Algorithmica (2023) 85:188–215
https://doi.org/10.1007/s00453-022-01017-8

General Lower Bounds and Improved Algorithms for Infinite–Domain CSPs

Peter Jonsson[1] · **Victor Lagerkvist**[1]

Received: 16 September 2020 / Accepted: 22 July 2022 / Published online: 11 August 2022
© The Author(s) 2022

Abstract
We study the fine-grained complexity of NP-complete, infinite-domain constraint satisfaction problems (CSPs) parameterised by a set of first-order definable relations (with equality). Such CSPs are of central importance since they form a subclass of *any* infinite-domain CSP parameterised by a set of first-order definable relations over a relational structure (possibly containing more than just equality). We prove that under the randomised exponential-time hypothesis it is not possible to find $c > 1$ such that a CSP over an arbitrary finite equality language is solvable in $O(c^n)$ time (n is the number of variables). Stronger lower bounds are possible for infinite equality languages where we rule out the existence of $2^{o(n \log n)}$ time algorithms; a lower bound which also extends to *satisfiability modulo theories* solving for an *arbitrary* background theory. Despite these lower bounds we prove that for each $c > 1$ there exists an NP-hard equality CSP solvable in $O(c^n)$ time. Lower bounds like these immediately ask for closely matching upper bounds, and we prove that a CSP over a finite equality language is always solvable in $O(c^n)$ time for a fixed c, and manage to extend this algorithm to the much broader class of CSPs where constraints are formed by first-order formulas over a unary structure.

Keywords Constraint satisfaction · Infinite domains · Equality languages · Fine-grained complexity · Lower bounds

1 Introduction

In this article we study the *fine-grained*, rather than classical, *coarse*, complexity of infinite-domain *constraint satisfaction problems*. We approach the subject in a

✉ Victor Lagerkvist
victor.lagerkvist@liu.se

Peter Jonsson
peter.jonsson@liu.se

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

systematic manner and obtain powerful lower bounds applicable to *all* infinite-domain CSPs where constraints consists of first-order definable relations over a fixed relational structure. In the direction of upper bounds we obtain improved, single-exponential time algorithms for *equality* CSPs, and the broader class of CSPs over reducts of *unary structures*. Some parts of this article have been presented in preliminary form in a conference publication [20].

1.1 Background

Let Γ denote a (finite or infinite) set of finitary relations over a (finite or infinite) set D. The input to the *constraint satisfaction problem* over Γ (CSP(Γ)) is a pair (V, C) where V is a set of variables (with domain D) and C is a set of *constraints* over Γ. A constraint is an expression $R(x_1, \ldots, x_n)$ where $x_1, \ldots, x_n$ are variables in V and n equals the arity of the relation R. The problem is to find an assignment $f : V \to D$ that satisfies every constraint in C, i.e. $(f(x_1), \ldots, f(x_n)) \in R$ for every constraint $R(x_1, \ldots, x_n)$ in C. The set Γ is called the *constraint language* or the *template*. The CSP is computationally hard in the general case; if the variable domains are finite, then the problem is NP-complete, and otherwise it may be of arbitrarily high complexity or even undecidable [6].

Depending on the constraint language Γ, it is possible to formulate many natural problems as CSP(Γ) problems. This is especially true if we allow templates over an infinite universe, which increases the expressive power of CSPs and e.g. makes it possible to formulate a rich amount of problems from artificial intelligence [7, 15]. The complexity of CSPs have also been the subject of intense theoretical research: for each constraint language Γ over a finite domain CSP(Γ) is always either polynomial-time solvable or is NP-complete [13, 41]. Infinite-domain CSPs are in general undecidable, but there exists a wealth of results when additional restrictions are imposed. Early examples include the CSP formulation of *Allen's interval algebra* [25], the *region connection calculus* [31], CSPs over first-order definable relations with equality [8] (equality CSPs), and *temporal* CSPs [9], i.e. CSPs where the constraint language is first-order definable in the structure $(\mathbb{Q}; <)$ whose domain is the set of rational numbers $\mathbb{Q}$ and where $<$ denotes the usual strict order of the rationals. More generally, it is common to consider *first-order reducts* of a fixed relational structure $\mathcal{A}$, i.e., languages that are first-order definable *with equality* over $\mathcal{A}$. Equality CSPs then correspond to CSP(Γ) when Γ is a first-order reduct of $(A; \emptyset)$ for some universe A (an *equality language*) while temporal CSPs correspond to CSP(Γ) when Γ is a first-order reduct of $(\mathbb{Q}; <)$. To make the intended meaning clearer we sometimes treat equality languages as first-order reducts of $(A; =)$, where $=$ is the equality relation over the universe A, even though this is strictly speaking not needed since the equality relation is always allowed in first-order formulas. Equality CSPs have previously been intensively studied due to their fundamental importance for understanding more complex CSPs, since any classification of a larger relational structure $\mathcal{A}$ necessarily also needs to include a classification of equality CSPs (an equality language Γ is a reduct of *any* countably infinite structure $\mathcal{A}$). Let us also remark that CSPs in this setting are very similar to reasoning problems occurring in artificial intelligence, where

one fixes a set of "base relations" $\mathcal{A}$, typically binary, and then consider a satisfiability problem where constraints are taken from e.g. the relation algebra generated by $\mathcal{A}$, or the set of all disjunctive clauses over $\mathcal{A}$ [15]. A recent comparison may also be found in *satisfiability modulo theories* (SMT) where a background theory $\mathcal{A}$ is fixed, and where one considers the satisfiability problem of first-order formulas (with equality) restricted to interpretations agreeing with $\mathcal{A}$ [3].

