > 3$, i.e., that R_1 is not an $S_{\mathbb{B}}$-extension. Let $R = \{t_1, t_2, t_3\}$ and for each $1 \leq i \leq 3$ let $S_i = \{t[n+1] \mid t \in R_1, t_i^\frown t' = t\}$, $1 \leq i \leq 3$. In other words S_i contains the possible values taken by the argument y_1 in the tuples of R_1 corresponding to $t_i \in R$. There are now a few cases to consider depending on the sets S_1, S_2, S_3:

1. $|S_1 \cup S_2 \cup S_3| = 1$,
2. $|S_1 \cup S_2 \cup S_3| = 2$, and

3. $|S_1 \cup S_2 \cup S_3| \geq 3$,

The first case implies that the $(n+1)$th argument of R_1 is constant, i.e., that $|\{t[n+1] | t \in R_1\}| = 1$. But then $|R_1| = |R|$, contradicting our assumption that $|R_1| > 3$. Assume instead that $|S_1 \cup S_2 \cup S_3| \geq 3$ and that we are in the third case. First assume that the sets S_1, S_2, S_3 are pairwise disjoint, i.e., $S_1 \cap S_2 = S_1 \cap S_3 = S_2 \cap S_3 = \emptyset$. In this case we choose $d_1 \in S_1, d_2 \in S_2$, and $d_3 \in S_3$, and it is then easy to see (by basically reasoning in the same way as in the proof of Lemma 23) that $\exists y_2, \ldots, y_m. \{(d_1), (d_2), (d_3)\}(y_1) \wedge \varphi(x_1, \ldots, x_n, y_1, \ldots, y_m)$ defines an $S_{\mathbb{B}}$-extension. Otherwise, $S_i \cap S_j \neq \emptyset$ for distinct i and j. Choose $d_1 \in S_i \cap S_j$ and then an element d_2 from the remaining set S_k. Let $E = \{d_1, d_2\}$, and observe that $E \cap S_1 \neq \emptyset$, $E \cap S_2 \neq \emptyset$, and $E \cap S_3 \neq \emptyset$. Consider the relation $R'_1(x_1, \ldots, x_n, y) \equiv R_1(x_1, \ldots, x_n, y) \wedge E(y)$. Then $|R'_1| \geq 3$, $\mathrm{Proj}_{1,\ldots,n}(R'_1) = \{t_1, t_2, t_3\}$, and if we define $S'_i = \{t[n+1] \mid t \in R'_1, t_i \frown t' = t\}$ for each $1 \leq i \leq 3$, it is then clear that $|S'_1 \cup S'_2 \cup S'_3| = 2$, and that we may proceed as in the second case.

Now assume that $|S_1 \cup S_2 \cup S_3| = 2$ and let $\{d_1, d_2\} = S_1 \cup S_2 \cup S_3$. If $|S_1| = |S_2| = |S_3| = 1$ then $|R_1| = 3$, and R_1 is already an $S_{\mathbb{B}}$-extension; therefore we assume that S_1, S_2, or S_3 is equal to $\{d_1, d_2\}$. Up to symmetry, we then have the following possible cases:

1. $S_1 = S_2 = S_3 = \{d_1, d_2\}$,
2. $S_1 = S_2 = \{d_1, d_2\}$, $S_3 = \{d_1\}$,
3. $S_1 = \{d_1\}$, $S_2 = \{d_1\}$, $S_3 = \{d_1, d_2\}$, or
4. $S_1 = \{d_1\}$, $S_2 = \{d_2\}$, $S_3 = \{d_1, d_2\}$.

The first three cases are easy to handle: in all three cases, choose the element d_1 and construct the relation $R_1(x_1, \ldots, x_n, y) \wedge R^{d_1}(y)$. This leaves only the case when $S_1 = \{d_1\}$, $S_2 = \{d_2\}$ and that $S_3 = \{d_1, d_2\}$. Since R is an $S_{\mathbb{B}}$-extension there exists $a, b \in D$, $a \neq b$, and indices i_1, i_2, i_3 such that $(t_1[i_1], t_2[i_1], t_3[i_1]) = (b, b, a)$, $(t_1[i_2], t_2[i_2], t_3[i_2]) = (b, a, b)$, and $(t_1[i_3], t_2[i_3], t_3[i_3]) = (a, b, b)$. Define the binary relation F such that

$$F(x, y_1) \equiv \exists x_1, \ldots x_{i_3-1}, x_{i_3+1}, \ldots, x_n. R_1(x_1, \ldots, x_{i_3-1}, x, x_{i_3+1}, \ldots, x_n, y_1) \wedge R^b(x_{i_1}).$$

We claim that $F = \{(a, d_1), (b, d_2)\}$. To see this, observe that the constraint $R^b(x_{i_1})$ rules out the tuple t_3. This implies that if variable x_{i_3} has value a, then the variable y_1 must have value d_1 and if the variable x_{i_3} has value b, then the variable y_1 must have value d_2.

From this observation and Lemma 23, it follows that Γ can qfpp-define a relation F' such that $|F'| = 2$ and such that $\mathrm{Proj}_{1,2}(F') = F$. Let $k+2$ denote the arity of F' and define a relation

$$\begin{aligned} R'_1(x_1, \ldots, x_{i_3}, \ldots, x_n, y_1, z_1, \ldots, z_k) \equiv & R_1(x_1, \ldots, x_{i_3}, \ldots, x_n, y_1) \wedge \\ & F'(x_{i_3}, y_1, z_1, \ldots, z_k). \end{aligned}$$

We claim that R'_1 is an $S_{\mathbb{B}}$-extension. There are three possible ways of simultaneously choosing variables $x_{i_1}, x_{i_2}, x_{i_3}$. Let us consider the assignment

$(x_{i_1}, x_{i_2}, x_{i_3}) = (b, b, a)$. This particular choice gives all variables $x_1, \ldots, x_n$ fixed values (via the constraint $R_1(x_1, \ldots, x_{i_3}, \ldots, x_n, y_1)$). Furthermore, y_1 is assigned the value d_2 (via the constraint $F'(x_{i_3}, y_1, z_1, \ldots, z_k)$) and the variables $z_1, \ldots, z_k$ are given fixed values (since there is only one tuple in F' that allows y_1 to have the value d_2). Thus, there is only one tuple in R'_1 that allows $(x_{i_1}, x_{i_2}, x_{i_3}) = (b, b, a)$. The two other cases can be verified similarly and we conclude that $|R'_1| = 3$.

Finally, we see that there are $m - 1$ existentially quantified variables in the definition of R'_1 since F' can be qfpp-defined. By repeating the procedure outlined above for the remaining arguments we will obtain an $S_{\mathbb{B}}$-extension which is qfpp-definable over Γ. This concludes the proof. $\square$

Example 3. *Let us consider a concrete example of the sets S_1, S_2, S_3 from the proof of Lemma 24. Assume e.g. that $D = \{0, 1, 2\}$, and $S_1 \cup S_2 \cup S_3 = \{0, 1, 2\}$ where $S_1 = \{0, 1\}$, $S_2 = \{1, 2\}$, and $S_3 = \{0, 2\}$. The sets S_1, S_2, S_3 are not pairwise disjoint, so we e.g. choose $1 \in S_1 \cap S_2$ and $0 \in S_3$, and construct the relation $R'_1(x_1, \ldots, x_n, y) \equiv R_1(x_1, \ldots, x_n) \wedge \{(0), (1)\}(y)$. If we then let the sets S'_1, S'_2, S'_3 be the corresponding sets defined with respect to R'_1 it is then clear that $S'_1 \cup S'_2 \cup S'_3 = \{0, 1\}$ and that $|S'_1 \cup S'_2 \cup S'_3| = 2$. Since $S'_1 = \{0, 1\}$, $S'_2 = \{1\}$, and $S'_3 = \{0\}$, we end up in a case symmetric to the fourth case listed in the proof of Lemma 24. This is then handled by qfpp-defining a relation F' such that $|F'| = 2$ and such that $\mathrm{Proj}_{1,2}(F') = \{(a, 0), (b, 1)\}$ (for distinct $a, b \in D$) which can then be used to define an $S_{\mathbb{B}}$-extension.*

Note that in the proof of Lemma 24 we only require conservative relations E with $|E| \leq 3$. This observation is in line with the proof of the dichotomy theorem for conservative CSP(Γ) [14, p. 8] which is valid even if it is only assumed that Γ contains all unary E with $|E| \leq 3$.

4.2 Properties of and Reductions between $S_{\mathbb{B}}$-Extensions

By Lemma 24, we can completely concentrate on $S_{\mathbb{B}}$-extensions. We will prove that $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\Gamma)$ for every ultraconservative Γ over D such that CSP(Γ) is NP-complete. To prove this, we begin in Section 4.2.1 by investigating properties of $S_{\mathbb{B}}$-extensions, which we use to simplify the total number of distinct cases we need to consider. With the help of these results we develop techniques in Section 4.2.2 to show that $\mathrm{CSP}(S_D) \leq^{\mathrm{CV}} \mathrm{CSP}(R)$ for every $S_{\mathbb{B}}$-extension R over D.