While theoretical CSP research has concentrated on classical complexity, complexity theory itself has partially shifted towards parameterised complexity and *fine-grained complexity*, which e.g. encompasses constructing improved exponential-time algorithms, and proving lower bounds with stronger assumptions than P $\neq$ NP. A popular conjecture for this purpose is the *exponential-time hypothesis* (ETH). It states that the 3-SAT problem is not solvable in subexponential time, i.e. it is not solvable in $2^{o(n)}$ time, where n is the number of variables. Another popular conjecture is the *strong ETH* (SETH) which, roughly, states that the fine-grained complexity of k-SAT tends to 2^n for increasing values of k.

In this article we study the fine-grained complexity of NP-hard infinite-domain CSPs, with a particular focus on equality CSPs using the number of variables, n, as the complexity parameter. As remarked, equality CSPs constitute a natural starting point for questions of fine-grained complexity, since if we cannot even overcome this obstacle there is little hope of understanding fine-grained complexity questions for larger classes of CSPs. Assume, for example, that we prove that there exists an equality language Γ such that CSP(Γ) is not solvable in $O(f(n))$ time, for some function f. Then, regardless of which relational structure $\mathcal{A}$ that we choose, we cannot hope to construct an algorithm with a running time of $O(f(n))$ which is applicable to CSP(Δ) for every first-order reduct Δ of $\mathcal{A}$. Under this viewpoint it is therefore crucial to prove lower bounds for equality CSPs before moving on to construct faster exponential-time algorithms for broader classes of infinite-domain CSPs.

Thus, among the class of NP-hard equality CSPs, how does the choice of Γ affect the fine-grained complexity of CSP(Γ)? For example, it is known that CSP(Γ) is solvable in $O^*(2^{n\cdot\log(\frac{0.792n}{\ln(n+1)})})$ time when Γ is an arbitrary equality language [18] (the O^* notation is used to suppress polynomial factors). Concerning lower bounds it is known that no NP-complete equality CSP(Γ) problem is solvable in subexponential time without violating the ETH. This follows from Barto & Pinsker [2]: if Γ is an equality language and CSP(Γ) is NP-hard, then Γ pp-interprets 3-SAT since Γ is a first-order reduct of the finitely bounded homogeneous structure $(\mathbb{N}; =)$. This fact combined with Theorem 3.1 in [22] gives the result. Furthermore, if Γ is the full first-order reduct of $(A; =)$ then there cannot exist an $O^*(c^n)$ time algorithm for CSP(Γ) for any constant c without violating the SETH [18]. Despite bounds like these, there are still large gaps in our understanding of fine-grained complexity of infinite-domain CSPs in general, and of equality CSPs in particular. For example, is it possible to find an equality language Γ such that CSP(Γ) is NP-complete but solvable in $O(c^n)$ time for a constant $c > 1$? Is it possible to solve CSP(Γ) in $O(c^n)$ time whenever Γ is a finite equality language, and in that case, does c depend on Γ or is it possible to find a uniform value? Furthermore, since no NP-complete equality CSP is solvable in

subexponential time without violating the ETH, does there exist a $c > 1$ such that no NP-complete equality CSP is solvable in $O(c^n)$ time?

1.2 Our Results

After defining the necessary preliminaries (in Sect. 2) we in Sect. 3 begin to answer the aforementioned questions by a careful study of lower bounds. First, we prove that under the *randomised ETH* for each $c > 1$ there exists a finite equality language Γ_c such that $\mathrm{CSP}(\Gamma_c)$ is not solvable in $O(c^n)$ time. Second, we showcase a striking difference between finite and infinite languages and prove the existence of an infinite equality language Γ such that $\mathrm{CSP}(\Gamma)$ is not solvable in $2^{o(n \log n)}$ time (under the ETH). In particular this lower bound rules out a uniform $O(c^n)$ time algorithm, $c > 1$, applicable to arbitrary equality CSPs (which previously was only known to hold under the much stronger SETH). We also manage to lift this lower bound to SMT, where little is known about the fine-grained complexity, despite being a framework with a wide range of applications due to the availability of efficient SAT solvers. We provide the first known lower bound under the ETH and show that *regardless* of the background theory it is not possible to solve the resulting SMT in $2^{o(n \log n)}$ time without violating the ETH. Importantly, this shows that existing algorithms for SMT running in $2^{O(n \log n)}$ time are close to being optimal (cf. Rodeh & Strichman [32]). It should also be noted that we are able to prove this as a straightforward consequence of our general bounds for equality CSPs, indicating yet another advantage of studying fine-grained complexity in this setting. Third, we prove that for each constant $c > 1$ there exists an NP-complete equality CSP which is solvable in $O(c^n)$ time, and thus rule out the existence of an "easiest NP-complete equality CSP". Such CSPs are known to exist for finite-domain CSPs [22] so we see a clear dividing line between finite and infinite-domain CSPs. We also provide an algebraic explanation of the lack of such an "easiest CSP problem", based on a connection between fine-grained complexity of CSPs and algebraic invariants called *partial polymorphisms*. Since partial polymorphisms recently have become an important tool for studying fine-grained complexity of CSPs and related problems [21–23, 26, 27] it therefore appears important to chart any differences to the finite-domain case. In short, an "easiest NP-complete CSP" would have a maximally large set of partial polymorphisms, and we prove that such a maximal set cannot exist for the set of all equality relations. The proof also generalises to other classes of languages, e.g., temporal languages, and can, interestingly, be proven independently of any complexity theoretical assumptions.

In light of these lower bounds, what is the best possible exponential-time algorithm for equality CSPs that we could hope for? We tackle this question in Sect. 4 and construct an $O^*(c^n)$ time algorithm for $\mathrm{CSP}(\Gamma)$ whenever Γ is a *finite* equality language, where c is a constant depending only on the arities of relations in Γ. Note that while the constant c likely can be improved, we have already established (under the randomised ETH) that it is not possible to find a uniform value. Similarly, it appears difficult to extend the algorithm to non-trivial classes of infinite equality languages since we have already proved that there is an infinite equality language that cannot be solved in $2^{o(n \log n)}$ time under the ETH. Here, it is also interesting to note that certain

classes of infinite-domain CSPs do not admit an $O(c^n)$ algorithm even if the template is finite. For instance, there is a finite temporal language whose CSP (under the randomised ETH) cannot be solved in $2^{o(n \log n)}$ time [19]. Could it even be the case that finite equality CSPs are the *only* reasonable class of infinite-domain CSPs solvable in single-exponential time, and that every non-trivial structure results in CSPs of higher complexity? This, however, is not the case, and we do manage to construct an $O^*(c^n)$ time algorithm applicable to a much richer and broader class of problems, namely CSPs over reducts of unary structures. More precisely, say that Γ is a *unary structure* (US) language if Γ is a first-order reduct of a structure $(A; U_1, \ldots, U_k)$ where each U_i is unary. Such CSPs are a subclass of first-order definable structures with atoms and have attracted recent attention from the automata theory community [11, 12, 24]. They have also been studied by the CSP community and the complexity has been fully classified [5, 10]. The algorithm works by partitioning the domains of the unary relations $U_1, \ldots, U_k$ in such a way that we create a finite "pseudo-universe" $\mathbf{U}$ where each element gives an implicit description of a finite equality language. This makes it possible to enumerate the elements of $\mathbf{U}$ and in each iteration test the satisfiability of the corresponding equality CSP instance.

These results paint a peculiar picture of the fine-grained complexity of equality CSPs (and *all* classes of infinite-domain CSPs over first-order reducts of relational structures). On the one hand, equality CSPs are incredibly hard to solve (no uniform $O(c^n)$ time algorithm for finite languages under the randomised ETH, and no $2^{o(n \log n)}$ time algorithm for infinite languages), but on the other hand one for any $c > 1$, say, $c = 1.00001$, can find an NP-hard equality CSP solvable in $O(c^n)$ time. These conflicting messages indicate that a complete understanding of fine-grained complexity of equality CSPs is well out of reach, but we have simultaneously unravelled several interesting research directions. We discuss some of these in Sect. 5.

2 Preliminaries

A *relational structure* is a tuple $(A; \sigma, I)$ where A is a set typically called a *domain*, or a *universe*, σ is a relational signature, and I is a function from σ to the set of all relations over A which assigns each relation symbol a corresponding relation over A. For simplicity, we will typically write a relational structure as $(A; R_1, \ldots, R_k)$ where each R_i is a relation over A, and will not make a sharp distinction between relations and their corresponding signatures. A set of relations Γ over A is a *first-order reduct* of a relational structure $\mathbf{A} = (A; R_1, \ldots, R_k)$ if each $R \in \Gamma$ is the set of models of a σ-formula (with equality) interpreted in $(A; R_1, \ldots, R_k)$. Alternatively, one may view Γ as a set of relations where each relation has a first-order definition (without parameters) in $\mathbf{A}$. In symbols, we write $R(x_1, \ldots, x_n) \equiv \varphi(x_1, \ldots, x_n)$ if R is the set of models of the first-order formula $\varphi(x_1, \ldots, x_n)$ with respect to the free variables $x_1, \ldots, x_n$.