4.2.1 Saturated $S_{\mathbb{B}}$-Extensions

In this section we reduce the number of cases we need to consider in Section 4.2.2. First note that if $R = \{t_1, t_2, t_3\}$ over D is a relation with $\mathrm{ar}(R) > |D|^3$ then there exists i and j such that $(t_1[i], t_2[i], t_3[i]) = (t_1[j], t_2[j], t_3[j])$. We say that the jth argument is *redundant*, and it is possible to get rid of this by identifying the ith and jth argument with the qfpp-definition

$$R'(x_1, \ldots, x_i, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n) \equiv R(x_1, \ldots, x_i, \ldots, x_{j-1}, x_i, x_{j+1}, \ldots, x_n).$$

This procedure can be repeated until no redundant arguments exist, and we will therefore always implicitly assume that $\mathrm{ar}(R) \leq |D|^3$ and that R has no redundant arguments. If R is an n-ary relation then the argument $i \in \{1, \ldots, n\}$ is said to be *k-choice*, $1 \leq k \leq |R|$, if $|\mathrm{Proj}_i(R)| = k$. Thus, if R is an $S_{\mathbb{B}}$-extension then each argument is either 1-choice, 2-choice, or 3-choice.

Definition 25. *An n-ary $S_{\mathbb{B}}$-extension $R = \{t_1, t_2, t_3\}$ is said to be* saturated *if there for each $1 \leq i \leq n$ and every function $\tau : \{1, 2, 3\} \to \{1, 2, 3\}$, exists $1 \leq j \leq n$ such that $(t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]) = (t_1[j], t_2[j], t_3[j])$.*

Example 4. *The relation S_D is saturated for every D, but if we consider the relations R and R' over $\{0, 1, 2\}$ defined by the matrices*

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 1 & 1 & 2 & 0 & 1 & 2 \end{pmatrix}$$

and

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 1 & 1 & 2 & 0 & 1 & 2 \end{pmatrix}$$

then neither relation is saturated. First, R is not saturated since its matrix representation, for example, does not contain the column $(0, 2, 0)$. Second, R' is not saturated since its matrix representation, for example, does not contain $(0, 2, 1)$ or $(0, 0, 2)$ as columns.

We now prove that we without loss of generality may assume that an $S_{\mathbb{B}}$-extension is saturated.

Lemma 26. *Let R be an $S_{\mathbb{B}}$-extension. Then there exists a saturated $S_{\mathbb{B}}$-extension $R' \in \langle R \rangle_{\nexists}$.*

Proof. Let D be the domain of R and let $a, b \in D$ be the two distinct domain values witnessing that R is an $S_{\mathbb{B}}$-extension, i.e.,

$$\mathrm{Proj}_{1,\ldots,8}(R) = \{(a, a, b, b, b, a, a, b), (a, b, a, b, a, b, a, b), (b, a, a, a, b, b, a, b)\}.$$

Let $R = \{t_1, t_2, t_3\}$ and let n denote the arity of R. For each $1 \leq i \leq n$ and each function $\tau : \{1, 2, 3\} \to \{1, 2, 3\}$ add a fresh argument taking the values $t_{\tau(1)}[i]$, $t_{\tau(2)}[i]$, $t_{\tau(3)}[i]$. Let R' be the resulting relation and let $R' = \{t'_1, t'_2, t'_3\}$ such that $\mathrm{Proj}_{1,\ldots,n}(t'_i) = t_i$. By construction, R' is a saturated $S_{\mathbb{B}}$-extension, but it remains to prove that $R' \in \langle R \rangle_{\nexists}$. Thus, let $1 \leq i \leq n$ be an index and τ a function $\tau : \{1, 2, 3\} \to \{1, 2, 3\}$. We will first show that R can qfpp-define a relation $S = \{s_1, s_2, s_3\}$ of arity $2n + 1$ where $\mathrm{Proj}_{1,\ldots,n}(S) = R$ and where $(s_1[2n + 1], s_2[2n + 1], s_3[2n + 1]) = (t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i])$. We have three distinct cases to consider based on the cardinality of $\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\}$.

Case 1: $|\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\}| = 1$. Then, $\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\} = \{c\}$ for a single element $c \in D$. Let $j, j' \in \{1, 2, 3\}$ such that $t[j] = t[j']$ if and only if $t = t_{\tau(1)}$ for $t \in R$. Then we define S as

$$\begin{aligned} S(x_1, \ldots, x_n, y_1, \ldots, y_n, z) \equiv\, & R(x_1, \ldots, x_n) \wedge R(y_1, \ldots, y_n) \wedge \\ & \mathrm{Eq}_D(y_j, y_{j'}) \wedge \mathrm{Eq}_D(y_i, z), \end{aligned}$$

which is correct since (1) $\mathrm{Proj}_{n+1,\ldots,2n}(S) = \{t_{\tau(1)}\}$ by the constraint $\mathrm{Eq}_D(y_j, y_{j'})$ and (2) $\mathrm{Proj}_{2n+1}(S) = \mathrm{Proj}_{n+i}(S) = \{(c)\}$ by the constraint $\mathrm{Eq}_D(y_i, z)$.
Case 2: $|\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\}| = 2$. Let $\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\} = \{c, d\}$, and assume without loss of generality that $t_{\tau(1)}[i] = c$ and that $t_{\tau(2)}[i] = d$ (hence, implying that $t_{\tau(3)}[i] = t_{\tau(1)}[i]$ or $t_{\tau(3)}[i] = t_{\tau(2)}[i]$). Let $t \in R \setminus \{t_{\tau(1)}, t_{\tau(2)}\}$ be the tuple in R which is distinct to both $t_{\tau(1)}$ and $t_{\tau(2)}$. Choose $i_1, i_2 \in [8]$ such that $t[i_1] \neq t[i_2]$, $t_{\tau(1)}[i_1] = t_{\tau(2)}[i_2]$, and $t_{\tau(2)}[i_1] = t_{\tau(2)}[i_2]$. This is possible since the 8 first arguments of R enumerate all ternary tuples over a 2-element subset of D, since R by assumption is an $S_{\mathbb{B}}$-extension. If we then define the relation

$$G(x_1, \ldots, x_n) \equiv R(x_1, \ldots, x_n) \wedge \mathrm{Eq}_D(x_{i_1}, x_{i_2})$$

it is clear that $G = \{t_{\tau(1)}, t_{\tau(2)}\}$ and that $\mathrm{Proj}_i(G) = \{(c), (d)\}$. Next, choose $j \in \{1, 2, 3\}$ such that $\mathrm{Proj}_{j,i}(G) = \{(a, c), (b, d)\}$, define the mapping $h(a) = c$, $h(b) = d$, and choose $i' \in [8]$ such that $h(t_1[i']) = t_{\tau(1)}[i]$, $h(t_2[i']) = t_{\tau(2)}[i]$, and $h(t_3[i']) = t_{\tau(3)}[i]$. This is possible since the sequence $t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]$ contains only the two distinct elements c and d, and since R is an $S_{\mathbb{B}}$-extension it is always possible to find an index in the first 8 positions which matches this sequence. For example, if $t_{\tau(1)}[i] = c, t_{\tau(2)}[i] = d, t_{\tau(3)}[i] = d$ then we choose $i' \in [8]$ such that $h(t_1[i']) = h(a) = c$, $h(t_2[i']) = h(b) = d$, and $h(t_3[i']) = h(b) = d$. Last, we define S as

$$\begin{aligned} S(x_1, \ldots, x_n, y_1, \ldots, y_n, z) \equiv\, & R(x_1, \ldots, x_n) \wedge G(y_1, \ldots, y_n) \wedge \\ & \mathrm{Eq}_D(x_{i'}, y_j) \wedge \mathrm{Eq}_D(y_i, z). \end{aligned}$$

Let $\{s_1, s_2, s_3\} = S$ be ordered such that $\mathrm{Proj}_{1,\ldots,8}(s_1) = \mathrm{Proj}_{1,\ldots,8}(t_1)$, $\mathrm{Proj}_{1,\ldots,8}(s_2) = \mathrm{Proj}_{1,\ldots,8}(t_2)$, and $\mathrm{Proj}_{1,\ldots,8}(s_3) = \mathrm{Proj}_{1,\ldots,8}(t_3)$. Then the constraint $\mathrm{Eq}_D(y_i, z)$ first enforces that $s_1[n+i] = s_1[2n+1]$, $s_2[n+i] = s_2[2n+1]$, $s_3[n+i] = s_3[2n+1]$, and the constraints $G(y_1, \ldots, y_n)$ and $\mathrm{Eq}_D(x_{i'}, y_j)$ then implies that $s_1[n+i] = t_{\tau(1)}(i)$, $s_2[n+i] = t_{\tau(2)}[i]$, and $s_3[n+i] = t_{\tau(3)}[i]$.
Case 3: $|\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\}| = 3$. Let $\{t_{\tau(1)}[i], t_{\tau(2)}[i], t_{\tau(3)}[i]\} = \{c, d, e\}$ for three distinct values c, d, e. Then we define the relation S as

$$\begin{aligned} S(x_1, \ldots, x_n, y_1, \ldots, y_n, z) \equiv\, & R(x_1, \ldots, x_n) \wedge R(y_1, \ldots, y_n) \wedge \\ & \mathrm{Eq}_D(y_1, x_{\tau(1)}) \wedge \mathrm{Eq}_D(y_2, x_{\tau(2)}) \wedge \\ & \mathrm{Eq}_D(y_3, x_{\tau(3)}) \wedge \mathrm{Eq}_D(y_i, z). \end{aligned}$$

Let $S = \{s_1, s_2, s_3\}$ be an enumeration such that $\mathrm{Proj}_{1,\ldots,8}(s_1) = \mathrm{Proj}_{1,\ldots,8}(t_1)$, $\mathrm{Proj}_{1,\ldots,8}(s_2) = \mathrm{Proj}_{1,\ldots,8}(t_2)$, and $\mathrm{Proj}_{1,\ldots,8}(s_3) = \mathrm{Proj}_{1,\ldots,8}(t_3)$. Consider, e.g., the tuple s_1. Then, due to the constraints $\mathrm{Eq}_D(y_1, x_{\tau(1)})$, $\mathrm{Eq}_D(y_2, x_{\tau(2)})$, and $\mathrm{Eq}_D(y_3, x_{\tau(3)})$ we have that $s_1[n+1] = t_{\tau(1)}[1], s_1[n+2] = t_{\tau(1)}[2]$, and $s_1[n+3] = t_{\tau(1)}[3]$, which implies that $\mathrm{Proj}_{n+1,\ldots,2n}(s_1) = t_{\tau(1)}$. Furthermore, due to the constraint $\mathrm{Eq}_D(z, y_i)$, we also have that $s_1[2n+1] = t_{\tau(1)}[i]$. Similarly, it holds that $s_2[2n+1] = t_{\tau(2)}[i]$ and $s_3[2n+1][i]$.