2.1 The Constraint Satisfaction Problem

Let Γ be a set of finitary relations over some set A of values, occasionally called a *constraint language*. The *constraint satisfaction problem* over Γ ($\mathrm{CSP}(\Gamma)$) is defined as follows.

> *Instance:* A set V of variables and a set C of *constraints* of the form $R(x_1, \ldots, x_k)$, where k is the arity of R, $x_1, \ldots, x_k \in V$ and $R \in \Gamma$.
> *Question:* Is there a function $f\colon V \to A$ such that $(f(x_1), \ldots, f(x_k)) \in R$ for every $R(x_1, \ldots, x_k) \in C$?

Concerning representation, we take a simple approach and only consider the case when Γ is a first-order reduct of a relational structure, and represent each relation $R \in \Gamma$ by a first-order formula. However, the exact representation is only important if Γ is infinite, since any reasonable representation can be chosen and precomputed if Γ is finite.

2.2 Primitive Positive Definitions and Interpretations

Let Γ be a constraint language over a domain A. 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 $R_i \in \Gamma \cup \{\mathrm{Eq}_A\}$ and each $\mathbf{x_i}$ is a tuple of variables over $x_1, \ldots, x_k$, $y_1, \ldots, y_{k'}$ matching the arity of R_i. Here, and in the sequel, Eq_A is the equality relation $\{(a, a) \mid a \in A)\}$ over A. Thus, R is definable by a first-order formula consisting only of existential quantification and conjunction over positive atoms from Γ and equality constraints. If Γ is a constraint language we let $\langle\Gamma\rangle$ be the smallest set of relations containing Γ closed under pp-definitions. Pp-definitions are typically only useful for comparing similar languages over the same domain, but can be generalised as follows.

Definition 1 Let A and B be two domains and let Γ and Δ be two constraint languages over A and B, respectively. A *primitive positive interpretation* (pp-interpretation) of Δ over Γ consists of:

1. a d-ary relation $F \subseteq A^d$,
2. and a surjective function $f\colon F \to B$

such that F, $f^{-1}(\mathrm{Eq}_B) \in \langle\Gamma\rangle$ and $f^{-1}(R) \in \langle\Gamma\rangle$ for every k-ary $R \in \Delta$, where $f^{-1}(R)$ denotes the $(k \cdot d)$-ary relation

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

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

Theorem 1 (cf. Theorem 3.1.4 in Bodirsky [4]) *If Γ and Δ are finite constraint languages and there exists a pp-interpretation of Δ over Γ, then CSP(Δ) is polynomial-time reducible to CSP(Γ).*

We invite the reader to verify that a standard reduction from the 3-coloring problem (formulable as a CSP over the inequality relation over a ternary domain) to 3-SAT (formulable as a Boolean CSP) can be expressed as a pp-interpretation of the 3-coloring relation over 3-SAT.

2.3 Equality Languages

We say that Γ is an *equality language* if each $R \in \Gamma$ admits a first-order definition over a relational structure $(A; \emptyset)$, i.e. the empty structure. Recall here that the equality relation is always accessible in first-order logic. Without loss of generality we henceforth assume that $A = \mathbb{N}$, write Eq (or $=$ in infix notation) for the equality relation over $\mathbb{N}$, and $\mathrm{R}_{\neq}$ or $\neq$ (in infix notation) for the inequality relation $\{(x, y) \in \mathbb{N}^2 \mid x \neq y\}$ over $\mathbb{N}$. Via these conventions an equality language can also be viewed as a set of first-order definable relations over $(\mathbb{N}; =)$, and we typically prefer this notation over $(A; \emptyset)$ since the intended meaning is clearer. The computational problem we consider is then CSP(Γ) when Γ is an equality language. This problem is easily seen to belong to NP for any finite language, and its classical complexity has been completely classified [8].

Theorem 2 *Let Γ be an equality language. Then either*

1. *CSP(Γ) is polynomial-time solvable or*
2. *there exists a finite $\Delta \subseteq \Gamma$ such that CSP(Δ) is NP-complete since Δ pp-interprets every finite-domain relation.*

Example 1 Let $S = \{(x, x, y), (x, y, y) \mid x, y \in \mathbb{N}, x \neq y\}$, and observe that $S(x, y, z) \equiv (x = y \wedge y \neq z) \vee (x \neq y \wedge y = z)$. Thus, $\{S\}$ is an equality language, and it is known that $\{S\}$ pp-interprets a language Δ where CSP(Δ) is NP-hard, which implies that CSP($\{S\}$) is NP-hard, too. For tractability, if we take $\{\mathrm{Eq}, \mathrm{R}_{\neq}\}$ then CSP($\{\mathrm{Eq}, \mathrm{R}_{\neq}\}$) is well-known to be polynomial-time solvable. This can be proven via Theorem 2, however, CSP($\{\mathrm{Eq}, \mathrm{R}_{\neq}\}$) can also be solved by elementary propagation methods.

2.4 Fine-Grained Complexity and the Exponential–Time Hypothesis

Assume that CSP(Γ) is NP-complete. How fast can we solve CSP(Γ), and is it possible to prove stronger lower bounds than an expected superpolynomial running time (under P $\neq$ NP)? Such questions, especially when the complexity parameter is the number of variables $|V|$ or the number of constraints $|C|$, fall under the umbrella of *fine-grained complexity*. To prove non-trivial lower bounds for NP-complete problems we typically need stronger assumptions than P $\neq$ NP. Say that CSP(Γ) is solvable in *subexponential time* if CSP(Γ) is solvable in $O(2^{\varepsilon|V|})$ for each $\varepsilon > 0$. The conjecture that 3-SAT is not solvable in subexponential time is called the *exponential-time*

hypothesis (ETH). There exists several stronger variants of the ETH. First, an algorithm A is said to be a $2^{c\cdot|V|}$-*randomised algorithm* if its running time is bounded by $2^{c\cdot|V|}\cdot \text{poly}(||I||)$ and its error probability is at most $1/3$ ($||I||$ is the number of bits required to represent a CSP instance I). For $k, d \geq 1$ we then define

$$c_k = \inf\{c \mid \exists \text{ a deterministic } 2^{c\cdot|V|} \text{ algorithm for } k-\text{SAT}\}$$

and

$$c_{d,k} = \inf\{c \mid \exists \text{ a } 2^{c\cdot|V|}\text{-randomised algorithm for CSP}(\Gamma_{d,k})\},$$

where $\Gamma_{d,k}$ is the set of all relations over the set $\{0,\ldots,d-1\}$ of arity at most k. The *randomised exponential-time hypothesis* (r-ETH) is then the conjecture that $c_{2,3} > 0$, i.e., that 3-SAT is not solvable in subexponential time even with randomised algorithms, and the *strong exponential-time hypothesis* (SETH) is the conjecture that the limit of the sequence $c_3, c_4, \ldots$ is equal to 1.

3 Lower Bounds on the Complexity of Equality Constraints

In this section we investigate lower bounds for equality CSPs. As remarked in Sect. 1, such lower bounds are valuable since if it is possible to prove that, for an equality language Γ, CSP(Γ) is not solvable in $O(f(|V|))$ time (for some function f) then, for some arbitrary relational structure $\mathbf{A}$, there exists a Δ such that CSP(Δ) is not solvable in $O(f(|V|))$ time and Δ is a first-order reduct of $\mathbf{A}$. Let us recapitulate two known lower bounds.

Theorem 3 *Let Γ be an equality language.*

1. *If CSP(Γ) is NP-hard then it is not solvable in subexponential time unless the ETH is false (Theorem 3.1 in [22]), and*
2. *If Γ is the full first-order reduct of $(\mathbb{N}; =)$ then CSP(Γ) is not solvable in $O(c^{|V|})$ time for* any $c > 1$ *unless the SETH is false (Theorem 19 in* [18]*).*

3.1 Finite Versus Infinite Equality Languages

We begin by proving that for every $c > 1$ there exists a finite equality language Γ_c such that CSP(Γ_c) is not solvable in $O(2^{c|V|})$ time without contradicting the r-ETH. This result is a substantial strengthening of Theorem 3(2). We first require the following result [39, Theorem 1].

Theorem 4 *If r-ETH holds, then there exists a universal constant $\alpha > 0$ such that $\alpha\cdot\log(d) \leq c_{d,2}$ for all $d \geq 3$,*

Theorem 5 *For every $c > 1$, there exists a finite equality language Γ_c such that CSP(Γ_c) cannot be solved in $O(2^{c\cdot|V|})$ (randomised) time unless the r-ETH is false.*

Proof For $1 \leq a, b \leq d$ define

$$R_{d,a,b}(c_1, \ldots, c_d, x, y) \equiv \bigvee_{i=1}^{d} x = c_i \wedge \bigvee_{i=1}^{d} y = c_i \wedge (x \neq c_a \vee y \neq c_b)\,.$$