By repeating this for every $i \in [n]$ and every function $\tau\colon \{1,2,3\} \to \{1,2,3\}$ we will therefore be able to qfpp-define the saturated relation R', by gradually constructing relations S that are closer and closer to being saturated. Note that this procedure might introduce a large number of redundant arguments, but that these can easily be removed by simple variable identification. □

Example 5. *If R is the relation from Example 4 then a saturated relation in $\langle R \rangle_{\not\exists}$ from Lemma 26 can be given by*

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 2 & 2 & 2 & 0 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 2 & 0 & 2 & 0 & 2 & 0 & 1 & 2 \\ 1 & 0 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 0 & 2 & 2 & 0 & 1 & 2 \end{pmatrix}.$$

4.2.2 Reductions Between $S_{\mathbb{B}}$-Extensions

The main result of this section (Theorem 31 and Theorem 32) shows that $\mathsf{T}(\{S_D\}) = \mathsf{T}(\{S_D\} \cup 2^D) \leq \mathsf{T}(\Gamma)$ whenever Γ is an ultraconservative constraint language over D such that $\mathrm{CSP}(\Gamma)$ is NP-complete. The result is proved by a series of CV-reductions that we present in Lemmas 27–30. Before we begin, we note that if R is an $S_{\mathbb{B}}$-extension over D then $\{R\}$ is not necessarily a core. For a simple counterexample, $\{S_{\mathbb{B}}\}$ is not a core over $\{0,1,2\}$ since the endomorphism $e(0) = 0$, $e(1) = 1$, $e(2) = 0$, is not an automorphism. However, if R is an $S_{\mathbb{B}}$-extension and if $E = \{d_1, \ldots, d_m\}$ is the set $\bigcup_{1 \leq i \leq \mathrm{ar}(R)} \mathrm{Proj}_i(R)$, every endomorphism $e : E \to E$ of R must be an automorphism. Hence, Theorem 11 is applicable, and we conclude that $\mathrm{CSP}(\{R, R^{d_1}, \ldots, R^{d_m}\}) \leq^{\mathrm{CV}} \mathrm{CSP}(R)$. When working with reductions between $S_{\mathbb{B}}$-extensions we may therefore freely make use of constant relations. Given an instance (V, C) of $\mathrm{CSP}(R)$, where R is a relation, we say that a variable $x \in V$ occurring in a k-choice position in a constraint in C is a *k-choice variable.* We begin with the following simplifying lemma which allows us to bound the distribution of 3-choice variables in a very precise way.

Lemma 27. *Let R be a saturated $S_{\mathbb{B}}$-extension. Then there exists a CV-reduction f from $\mathrm{CSP}(R)$ to $\mathrm{CSP}(R)$ such that for every instance I of $\mathrm{CSP}(R)$, each variable in $f(I)$ occurs as a 3-choice variable in at most one constraint.*

Proof. Let n denote the arity of R and let $\{t_1, t_2, t_3\} = R$. Let $I = (V, C)$ be an instance of $\mathrm{CSP}(R)$. We will create an instance $I' = (V', C')$ of $\mathrm{CSP}(R)$ such that if $x \in V'$ is a 3-choice variable in a constraint then x does not occur as a 3-choice variable in any other constraint. Hence, let $x \in V$ be a 3-choice variable occurring in a constraint $c = R(x_1, \ldots, x_n)$ in position i_1. Assume that x also appears as a 3-choice variable in a constraint $c' = R(x'_1, \ldots, x'_n)$, distinct from c, in position i_2. Let $S = (t_1[i_1], t_2[i_1], t_3[i_1])$ and $S' = (t_1[i_2], t_2[i_2], t_3[i_2])$.

Assume first that $\text{Proj}_{i_1}(R) = \text{Proj}_{i_2}(R)$ (hence, $|\text{Proj}_{i_1}(R) \cap \text{Proj}_{i_2}(R)| = 3$). Define the function τ such that for each $1 \leq i \leq 3$, $\tau(S[i]) = j$ if and only if $t_j[i_2] = S[i]$ where $1 \leq j \leq 3$. Using the function τ we then define the permutation $\rho : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $\rho(i) = j$ if and only if $(t_1[i], t_2[i], t_3[i]) = (t_{\tau(1)}[j], t_{\tau(2)}[j], t_{\tau(3)}[j])$. This is indeed a well-defined permutation over $\{1, \ldots, n\}$ since R is saturated. Last, identify each variable $x'_{\tau(i)}$ occurring in c' with the variable x_i in c, and remove the constraint c'.

Second, assume that $|\text{Proj}_{i_1}(R) \cap \text{Proj}_{i_2}(R)| = 2$, and let $\text{Proj}_{i_1}(R) \cap \text{Proj}_{i_2}(R) = \{d, d'\}$. Assume without loss of generality that $t_1[i_1] = d$, $t_2[i_1] = d'$, and that $t_3[i_1] \notin \{d, d'\}$. Choose $i \in \{1, \ldots, n\}$, distinct from both i_1 and i_2, such that $t_1[i] = t_1[i_1]$, $t_2[i] = t_2[i_1]$, and $t_3[i] \neq t_3[i_1]$. Such an i must exist since R is saturated. Then identify x with x_i. Define the function τ such that for $1 \leq i \leq 2$, $\tau(S[i]) = j$ if and only if $t_j[i_2] = S[i]$. Using the function τ we then define the permutation $\rho : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $\rho(i) = j$ if and only if $(t_1[i], t_2[i]) = (t_{\tau(1)}[j], t_{\tau(2)}[j])$. Clearly, ρ is a well-defined permutation over $\{1, \ldots, n\}$ since R is saturated. Last, identify each variable $x'_{\tau(i)}$ occurring in c' with the variable x_i in c, and remove the constraint c'. The case when $|\text{Proj}_{i_1}(R) \cap \text{Proj}_{i_2}(R)| = 1$, i.e., when x is assigned the same value in any satisfying assignment, is very similar.

Each time this procedure is performed, at least one constraint is removed. Thus, we let I' denote the fixpoint that we will reach in at most $|C|$ iterations. It is not difficult to verify that I is satisfiable if and only if I' is satisfiable. Furthermore, $|V'| \leq |V|$ and the reduction can be computed in polynomial time. We have thus showed that the reduction is a CV-reduction and have therefore proved the lemma. □

Next, investigate how the removal, or rather, the addition, of 3-choice arguments affect the fine-grained complexity of CSPs.

Lemma 28. *Let R be a saturated $S_{\mathbb{B}}$-extension and let R' be R with one or more 3-choice arguments removed, such that R' is still saturated. Then* $\text{CSP}(R) \leq^{\text{CV}} \text{CSP}(R')$.

Proof. Let $n = \text{ar}(R)$, $n' = \text{ar}(R')$, $R = \{t_1, t_2, t_3\}$, and assume that $\text{Proj}_{1,\ldots,n'}(R) = R'$. Let $I = (V, C)$ be an instance of $\text{CSP}(R)$. First apply Lemma 27 in order to obtain an instance $I_1 = (V_1, C_1)$ of $\text{CSP}(R)$ such that each 3-choice variable only occurs in a 3-choice position in a single constraint. Assume there exists $x \in V_1$ and two distinct constraints $c, c' \in C_1$ such that x occurs in position $i \in \{n'+1, \ldots, n\}$ in c and in a 1- or 2-choice position $j \in \{1, \ldots, n\}$ in c'. Let $S = \text{Proj}_i(R) \cap \text{Proj}_j(R)$, and note that $|S| \leq 2$. Assume first that $|S| = 2$, let $S = \{d_1, d_2\}$, and assume without loss of generality that $t_1[i] = t_1[j] = d_1$, $t_2[i] = t_2[j] = d_2$, and that $t_3[i] \neq t_3[j]$ (the other cases can be treated similarly). Since R is saturated there exists a 2-choice argument $i' \in \{1, \ldots, n\}$ such that $t_1[i'] = t_1[i] = t_1[j]$, $t_2[i'] = t_2[i] = t_2[j]$, and such that $t_3[i'] \neq t_3[i]$. Let y be the variable occurring in the i'th position of c. Create a fresh variable $\hat{x}$, replace x in position i with $\hat{x}$, and for each constraint where x occurs as a 1- or 2-choice variable, replace x with y. Repeat this procedure until every

3-choice variable occurring in position $n'+1, \ldots, n$ only occurs in a single constraint, and let $I_2 = (V_2, C_2)$ be the resulting instance. Hence, with the help of the variables $\hat{x}$ we may assume that any variable occurring in a 3-choice position not present in R', only occurs in a single constraint. We will shortly also see that these variables can safely be removed from the instance.

Assume there exists $x \in V_2$ and a constraint $c \in C_2$ such that x occurs as a 3-choice variable in position $i \in \{n'+1, \ldots, n\}$ and also in a position $j \in \{1, \ldots, n\}$, $i \neq j$, in c. Let $L = \{t_r \mid 1 \leq r \leq 3, t_r[i] = t_r[j]\}$. Since R does not have any redundant arguments it must be the case that $|L| < 3$. If $|L| = 0$ then the instance is unsatisfiable, in which case we output an arbitrary unsatisfiable instance, and if $|L| = 1$ it is easy to see that any variable occurring in c can be assigned a fixed value, and the constraint may be removed. Therefore, assume that $|L| = 2$, and e.g. that $L = \{t_1, t_2\}$. Since R is saturated there exists a 2-choice argument $j' \in \{1, \ldots, n\}$ such that $t_1[j'] = t_2[j'] \neq t_3[j']$. Let y be the variable occurring in position j' in c and add the constraint $R^{t_1[j']}(y)$. Repeat this for every variable occurring in position $n'+1, \ldots, n$ in a constraint in C_2, and then replace each constraint $R(x_1, \ldots, x'_n, \ldots, x_n)$ by $R'(x_1, \ldots, x_n)$. Note that any variable $\hat{x}$ introduced in the previous step of this reduction is removed in this transformation, and without affecting the satisfiability of the instance since each $\hat{x}$ variable occur in a unique constraint, and in a unique 3-choice position. Hence, the reduction is a CV-reduction. $\square$

Moreover, for saturated relations, the addition of 2-choice arguments does not affect the fine-grained complexity.

Lemma 29. *Let R be a saturated $S_{\mathbb{B}}$-extension and let R' be an $S_{\mathbb{B}}$-extension obtained by adding additional 2-choice arguments to R. Then* $\mathrm{CSP}(R') \leq^{\mathrm{CV}} \mathrm{CSP}(R)$.

Proof. Let $n = \mathrm{ar}(R)$, $n' = \mathrm{ar}(R')$, and $R' = \{t'_1, t'_2, t'_3\}$. By the statement of the lemma we may assume that $\mathrm{Proj}_{1,\ldots,n}(R') = R$, and that $|\mathrm{Proj}_i(R')| = 2$ for every $n < i \leq n'$. Furthermore, since R is assumed to be saturated, we may also assume that each fresh argument $n < i \leq n'$ satisfies $\mathrm{Proj}_i(R') = \{a, b\}$ for some b which does not occur in any of the original n arguments (otherwise i is redundant). Thus, we may also assume that $\mathrm{Proj}_i(R')$ for every $n < i \leq n'$ is distinct from $\mathrm{Proj}_j(R')$ for every $1 \leq j \leq n$. To simplify the presentation we also assume that $\mathrm{Proj}_{1,\ldots,8}(R') = S_{\mathbb{B}}$. Let $I = (V, C)$ be an instance of $\mathrm{CSP}(R')$. Our aim is to construct an instance of $\mathrm{CSP}(R')$ where any variable which occurs as a 2-choice variable in a constraint in position $n < i \leq n'$, does not occur in any other constraint or in any other position within the constraint. Then we can easily create a $\mathrm{CSP}(R)$ instance by replacing each constraint over R' by the corresponding constraint over R, and in the process removing any variable which occurs in position $n < i \leq n'$.

Let x be a variable that appears in (at least) two distinct constraints $c_1, c_2 \in C$. Assume that x occurs at position $n+1 \leq i \leq n'$ in c_1 and at position $1 \leq j \leq n'$ in c_2. We consider a number of cases based on the cardinality of $S = \mathrm{Proj}_i(R') \cap \mathrm{Proj}_j(R')$.

- $|S| = 3$. This is not possible since $|\mathrm{Proj}_i(R')| = 2$.
- $|S| = 2$. Assume that $S = \{a, b\}$ for two distinct elements a and b, and observe that this implies that $\mathrm{Proj}_i(R') = \mathrm{Proj}_j(R') = \{a, b\}$, since $|\mathrm{Proj}_j(R')| = 3$ cannot happen. Define $f : \{a, b\} \to \{0, 1\}$ such that $g(a) = 0$ and $g(b) = 1$. It follows that there exist indices $l, m \in \{1, \dots, 8\}$ such that $f(t'_r[i]) = t'_r[l]$ and $g(t'_r[j]) = t'_r[m]$ when $r \in \{1, 2, 3\}$. Now, let y be the variable at position l in c_1 and let z be the variable at position m in c_2. Then we identify z with y, introduce a fresh variable $\hat{x}$, and replace x at the ith position of c_1 with $\hat{x}$.
- $|S| = 1$. Assume $S = \{a\}$, $\mathrm{Proj}_i(R') = \{a, b\}$ (where a, b are distinct elements), and $\mathrm{Proj}_j(R') = \{a, d, d'\}$ (where a, d, d' are not necessarily distinct). Define $f : \{a, b\} \to \{0, 1\}$ such that $f(a) = 0$ and $f(b) = 1$, and $g : \{a, d, d'\} \to \{0, 1\}$ such that $g(a) = 0$ and $g(x) = 1$ if $x \neq a$. It is not hard to see that there exists $l, m \in \{1, \dots, 8\}$ such that $f(t'_r[i]) = t'_r[l]$ and $g(t'_r[j]) = t'_r[m]$ when $r \in \{1, 2, 3\}$. Let y be the variable at position l in c_1 and z be the variable at position m in c_2. Add the unary relations $R^0(y)$ and $R^0(z)$, introduce a new variable $\hat{x}$, and replace x at the ith position of c_1 with $\hat{x}$.
- $|S| = 0$. This implies that I_1 is unsatisfiable, and we simply output an arbitrary unsatisfiable instance.

By repeating the procedure above until a fixpoint is reached, we will obtain an instance $I_1 = (V_1, C_1)$ such that if $x \in V_1$ and if x appears in a constraint $c \in C_1$ at position $n+1, \dots, n'$, then it does not appear in any other constraint. Note that if a variable x occurs in more than two constraints then this procedure will gradually lower the number of occurrences of x by picking two distinct constraints at a time. However, it is still possible that $x \in V_1$ appear more than once in a single constraint $c \in C_1$ where (at least) one of the occurrences of x is at position $n+1, \dots, n'$. Therefore, assume that x appears in positions i and j in $c \in C_1$ where $i \in \{n+1, \dots n'\}$ and $j \in \{1, \dots n'\}$. Let $L \subseteq \{1, 2, 3\}$ denote the set $\{l \mid t'_l[i] = t'_l[j]\}$.

- $|L| = 3$. This is not possible since there are no redundant arguments in the relation R'.
- $|L| = 2$. Assume (without loss of generality) that $t'_1[i] = t'_1[j]$, $t'_2[i] = t'_2[j]$, and $t'_3[i] \neq t'_3[j]$. Pick $k \in \{1, \dots, 8\}$ such that $t'_1[k] = t'_2[k] \neq t'_3[k]$. Let y be the variable that appear in the kth position in c. Add a unary constraint $R^{t'_1[k]}(y)$, introduce a fresh variable $\hat{x}$, and replace the x at position i in c with $\hat{x}$.
- $|L| = 1$. Without loss of generality we can assume that $t'_1[i] = t'_1[j]$. For each variable y occurring in the lth position in c add the unary constraint $R^{t'_1[l]}(y)$, and then remove the constraint c.
- $|L| = 0$. This implies that I_1 is unsatisfiable, and we simply output an arbitrary unsatisfiable instance.

Repeat the procedure above until a fixpoint is reached and let $I_2 = (V_2, C_2)$ be the resulting instance. Observe that a variable x that occurs in a constraint at position $n+1, \ldots, n'$ only occur in a single constraint and in a unique position. Finally, let $I_3 = (V_3, C_3)$ be the instance of $\mathrm{CSP}(R)$ obtained by replacing each constraint $R'(x_1, \ldots, x_n, x_{n+1}, \ldots, x_{n'}) \in C_2$ by $R(x_1, \ldots, x_n)$. Note that every fresh variable $\hat{x}$ that were introduced in the previous steps are removed in the conversion of I_2 into I_3. This shows that the reduction is indeed a CV-reduction. □

Lemma 30. *Let R be a saturated $S_{\mathbb{B}}$-extension over D with at least one 3-choice argument. Then* $\mathrm{CSP}(S_D) \leq^{\mathrm{CV}} \mathrm{CSP}(R)$.

Proof. Let $n = \mathrm{ar}(R)$. Choose three distinct values $d_1, d_2, d_3 \in D$ such that there does not exist any i such that $\mathrm{Proj}_i(R) = \{d_1, d_2, d_3\}$. If no such i exists, then R is saturated and already contains all possible 3-choice arguments over D, implying (1) that $\langle R \rangle_{\nexists} = \langle S_D \rangle_{\nexists}$ and (2) by Theorem 8 and Theorem 10, that we are done. Hence, we assume that such d_1, d_2, d_3 exist. First, construct the relation S (with domain D) such that $\mathrm{Proj}_{1,\ldots,n}(S) = R$, $\mathrm{Proj}_{n+1}(S) = \{d_1, d_2, d_3\}$, and then add the minimum number of arguments to make S saturated. Second, let S' be the relation obtained from S by projecting away every argument i of the form $\mathrm{Proj}_i(S) = \{d_1, d_2, d_3\}$. Thus, if we let $i_1, \ldots, i_m \in [\mathrm{ar}(S)]$ be the set of indices such that $\mathrm{Proj}_{i_j}(S) \neq \{d_1, d_2, d_3\}$, $1 \leq j \leq m$, then $S' = \mathrm{Proj}_{i_1,\ldots,i_m}(S)$. In other words, S' is equal to R in the first n coordinates, but potentially contains more 1-choice and 2-choice arguments. Note that S' is saturated. Via Lemma 29, and the observation that 1-choice arguments do not affect the complexity of $\mathrm{CSP}(S')$, it then follows that $\mathrm{CSP}(S') \leq^{\mathrm{CV}} \mathrm{CSP}(R)$. An application of Lemma 28 then gives the desired result that $\mathrm{CSP}(S) \leq^{\mathrm{CV}} \mathrm{CSP}(S') \leq^{\mathrm{CV}} \mathrm{CSP}(R)$. Since d_1, d_2, d_3 were arbitrarily chosen and since S is saturated, it is clear that we can repeat the above argument by choosing three fresh values from D, and extending S. Eventually, the resulting relation is saturated and will contain all possible 3-choice arguments, and can therefore qfpp-define S_D. Hence, $\mathrm{CSP}(S_D) \leq^{\mathrm{CV}} \mathrm{CSP}(R)$. □

We have thus proved the main result of this section.