For arbitrary d then define the finite equality language

$$\Theta_d = \{\neq\} \cup \{R_{d,a,b} \mid 1 \leq a, b \leq d\}.$$

We present a polynomial-time reduction from CSP($\Gamma_{d,2}$) to CSP(Θ_d) only introducing a constant number of fresh variables. Let (V, C) be an instance of CSP($\Gamma_{d,2}$). Introduce d fresh variables $c_1, \ldots, c_d$ together with constraints $\{c_i \neq c_j \mid 1 \leq i < j \leq d\}$. For each $R(x, y) \in C$, add the constraints $R_{d,a,b}(c_1, \ldots, c_d, x, y)$ for every $1 \leq a, b \leq d$ such that $(a, b) \notin R$. The resulting instance $(V \cup \{c_1, \ldots, c_d\}, C')$ can be constructed in polynomial time, and is clearly satisfiable if and only if (V, C) is satisfiable. Furthermore, d is fixed so only a constant number of variables are introduced. By Theorem 4, CSP(Θ_d) cannot be solved in $2^{(c_{d,2}-\epsilon)\cdot|V|}$ time for any $\epsilon > 0$ unless r-ETH is false, and the result follows by choosing d such that $c_{d,2} \geq c$. We know that $\alpha \cdot \log(d) \leq c_{d,2}$ so it is sufficient to choose a d such that $\alpha \cdot \log(d) \geq c$, e.g. $d = 2^{\lceil \frac{c}{\alpha} \rceil}$. □

Thus, assuming the r-ETH, there cannot exist an algorithm solving CSP(Γ) in $O(c^{|V|})$ time for every finite equality language Γ. This can be strengthened even further for infinite equality languages, and we will show the existence of Γ such that CSP(Γ) is not solvable in $O(2^{o(|V|\log|V|)})$ time without contradicting the ETH. In contrast, the second statement of Theorem 3 is only valid under the much stronger SETH, and only if Γ consists of all first-order definable relations over $(\mathbb{N}; =)$. For this lower bound we provide a reduction from the $k \times k$ Independent Set problem: given a graph G over the vertex set $\{1, \ldots, k\} \times \{1, \ldots, k\}$ (where k is part of the input), is there an independent set of size k in G with exactly one element from each row? One may view this problem as a variant of the standard Independent Set problem where the vertices are the elements of a $k \times k$ table and one wants to find an independent set that contain exactly one element from each row. The following lower bound is known under the ETH [29].

Theorem 6 *$k \times k$ Independent Set is not solvable in $2^{o(k \log k)}$ time unless the ETH is false.*

For $n \geq 1$ define $R_n(y, x_1, \ldots, x_n) \equiv y = x_1 \vee y = x_2 \vee \cdots \vee y = x_n$, and let $R(x, y, z, w) \equiv x \neq y \vee z \neq w$. Let $\Gamma_{\inf}$ be the infinite equality language $\{\neq, R, R_1, R_2, \ldots\}$.

Theorem 7 *CSP($\Gamma_{\inf}$) cannot be solved in $2^{o(|V|\log|V|)}$ time unless the ETH is false.*

Proof To prove the result, we present a polynomial-time reduction from $k \times k$ Independent Set to CSP($\Gamma_{\inf}$) such that the resulting CSP($\Gamma_{\inf}$) instance only contains $2k$ variables. Let $G = (V, E)$ denote an arbitrary graph where $V = \{1, \ldots, k\} \times$

$\{1, \ldots, k\}$. We then begin by introducing k variables $a_1, \ldots, a_k$ together with the constraints $a_i \neq a_j$, $1 \leq i < j \leq k$. Second, for each row $1 \leq i \leq k$ in G, introduce a variable x_i and the constraint $R_k(x_i, a_1, \ldots, a_k)$. This constraint ensures that x_i equals one of the variables $a_1, \ldots, a_k$. Third, for each edge $e = ((a, b), (c, d)) \in E$, introduce the constraint $R(x_a, a_b, x_c, a_d)$. This constraint guarantees that both endpoints of an edge are not put into the independent set simultaneously. □

Hence, we cannot even hope to solve CSP(Γ) in $O(c^{|V|})$ time for *any* c when Γ is allowed to be infinite. Furthermore, since an equality CSP is always solvable in $2^{O(|V| \log |V|)}$ time [18], the bound in Theorem 7 is asymptotically tight.

Thus, the distinction between finite and infinite languages seems to be rather important in the context of equality CSPs, but if one considers slightly richer structures than $(\mathbb{N}; =)$ then significantly stronger bounds can be obtained also for finite languages. Let $\prec \subseteq D^2$ denote a binary relation over a set D and let $\succ$ denote its converse where $x \succ y$ holds if and only if $y \prec x$ holds. We say that $\prec$ is an *acyclic order* if there does not exist any finite subset $\{d_1, \ldots, d_k\} \subseteq D$ such that $d_1 \prec d_2 \prec \cdots \prec d_{k-1} \prec d_k \prec d_1$. Acyclic orders are irreflexive (i.e. they do not contain any element d such that $d \prec d$) by definition. We say that $\prec$ is a *strict partial order* if it is irreflexive and for arbitrary $d, d', d'' \in D$: $d \prec d'$ and $d' \prec d''$ imply $d \prec d''$ (transitivity). Note that these two properties also ensure that $\prec$ is antisymmetric, i.e. if $d \prec d'$, then $d' \prec d$ does not hold. We say that $\prec$ is a *strict total order* if $\prec$ is a strict partial order and it is a *connex* relation, i.e. for arbitrary distinct $d, d' \in D$, either $d \prec d'$ or $d' \prec d$ holds. Finally, we say that $\prec$ contains *unbounded total orders* if for every $k \in \mathbb{N}$, there exists a subset $L \subseteq D$ such that $|L| \geq k$ and $\prec$ is a strict total order on L.

Example 2 The less-than relation $<$ over $\mathbb{Q}$ is an acyclic order containing unbounded total orders. For the latter property, simply observe that $<$ is a strict total order $\{1, \ldots, k\}$. However, there exists a *wealth* of examples of acyclic orders containing unbounded total orders in the artificial intelligence literature, especially in combination with qualitative reasoning problems, e.g., temporal and spatial reasoning problems such as *Allen's interval algebra* and the *region connection calculus*. For many additional examples of this kind, see e.g. the survey by Dylla et al [15].

The following result is a significant strengthening of Theorem 11 in [19].

Theorem 8 *Let $\prec \subseteq D^2$ be an acyclic order that contains unbounded total orders. Then, there exists a constraint language Γ such that*

1. *Γ is finite,*
2. *Γ is first-order definable in $(D; \prec)$ (even with quantifier-free definitions), and*
3. *CSP(Γ) is not solvable in $2^{o(n \log n)}$ time unless the ETH is false.*

Proof Let $\Gamma = \{\prec, R, S\}$ where

- $R(x, y) \equiv x \prec y \vee y \prec x$ and
- $S(x, a, b, y, c, d) \equiv x \prec a \vee b \prec x \vee y \prec c \vee d \prec y$.

Clearly, Γ is finite and (quantifier-free) first-order definable in $(D; \prec)$. Assume that CSP(Γ) can be solved in $2^{o(n \log n)}$ time. We show how to polynomial-time reduce $k \times k$

Independent Set to CSP(Γ) in a way such that only $O(k)$ variables are used. Hence, $k \times k$ Independent Set can be solved in $2^{o(k \log k)}$ time and this contradicts the ETH via Theorem 6.

Let $G = (V, E)$ be an arbitrary instance of $k \times k$ Independent Set. Introduce $2k + 1$ fresh variables $y_1, \ldots, y_k, t_1, \ldots, t_{k+1}$. The idea behind these variables is that y_i, $1 \leq i \leq k$, points out the vertex in row i that is to be included in the independent set. This is done with the aid of variables $t_1, \ldots, t_{k+1}$. Informally speaking, if y_i "lies between" t_j and t_{j+1}, then we will put the jth vertex on row i into the independent set. Now, let $V_1 = \{y_1, \ldots, y_k, t_1, \ldots, t_{k+1}\}$ and

$$C_1 = \{t_i \prec t_j \mid 1 \leq i < j \leq k\}.$$

Since $\prec$ is a acyclic order that contains unbounded total orders, we know that $I_1 = (V_1, C_1)$ is satisfiable.

In every solution s to I_1, it holds that $s(t_i) \prec s(t_j)$ when $1 \leq i < j \leq k + 1$. Constrain each y_i, $1 \leq i \leq k$, as follows:

- $t_1 \prec y_i$,
- $R(y_i, t_j)$ for $2 \leq j \leq k$, and
- $y_i \prec t_{k+1}$.

Let C_2 denote the corresponding set of constraints and let $I_2 = (V_1, C_1 \cup C_2)$. It is easy to verify that in every solution s to I_2 and for each $1 \leq j \leq k$, the variable y_j, satisfies $s(t_i) \prec s(y_j) \prec s(t_{i+1})$ for exactly one $1 \leq i \leq d$. We will interpret this as "the ith vertex on row j is chosen for inclusion in the independent set". Note that a solution always exists to I_2 since $\prec$ is a acyclic order that contains unbounded total orders.

For each edge $\{(x_a, x_b), (x_c, x_d)\}$ in E, we introduce the following constraint

$$S(y_a, t_b, t_{b+1}, y_c, t_d, t_{d+1}).$$

With the given interpretations of y_a, t_b, y_c, t_d, this constraint implies that we cannot simultaneously choose vertex b on row a and vertex d on row c for inclusion in the independent set. Let C_3 denote the resulting set of constraints and let $I_3 = (V_1, C_1 \cup C_2 \cup C_3)$. Given the explanations above, it is easy to verify that I_3 is satisfiable if and only if G is a yes-instance. We conclude the proof by noting that I_3 can be computed in polynomial time and contains $O(k)$ variables. □

3.2 Satisfiability Modulo Theories

We will now consider a problem which is related to equality CSPs, for which we rather effortlessly can obtain lower bounds by reducing from CSP(Γ_{inf}). *Satisfiability modulo theories* (SMT) is a decision problem for logical formulas with respect to a given background theory. The logical formulas are expressed in classical first-order logic with equality. However, it is quite common to not use the full power of this framework; for instance, a frequent restriction is to require that the formulas are quantifier-free (and we will use this fragment ourselves below). An accessible introducion to SMT can be

found in the survey by Barrett et al. [1]. Let SMT($\mathcal{T}$) be the problem of determining whether a first-order formula (with respect to a background theory $\mathcal{T}$) is satisfiable, and let SMT$_\forall(\mathcal{T})$ be the subproblem where universal quantifiers are not allowed. We can then readily prove a matching lower bound valid for *any* background theory $\mathcal{T}$.