Theorem 31. *Let D be a finite domain and let Γ be a finite, ultraconservative constraint language over D. If* $\mathrm{CSP}(\Gamma)$ *is NP-complete then* $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\Gamma)$.

Proof. We first show that if R is an $S_{\mathbb{B}}$-extension over a finite domain D, then $\mathrm{CSP}(S_D) \leq^{\mathrm{CV}} \mathrm{CSP}(R)$. By Lemma 26 we may assume that R is saturated. If R does not contain any 3-choice arguments we use Lemma 28 together with Lemma 29 and obtain a CV-reduction from $\mathrm{CSP}(S_D)$ to $\mathrm{CSP}(R)$. Hence, assume that R contains one or more 3-choice arguments. In this case we use Lemma 30 and obtain a CV-reduction from $\mathrm{CSP}(S_D)$ to $\mathrm{CSP}(R)$.

By Lemma 24 there exists an $S_{\mathbb{B}}$-extension $R \in \langle \Gamma \rangle_{\nexists}$, implying that $\mathrm{CSP}(R) \leq^{\mathrm{CV}}$ $\mathrm{CSP}(\Gamma)$ via Theorem 8 and Theorem 10, and we know that $\mathrm{CSP}(S_D) \leq^{\mathrm{CV}}$ $\mathrm{CSP}(R)$. We conclude that $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\{R\}) \leq \mathsf{T}(\Gamma)$. □

4.2.3 Consequences For Finite-Domain CSPs

In this section we wish to make a few additional remarks concerning the time complexity of finite-domain CSP(Γ), using Theorem 31 as a starting point. Clearly, $\{S_D\}$ is not an ultraconservative constraint language, but it is possible to prove that the complexity of CSP(S_D) does not change when we expand the language by adding all unary relations over D.

Lemma 32. *Let D be a finite domain. Then $\mathsf{T}(\{S_D\}) = \mathsf{T}(\{S_D\} \cup 2^D)$.*

Proof. $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\{S_D\} \cup 2^D)$ holds trivially. To prove $\mathsf{T}(\{S_D\} \cup 2^D) \leq \mathsf{T}(\{S_D\})$ we show that CSP($\{S_D\} \cup 2^D$) $\leq^{\mathrm{CV}}$ CSP(S_D). Since we have already seen many reductions akin to this we only provide a sketch. Let (V, C) be an instance of CSP($\{S_D\} \cup 2^D$). Assume $x \in V$ appears in a unary constraint $E(x) \in C$. If x also appears in another unary constraint $E'(x)$ then these two constraints can be replaced by $(E \cap E')(x)$; hence, we may assume that each variable occurs in at most one unary constraint. If x does not occur in any other constraint, then we first check if $E = \emptyset$. If this is the case, the instance is unsatisfiable and we abort the procedure, and otherwise we simply remove the constraint $E(x)$. Now assume that x also appears in the ith position in a constraint $S_D(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_{\mathrm{ar}(S_D)})$. If $E \cap \mathrm{Proj}_i(S_D) = \emptyset$ then the instance is unsatisfiable, and if $E \supseteq \mathrm{Proj}_i(S_D)$ then we may safely remove the constraint E. Therefore assume that either $|\mathrm{Proj}_i(S_D) \cap E| = 1$ or that $|\mathrm{Proj}_i(S_D) \cap E| = 2$. The first of these cases is easy to handle since it implies that x is forced a constant value in any satisfying assignment, which can be handled by identifying x with the corresponding 1-choice variable in $S_D(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_{\mathrm{ar}(S_D)})$. The second case implies that x appears in a 2- or 3-choice position, i.e., $\mathrm{Proj}_i(S_D) = \{d_1, d_2, d_3\}$, for three distinct values d_1, d_2, and d_3, or $\mathrm{Proj}_i(S_D) = \{d_1, d_2\}$ for two distinct values d_1 and d_2. For simplicity, we assume that x appears in a 3-choice position, since the 2-choice case is similar. Assume that $E \supseteq \{(d_1), (d_2)\}$, and let $t \in S_D$ be the tuple satisfying $t[i] = d_3$. Let $\{s, u\} = S_D \setminus \{t\}$ and choose j such that $s[j] = s[i]$, $u[i] = u[j]$, and $t[j] \in \{s[j], u[j]\}$. Then identify x with the variable x_j throughout the instance. If we repeat this procedure for the remaining constraints containing x, remove the constraint $E(x)$, and then continue with all remaining unary constraints, we will obtain an instance of CSP(S_D) which is satisfiable if and only if (V, C) is satisfiable. Furthermore, since D is fixed and finite, this procedure can be carried out in polynomial time with respect to $|V|$, showing that CSP($\{S_D\} \cup 2^D$) $\leq^{\mathrm{CV}}$ CSP(S_D). □

Thus, no NP-complete CSP over an ultraconservative constraint language over D is solvable strictly faster than CSP(S_D), and, in particular, $\mathsf{T}(\{S_{D'}\}) \leq \mathsf{T}(\{S_D\})$ whenever $D' \supseteq D$. This raises the question of whether $\mathsf{T}(S_D) = \mathsf{T}(S_{D'})$ for all $D, D' \supseteq \{0, 1\}$, or if it is possible to find D and D' such that $\mathsf{T}(\{S_{D'}\}) < \mathsf{T}(\{S_D\})$. As the following theorem shows, this is indeed the case, unless $\mathsf{T}(\{S_D\}) = 0$ for every finite D and the ETH fails.

Theorem 33. *The infimum of* $\{\mathsf{T}(\{S_D\} \cup 2^D) \mid D$ *is finite,* $|D| \geq 2\}$ *equals 0.*

Proof. According to Theorem 32, $\mathsf{T}(\{S_D\}) = \mathsf{T}(\{S_D\} \cup 2^D)$, so as a simplification we will instead prove that $\inf\{\mathsf{T}(\{S_D\}) \mid D$ finite and $|D| \geq 2\} = 0$.

Let $D_k = \{0, \ldots, k-1\}$, $k \geq 5$. We will analyse a simple algorithm for CSP(S_{D_k}). Let $I = (V, C)$ be an arbitrary instance of CSP(S_{D_k}). Extend the instance with fresh variables $Z = \{z_0, \ldots, z_{k-1}\}$ and the constraints $R^i(z_i)$, $0 \leq i \leq k-1$. These variables will be used to force variables to take constant values when branching over the constraints in the instance. Arbitrarily choose a constraint $c = S_{D_k}(x_1, \ldots, x_{k^3})$ and let $X = \{x_1, \ldots, x_{k^3}\}$. Since each tuple in S_D contains exactly k^2 occurrences of every domain element in D, it follows that if a variable x appears in k^2+1 or more positions, then the constraint c cannot be satisfied. Thus, $|X| \geq k$.

First, assume that $X \cap Z = \emptyset$. Then, we branch on the three tuples in S_D, by for each tuple $s \in S_D$ and $1 \leq i \leq k^3$, identifying x_i with $z_{s[i]}$ and removing the constraint c. Hence, in each branch at least k variables are removed.

Second, assume to the contrary that $X \cap Z \neq \emptyset$. If a variable $z \in Z$ occurs in a 3-choice position, then the variables in $X \setminus Z$ can be assigned fixed values and no branching is needed. If no variable $z \in Z$ occurs in a 3-choice position, then all $k(k-1)(k-2)$ 3-choice positions in S_{D_k} are covered by variables outside Z. Thus, we perform three branches based on the tuples in S_{D_k}, and similarly to the case when $X \cap Z = \emptyset$ identify variables with $z_0, \ldots, z_{k-1}$ as prescribed by the chosen tuple. Recall that a variable can occur in at most k^2 positions in the constraint c, since c is otherwise not satisfiable. This implies that at least $\lfloor \frac{k(k-1)(k-2)}{k^2} \rfloor \geq 1$ variables are given fixed values in each branch.

When there are no S_{D_k} constraints left, we check whether the remaining set of unary constraints are satisfiable or not. It is straightforward to perform this test in polynomial time. A recursive equation that gives an upper bound on the time complexity of this algorithm is thus

$$T(1) = poly(||I||), T(n) = 3T\left(n - \lfloor \frac{k(k-1)(k-2)}{k^2} \rfloor\right) + poly(||I||)),$$

(where n denotes the number of variables and $||I||$ the number of bits required to represent I) so

$$T(n) \in O(3^{n \cdot \frac{k^2}{k(k-1)(k-2)}} \cdot poly(||I||)).$$

The function $\frac{k^2}{k(k-1)(k-2)}$ obviously tends to 0 with increasing k so the infimum of the set $\{\mathsf{T}(\{S_D\}) \mid D$ is finite and $|D| \geq 2\}$ is equal to 0. □

We may summarise the results obtained thus far as follows. For each finite domain D there exists a unique lower bound on $\mathsf{T}(\Gamma)$ for ultraconservative Γ, namely $\mathsf{T}(\{S_D\}) = \mathsf{T}(\{S_D\} \cup 2^D)$. Furthermore, unless the ETH fails, this value is strictly greater than 0, but tends to 0 for increasing values of $|D|$.

A related question is whether it is also possible to find a "hardest" NP-complete CSP over each domain D, i.e., a Γ such that $\mathsf{T}(\Gamma) \geq \mathsf{T}(\Delta)$ for each Δ over D. This question is in a sense trivial since D-CSP can be phrased as a CSP over the set of all relations over D, which admits a CV-reduction from every other CSP problem over the domain. However, we may in fact also prove that D-CSP is not solvable in $O(c^n)$ time for *any* $c < |D|$, assuming the SETH (i.e., that the limit of the sequence $\mathsf{T}(\Gamma^3_{\text{SAT}})$, $\mathsf{T}(\Gamma^4_{\text{SAT}}), \ldots$ tends to 1). This is a straightforward consequence of a reduction from Lagerkvist [36], but to the best of our knowledge, this connection between SETH and the complexity of D-CSP has not explicitly been stated in the literature before.

Theorem 34. *Let D be a finite domain. Then D-CSP is not solvable in $O(c^n)$ time for any $c < |D|$ unless the SETH is false.*

Proof. In Lagerkvist [36, Corollary 19] it is proved that D-CSP is not solvable in $O(c^n)$ time for any $c < |D|$ if $\{0,1\}$-CSP is not solvable in $O(d^n)$ time for any $d < 2$. Thus, we only need to prove that a $O(d^n)$ time algorithm, $d < 2$, for $\{0,1\}$-CSP is sufficient to contradict the SETH. Hence, assume that $\{0,1\}$-CSP is solvable in $O(d^n)$ time for some $d < 2$. Since $\lim_{k \to \infty} \mathsf{T}(\Gamma^k_{\text{SAT}}) = 1$ by the SETH it follows that there exists k such that $\text{SAT}(\Gamma^k_{\text{SAT}})$ is not solvable in $O(d^n)$ time. But since there is a trivial CV-reduction from $\text{SAT}(\Gamma^k_{\text{SAT}})$ to $\{0,1\}$-CSP, this contradicts the assumption that $\{0,1\}$-CSP is solvable in $O(d^n)$ time. We conclude that D-CSP is not solvable in $O(c^n)$ time for any $c < |D|$. $\square$

5 The Conservative Case

Lemma 24 (and as a consequence, Theorem 31) is only valid for ultraconservative constraint languages. In this section we develop techniques that will be useful when Γ is conservative but not necessarily ultraconservative. For the ternary domain $\{0,1,2\}$ these techniques are sufficient to prove that $\text{CSP}(S_{\{0,1,2\}})$ results in the easiest NP-complete conservative CSP problem over $\{0,1,2\}$, but for larger domains the situation appears to be more complex. We begin with the following easy lemma, which provides a shortcut if the constant relations over D are not qfpp-definable over a conservative constraint language Γ.

Lemma 35. *Let Γ be a conservative constraint language over a finite domain $D = \{d_1, \ldots, d_k\} \subseteq \mathbb{N}$. Then, for each $d_i \in D$, there exists a k-ary relation $R \in \langle \Gamma \rangle_{\nexists}$ such that $\text{Proj}_j(R) = R^{d_i}$ for some $j \in \{1, \ldots, k\}$.*

Proof. Note first that every endomorphism of Γ is idempotent, i.e., it is simply the unary projection. It is then well-known that Γ can qfpp-define the relation $\{(e(d_1), \ldots, e(d_k)) \mid e \text{ is an endomorphism of } \Gamma\}$ (see e.g. Theorem 3.6 in Barto [2]). Hence, Γ can qfpp-define the relation $\{(d_1, \ldots, d_k)\}$, which satisfies the claim of the lemma. $\square$

We will now define a useful relation which is guaranteed to be qfpp-definable by any constraint language Γ. This relation will then be used as a gadget when constructing qfpp-definitions of certain conservative relations.

Definition 36. *Let Γ be a constraint language over a finite domain $D = \{d_1, \ldots, d_k\}$. Let $t_1, \ldots, t_{k^2} \in D^2$ be a lexicographical enumeration of all binary tuples over D. We define the relation B^{Γ}_D as*

$$B^{\Gamma}_D = \{(f(t_1), \ldots, f(t_{k^2})) \mid f \in \mathrm{Pol}(\Gamma), \mathrm{ar}(f) = 2\}.$$

The intuitive reasoning behind the relation B^{Γ}_D is that each tuple can be viewed as a representation of a binary function in $\mathrm{Pol}(\Gamma)$. Even better, B^{Γ}_D is known to be qfpp-definable by Γ for any contraint language Γ over a finite domain (for proof, see e.g., Theorem 4.1 in Jeavons et al. [33])

Lemma 37. *$B^{\Gamma}_D \in \langle\Gamma\rangle_{\nexists}$ for any constraint language Γ over a finite domain D.*

With the help of the relation B^{Γ}_D we are now ready to prove the main technical lemma of this section. If $t_1, \ldots, t_m$ are k-ary tuples we write $SetCols(t_1, \ldots, t_m)$ for the set

$$\{(t_1[1], \ldots, t_m[1]), \ldots, (t_1[k], \ldots, t_m[k])\}$$

(in other words, the set corresponding to the columns of the matrix representation of $\{t_1, \ldots, t_m\}$). For example,

$$SetCols((0,0,1,1),(0,1,0,1)) = \{(0,0),(0,1),(1,0),(1,1)\}.$$

Lemma 38. *Let Γ be a conservative constraint language over a finite domain $D \subseteq \mathbb{N}$. For each conservative relation $E \subseteq D$ such that $|E| = 2$ there exists an n-ary relation $R_E \in \langle\Gamma\rangle_{\nexists}$ such that $\mathrm{Proj}_i(R_E) = E$ for some $i \in \{1, \ldots, n\}$ and $|R_E| = 2$.*

Proof. Let $n = |D|^2$ be the arity of the relation B^{Γ}_D from Definition 36, and recall that B^{Γ}_D is qfpp-definable from Γ via Lemma 37. Since each $f \in \mathrm{Pol}(\Gamma)$ is conservative it furthermore follows that $|\mathrm{Proj}_i(B^{\Gamma}_D)| \leq 2$ for each $i \in \{1, \ldots, n\}$.

Let $E = \{a, b\} \subseteq D$ be a conservative relation. Arbitrarily pick $t_1, t_2 \in B^{\Gamma}_D$ such that $(a, b) \in SetCols(t_1, t_2)$. Assume first that $SetCols(t_1, t_2) = \{(d, d) \mid d \in D\} \cup \{(a, b)\}$, i.e., the arguments of the relation $\{t_1, t_2\}$ are either constant or (a, b). It is then easily verified that there cannot exist $f \in \mathrm{pPol}(B^{\Gamma}_D)$ which does not preserve $\{t_1, t_2\}$. Hence, $\{t_1, t_2\} \in \langle B^{\Gamma}_D\rangle_{\nexists}$, and we are done.

Now assume that

$$SetCols(t_1, t_2) = \{(d, d) \mid d \in D\} \cup \{(a, b), (d_1, e_1), \ldots, (d_k, e_k)\}$$

where $d_j \neq e_j$ in each pair (d_j, e_j), $1 \leq j \leq k$. We claim that either $\{t_1, t_2\} \in \langle B^{\Gamma}_D\rangle_{\nexists}$ or there exists a tuple $t_3 \in B^{\Gamma}_D$ such that $SetCols(t_i, t_3) \subset SetCols(t_1, t_2)$ for $i = 1$ or $i = 2$, and such that $(a, b) \in SetCols(t_i, t_3)$.

Hence, assume that $\{t_1, t_2\} \notin \langle B^{\Gamma}_{D} \rangle_{\not\exists}$. Then there exists a binary partial operation $f \in \mathrm{pPol}(B^{\Gamma}_{D})$ which does not preserve $\{t_1, t_2\}$. Assume without loss of generality that $f(t_1, t_2) = t_3 \notin \{t_1, t_2\}$ for a tuple t_3 included in B^{Γ}_{D}.

Let $I_1 \subseteq \{1, \ldots, k\}$ denote the set of indices such that $f(d_i, e_i) = d_i$ whenever $i \in I_1$, and $I_2 \subseteq \{1, \ldots, k\}$ the set of indices such that $f(d_i, e_i) = e_i$ for $i \in I_2$. Note that, for each $1 \leq i \leq k$, either $f(d_i, e_i) = d_i$, or $f(d_i, e_i) = e_i$, as otherwise there exists $1 \leq j \leq n$ such that $|\mathrm{Proj}_j(B^{\Gamma}_{D})| \geq 3$, which we have already concluded is impossible. Assume, without loss of generality, that $f(a, b) = a$ (the case when $f(a, b) = b$ is symmetric).

If I_2 is empty, then $f(d_i, e_i) = d_i$ for each $1 \leq i \leq k$, implying that $f(t_1, t_2) = t_1 \in \{t_1, t_2\}$. Therefore, I_2 contains at least one element. Now consider the relation $\{t_2, t_3\}$ and the set $\mathit{SetCols}(t_3, t_2)$. Then $(a, b) \in \mathit{SetCols}(t_3, t_2)$, and we also claim that $\mathit{SetCols}(t_3, t_2) \subset \mathit{SetCols}(t_1, t_2)$. Let $1 \leq j \leq n$ and consider the two tuples $(t_1[j], t_2[j])$ and $(t_3[j], t_2[j])$. Then $(t_3[j], t_2[j]) = (f(t_1[j], t_2[j]), t_2[j])$, and hence, either

$$(f(t_1[j], t_2[j]), t_2[j]) = (t_2[j], t_2[j]) \in \mathit{SetCols}(t_1, t_2)$$

or

$$(f(t_1[j], t_2[j]), t_2[j]) = (t_1[j], t_2[j]) \in \mathit{SetCols}(t_1, t_2).$$

Furthermore, since I_2 is non-empty, there exists at least one $j \in I_2$ such that $(d_j, e_j) \in \mathit{SetCols}(t_1, t_2)$ but $(d_j, e_j) \notin \mathit{SetCols}(t_3, t_2)$. Hence, $\mathit{SetCols}(t_3, t_2) \subset \mathit{SetCols}(t_1, t_2)$.

By repeating this procedure we will arrive at a pair of tuples t, t' such that $(a, b) \in \mathit{SetCols}(t, t')$ and such that $\{(t, t')\} \in \langle B^{\Gamma}_{D} \rangle_{\not\exists}$, and we therefore let $R_E = \{t, t'\}$. □

Lemma 38 can then be used to prove an analogue of Lemma 23 from Section 4.1.

Lemma 39. *Let Γ be a conservative language over a finite domain D and let $R \in \langle \Gamma \rangle$ be an n-ary relation such that $|R| = 2$. Then there exists $R' \in \langle \Gamma \rangle_{\not\exists}$ such that (1) $|R'| = 2$ and (2) $\mathrm{Proj}_{1,\ldots,n}(R') = R$.*

Proof. This follows almost immediately by combining Lemma 23 from Section 4.1 with Lemma 38. To see this, simply note that all conservative relations in Lemma 23 are of the form $\{(d_1), (d_2)\}$ for $d_1, d_2 \in D$ or $\{(d)\}$, $d \in D$, which can be handled using Lemma 35 and Lemma 38. □

With this result we can now without too much difficulty show that we can always qfpp-define an $S_{\mathbb{B}}$-extension when $\mathrm{CSP}(\Gamma)$ is NP-complete and conservative over $\{0, 1, 2\}$.

Lemma 40. *Let Γ be a conservative constraint language over $\{0, 1, 2\}$ such that $\mathrm{CSP}(\Gamma)$ is NP-complete. Then there exists a relation in $\langle \Gamma \rangle_{\not\exists}$ which is an $S_{\mathbb{B}}$-extension.*

Proof. We will show how Lemma 24 can be adapted to the conservative setting with the help of Lemma 38 and Lemma 39. In doing so we will also recapitulate some of the crucial steps even though the constructions will be mostly identical.

Hence, by Lemma 22 we first know that there exists a relation $R \in \langle\Gamma\rangle$ which is an $S_{\mathbb{B}}$-extension. Let

$$R(x_1, \ldots, x_n) \equiv \exists y_1, y_2, \ldots, y_m.\varphi(x_1, \ldots, x_n, y_1, \ldots, y_m)$$

denote its pp-definition over Γ. Using this pp-definition we will show that Γ can qfpp-define an $S_{\mathbb{B}}$-extension by gradually removing each existentially quantified variable. First consider the relation

$$R_1(x_1, \ldots, x_n, y_1) \equiv \exists y_2, \ldots, y_m.\varphi(x_1, \ldots, x_n, y_1, y_2, \ldots, y_m).$$

Assume that $|R_1| > 3$, i.e., that R_1 is not an $S_{\mathbb{B}}$-extension. Let $R = \{t_1, t_2, t_3\}$ and for each $1 \leq i \leq 3$ let $S_i = \{t[n+1] \mid t \in R_1, t_i \frown t' = t\}$, $1 \leq i \leq 3$. In other words S_i contains the possible values taken by the argument y_1 in the tuples of R_1 corresponding to $t_i \in R$. There are now a few cases to consider depending on the sets S_1, S_2, S_3:

1. $|S_1 \cup S_2 \cup S_3| = 1$,
2. $|S_1 \cup S_2 \cup S_3| = 2$, and
3. $|S_1 \cup S_2 \cup S_3| = 3$,

The first and second case follows via identical arguments as the proof of Lemma 24 but using Lemma 39 instead of Lemma 23.

Hence, assume that $|S_1 \cup S_2 \cup S_3| = 3$, and first assume that $S_1 \cap S_2 \cap S_3 = \emptyset$. If $S_1 = \{d_1\}$, $S_2 = \{d_2\}$, and $S_3 = \{d_3\}$, then R_1 is already an $S_{\mathbb{B}}$-extension. Next, assume that $S_1 = S_2 = \{d_1, d_2\}$ and that $S_3 = \{d_3\}$ (or any other case symmetric to this one). Let $E = \{d_1, d_3\}$ and let E' be the corresponding relation from Lemma 39, and assume that $\text{ar}(E') = k + 1$. Then we may qfpp-define the $S_{\mathbb{B}}$-extension R'_1 of arity $n + 1 + k$ as

$$R'_1(x_1, \ldots, x_n, y_1, z_1, \ldots, z_k) \equiv R_1(x_1, \ldots, \ldots, x_n, y_1) \wedge E'(y, z_1, \ldots, z_k).$$

Last, assume that $S_1 = \{d_1, d_2\}$, $S_2 = \{d_1, d_3\}$, $S_3 = \{d_2, d_3\}$ (or a case symmetric to this one). Then, just as in the corresponding case of the proof of Lemma 24, we let $E = \{d_1, d_2\}$, E' be the $(k+1)$-ary relation from Lemma 39, and qfpp-define the relation R'_1 of arity $n + 1 + k$ as

$$R'_1(x_1, \ldots, x_n, y_1, z_1, \ldots, z_k) \equiv R_1(x_1, \ldots, \ldots, x_n, y_1) \wedge E'(y, z_1, \ldots, z_k).$$

The argument then proceeds as in the case when $|S_1 \cup S_2 \cup S_3| = 2$.

Second, assume that $S_1 \cap S_2 \cap S_3 \neq \emptyset$. Then we arbitrarily pick an element $d \in S_1 \cap S_2 \cap S_3$. It is easily verified that $R_1(x_1, \ldots, x_n, y) \wedge \{(d)\}(y)$ defines an $S_{\mathbb{B}}$-extension, and using Lemma 35 this relation, or possibly a relation with additional arguments, is qfpp-definable over Γ. If, instead, $S_1 = \{d_1, d_2\}$, $S_2 = \{d_1, d_3\}$, and $S_3 = \{d_2, d_3\}$, then we let $E = \{d_1, d_3\}$ and let E' be the corresponding relation from Lemma 39 of arity $k+1$, and construct the relation $R'_1(x_1, \ldots, x_n, y_1, z_1, \ldots, z_k) \equiv R_1(x_1, \ldots, \ldots, x_n, y_1) \wedge E'(y, z_1, \ldots, z_k)$. Similarly to the proof of Lemma 24 this case then reduces to the case when $|S_1 \cup S_2 \cup S_3| = 2$. $\square$

It is then easy to see that the reductions in Section 4.2 allows us to claim the following theorem, since Γ can qfpp-define an $S_{\mathbb{B}}$-extension whenever Γ is conservative and $\mathrm{CSP}(\Gamma)$ is NP-complete.

Theorem 41. *Let Γ be a finite, conservative constraint language over $\{0, 1, 2\}$. If $\mathrm{CSP}(\Gamma)$ is NP-complete then $\mathsf{T}(\{S_D\}) \leq \mathsf{T}(\Gamma)$.*

It is not obvious how to generalise Theorem 41 to arbitrary finite domains. A good starting point would be to lift Lemma 38 to relations E with $|E| \geq 3$. However, the proof strategy used in Lemma 38 is based on finding tuples t_1, t_2 which either correctly define E, or induce a binary partial operation which can be used to select a third, refined tuple t_3. But if $|E| = 3$ then we cannot necessarily guarantee that the tuple t_3 results in a refinement, and the procedure might not terminate.

6 Concluding Remarks and Future Research

In this article we have studied the time complexity of NP-complete CSPs. We have ruled out subexponential-time algorithms for NP-complete, finite-domain CSPs, unless the ETH is false. This proof also extends to degree-bounded CSPs and many classes of CSPs over infinite domains. We then proceeded to study the time complexity of CSPs over ultraconservative constraint languages, and proved that no such NP-complete CSP is solvable strictly faster than $\mathsf{T}(\{S_D\})$. It is an interesting open question if this can be extended to arbitrary constraint languages, and we proved that for ternary domains it is in fact sufficient that Γ is merely conservative. These results raise several directions for future research.

Structurally restricted CSPs and the ETH. Theorem 13 shows that the algebraic approach is viable for analysing the existence of subexponential algorithms for certain structurally restricted $\mathrm{CSP}(\Gamma)$ problems. An interesting continuation would be to try to determine which of the structurally restricted (but not constraint language restricted) CSPs investigated by De Haan et al. [23] could be used to prove similar results. For example, is it the case that $\mathrm{CSP}(\Gamma)$ is not solvable in subexponential time whenever $\mathrm{CSP}(\Gamma)$ is NP-complete and the primal treewidth of an instance is bounded by $\Omega(n)$, unless the ETH fails?

Stronger results for arbitrary constraint languages. We have seen that for the ternary domain $\{0, 1, 2\}$ it is possible to show that $\mathrm{CSP}(S_{\{0,1,2\}})$ results in the easiest conservative, NP-complete CSP. However, generalising this to larger domains seems to be riddled with technical difficulties. Is it possible to circumvent this difficulty by directly proving that $\mathrm{CSP}(\Gamma \cup 2^D) \leq^{\mathrm{CV}} \mathrm{CSP}(\Gamma)$ whenever $\mathrm{CSP}(\Gamma)$ is NP-complete?

For non-conservative constraint languages the situation appears to be even more complicated. Most importantly, it is not clear if Lemma 22 is valid for arbitrary constraint languages since the construction explicitly utilises conservative constraints over the domain, and it is thus uncertain whether Γ can qfpp-define an $S_{\mathbb{B}}$-extension whenever $\mathrm{CSP}(\Gamma)$ is NP-complete. It might therefore be necessary to consider a less algebraic proof and instead directly try to prove that $\mathrm{CSP}(R) \leq^{\mathrm{CV}} \mathrm{CSP}(\Gamma)$ for an $S_{\mathbb{B}}$-extension R, rather than proving that R can be qfpp-defined.

Infinite-domain CSPs. If the infinite-domain CSP dichotomy conjecture for CSPs over reducts of finitely bounded homogeneous structures is correct, we may already deduce that these problems are not solvable in subexponential time [3] whenever they are NP-hard. However, for arbitrary constraint languages over infinite domains there is reason to believe that the situation is overall much more complex. For example, there are NP-complete problems $\mathrm{CSP}(\Gamma)$ not solvable in $O(c^n)$ time for *any* $c \geq 0$, assuming the SETH is true [34], which is in stark contrast to finite-domain CSPs which are always solvable in $O(|D|^n)$ time for every finite domain D. In contrast to Theorem 13, the results by Jonsson & Lagerkvist [34] were not obtained by pp-interpretability, and it would be interesting to investigate if such strong, lower bounds could be obtained using algebraically informed approaches.

Last, we have seen that the time complexity of $\mathrm{CSP}(S_D)$ strictly decreases for increasingly larger domains D, which makes it very tempting to find examples of NP-complete CSPs over infinite domains solvable in subexponential time. Due to Theorem 13, such CSPs — if they exist — would fall outside the infinite-domain CSP dichotomy conjecture. For this question it is important to note that there is no obvious generalisation of S_D to infinite domains since such a relation would consist of an infinite number of arguments. Also, the language $\{S_D \mid D \subset \mathbb{N}\}$, while defined over the infinite domain $\mathbb{N}$, is not relevant in this pursuit since the time complexity of $\mathrm{CSP}(\{S_D \mid D \subset \mathbb{N}\})$ is not lower than $\mathrm{CSP}(S_{\{0,1\}})$.

Acknowledgements

We thank Hannes Uppman for several helpful discussions on the topic of this article. The first two authors are partially supported by the Swedish Research Council (VR) under grant 2017-04112. In addition, the second author has received funding from the DFG-funded project "Homogene Strukturen, Bedingungserfüllungsprobleme, und topologische Klone" (Project number 622397), the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 681988, CSP-Infinity), and by VR under grant 2019-03690. The third author is partially supported by the *National Graduate School in Computer Science* (CUGS), Sweden.

References

[1] V. B. Alekseev and A. A. Voronenko. On some closed classes in partial two-valued logic. *Discrete Mathematics and Applications*, 4(5):401–419, 1994.

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

[3] L. Barto and M. Pinsker. The algebraic dichotomy conjecture for infinite domain constraint satisfaction problems. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS-2016)*, pages 615–622, New York, NY, USA, 2016. ACM.

[4] M. Behrisch, M. Hermann, S. Mengel, and G. Salzer. Minimal distance of propositional models. *Theory of Computing Systems*, 63(6):1131–1184, Aug 2019.

[5] M. Bodirsky. Complexity classification in infinite-domain constraint satisfaction. Mémoire d'habilitation à diriger des recherches, Université Diderot – Paris 7. Available at arXiv:1201.0856, 2012.

[6] M. Bodirsky, P. Jonsson, and T. V. Pham. The complexity of phylogeny constraint satisfaction problems. *ACM Transactions on Computational Logic*, 18(3):23:1–23:42, 2017.

[7] M. Bodirsky and J. Kára. The complexity of equality constraint languages. *Theory of Computing Systems*, 43(2):136–158, Aug 2008.

[8] M. Bodirsky and J. Kára. The complexity of temporal constraint satisfaction problems. *Journal of the ACM*, 57(2):9:1–9:41, 2010.

[9] M. Bodirsky and M. Pinsker. Schaefer's theorem for graphs. *Journal of the ACM*, 62(3):19:1–19:52, June 2015.

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

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

[12] E. Böhler, E. Hemaspaandra, S. Reith, and H. Vollmer. Equivalence and isomorphism for boolean constraint satisfaction. In *In Proceedings of the 16th International Workshop on Computer Science Logic (CSL-2002)*, pages 412–426, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[13] F. Börner. Basics of Galois connections. In N. Creignou, P.G. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, volume 5250 of *Lecture Notes in Computer Science*, pages 38–67. Springer Berlin Heidelberg, 2008.

[14] A. Bulatov. Complexity of conservative constraint satisfaction problems. *ACM Transactions on Computational Logic*, 12(4):24:1–24:66, July 2011.

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

[16] A. Bulatov and A. Hedayaty. Counting problems and clones of functions. *Multiple-Valued Logic and Soft Computing*, 18(2):117–138, 2012.

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

[18] C. Calabro, R. Impagliazzo, and R. Paturi. The complexity of satisfiability of small depth circuits. In J. Chen and F. V. Fomin, editors, *Parameterized and Exact Computation*, volume 5917 of *Lecture Notes in Computer Science*, pages 75–85. Springer Berlin Heidelberg, 2009.

[19] C. Carbonnel and M. C. Cooper. Tractability in constraint satisfaction problems: a survey. *Constraints*, 21(2):115–144, Apr 2016.

[20] M. C. Cooper and S. Zǐvný. Hybrid Tractable Classes of Constraint Problems. In Andrei Krokhin and Stanislav Zǐvný, editors, *The Constraint Satisfaction Problem: Complexity and Approximability*, volume 7 of *Dagstuhl Follow-Ups*, pages 113–135. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2017.

[21] N. Creignou, U. Egly, and J. Schmidt. Complexity classifications for logic-based argumentation. *ACM Transactions on Computational Logic*, 15(3):19:1–19:20, 2014.

[22] M. Cygan, F. V. Fomin, A. Golovnev, A. S. Kulikov, I. Mihajlin, J. Pachocki, and A. Socała. Tight bounds for graph homomorphism and subgraph isomorphism. In *Proceedings of the Twenty-seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-2016)*, pages 1643–1649, Philadelphia, PA, USA, 2016. Society for Industrial and Applied Mathematics.

[23] R. de Haan, I. A. Kanj, and S. Szeider. On the subexponential-time complexity of CSP. *Journal of Artificial Intelligence Research (JAIR)*, 52:203–234, 2015.

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

[25] R. V. Freivald. A Completeness Criterion for Partial Functions of Logic and Many-Valued Logic Algebras. *Soviet Physics Doklady*, 11:288, October 1966.

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

[27] M. Grohe. The structure of tractable constraint satisfaction problems. In *Proceedings of the 31st International Symposium on Mathematical Foundations of Computer Science (MFCS-2006)*, pages 58–72, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[28] L. Ham. Gap theorems for robust satisfiability: Boolean CSPs and beyond. *Theoretical Computer Science*, 676:69 – 91, 2017.

[29] T. Hertli. 3-SAT faster and simpler - unique-SAT bounds for PPSZ hold in general. *SIAM Journal on Computing*, 43(2):718–729, 2014.

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

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

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

[33] P. Jeavons, D. A. Cohen, and M. Gyssens. How to determine the expressive power of constraints. *Constraints*, 4(2):113–131, 1999.

[34] P. Jonsson and V. Lagerkvist. An initial study of time complexity in infinite-domain constraint satisfaction. *Artificial Intelligence*, 245:115–133, 2017.

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

[36] V. Lagerkvist. Precise upper and lower bounds for the monotone constraint satisfaction problem. In *Proceedings of the 40th International Symposium on Mathematical Foundations of Computer Science (MFCS-2015)*, pages 357–368, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.

[37] R. Pöschel. Galois connections for operations and relations. In K. Denecke, M. Erné, and S.L. Wismath, editors, *Galois Connections and Applications*, volume 565 of *Mathematics and Its Applications*, pages 231–258. Springer Netherlands, 2004.

[38] E. Post. The two-valued iterative systems of mathematical logic. *Annals of Mathematical Studies*, 5:1–122, 1941.

[39] B. A. Romov. The completeness problem in partial hyperclones. *Discrete Mathematics*, 306(13):1405–1414, July 2006.

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

[41] F. Rossi, P. van Beek, and T. Walsh, editors. *Handbook of Constraint Programming*, volume 2 of *Foundations of Artificial Intelligence*. Elsevier, 2006.

[42] S. J. Russell and P. Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.

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

[44] H. Schnoor and I. Schnoor. Partial polymorphisms and constraint satisfaction problems. In N. Creignou, P. G. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, volume 5250 of *Lecture Notes in Computer Science*, pages 229–254. Springer Berlin Heidelberg, 2008.

[45] K. Schölzel. Dichotomy on intervals of strong partial Boolean clones. *Algebra Universalis*, 73(3-4):347–368, 2015.

[46] M. Wahlström. *Algorithms, measures and upper bounds for satisfiability and related problems*. PhD thesis, Linköping University, 2007.

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