Theorem 9 *SMT$_\forall(\emptyset)$ cannot be solved in $2^{o(|V|\log|V|)}$ time unless the ETH is false.*

Proof We present a polynomial-time reduction from CSP(Γ_{inf}) which does not introduce any fresh variables. Let (V, C) be an instance of CSP(Γ_{inf}), where $V = \{x_1, \ldots, x_k\}$ and $C = \{c_1, \ldots, c_p\}$. Define F to be the formula $\exists x_1 \ldots \exists x_k \colon F_1 \wedge \cdots \wedge F_p$ where

- $F_i = (\neg(x = y))$ if $c_i = x \neq y$,
- $F_i = (y = x_1 \vee y = x_2 \vee \ldots \vee y = x_n)$ if $c_i = R_n(y, x_1, \ldots, x_n)$, and
- $F_i = (\neg(x = y) \vee \neg(z = w))$ if $c_i = S(x, y, z, w)$.

It is obvious that F is true if and only if (V, C) has a solution, that F can easily be constructed in polynomial time, and that F contains as many variables as there are variables in V. The result then follows from Theorem 7. □

SMT$_\forall(\emptyset)$ is often referred to as *equality logic* and this problem is important in, for instance, hardware verification [14]. In fact, a slightly more expressive logic known as *the logic of equality with uninterpreted functions* (EUF) is extensively used in hardware verification. There are several known algorithms that solve EUF in $O(|V|!) = 2^{O(|V|\log|V|)}$ time — see, for instance, the discussion in Sect. 12 in the article by Rodeh & Strichman [32]. We conclude, with the aid of Theorem 9, that such algorithms are close to optimal.

To present another optimality result in SMT, we consider the well-known *unit two variable per inequality* (UTVPI) class of constraints, i.e., SMT$_\forall$(UTVPI) where UTVPI for each integer b and coefficients $c_1, c_2 \in \{-1, 1\}$ contains the relation $\{(x, y) \in \mathbb{Z}^2 \mid c_1 \cdot x + c_2 \cdot y \geq b\}$. The UTVPI class has many applications in, for instance, abstract interpretation, spatial databases, and theorem proving (cf. Schutt and Stuckey [37] and the references therein). It is known [38] that SMT$_\forall$(UTVPI) can be solved in $2^{O(|V|\log d)}$ time where $d = 2|V|(b_{\max}+1)+1$ and $b_{\max}$ is the maximum over the absolute values of constant terms in the constraints. Using Theorem 9 we can prove that this algorithm is close to optimal.

Theorem 10 *SMT$_\forall$(UTVPI) cannot be solved in $2^{o(|V|\log d)}$ time unless the ETH is false.*

Proof Assume there is an algorithm A that solves SMT$_\forall$(UTVPI) in $2^{o(|V|\log d)}$ time. The formulas constructed in Theorem 9 are SMT$_\forall$(UTVPI) formulas (degenerate ones, though, since they do not contain UTVPI constraints). Thus, $b_{\max}$ for this class X of formulas is 0, implying that A can solve SMT$_\forall$(UTVPI) restricted to X in $2^{o(n\log n)}$ time, contradicting Theorem 9. □

Difference logic is an interesting fragment of SMT$_\forall$(UTVPI) where only constraints of the form $x - y \geq b$ are allowed. Difference logic has found applications in, for example, verification of timed automata [30] and analysis of dynamic fault trees [40]. The lower bound in Theorem 10 naturally holds also in this restricted case.

3.3 No Easiest NP-Hard Infinite–Domain CSP

Our lower bounds suggest that equality CSPs are rather different from finite-domain CSPs when viewed under the lens of fine-grained complexity. In this section we prove yet another differentiating factor. For each finite A it is known that there exists a constraint language Γ_A with domain A such that CSP(Γ_A) is NP-complete, and if an NP-complete CSP(Δ)[1] over A is solvable in $O(c^{|V|})$ time, then CSP(Γ_A) is solvable in $O(c^{|V|})$ time, too [22]. More generally, if $\mathcal{G}$ is a set of constraint languages over A, we say that CSP(Γ) for some $\Gamma \in \mathcal{G}$ is the *easiest CSP problem in* $\mathcal{G}$ if CSP(Γ) is solvable in $O^*(c^{|V|})$ time whenever CSP(Δ) for $\Delta \in \mathcal{G}$ is solvable in $O^*(c^{|V|})$ time.

Contrary to the finite-domain case we will prove that there does *not* exist an easiest NP-complete equality CSP, unless the ETH is false. In order to prove this, we show that for every $c > 1$ there exists an equality language Γ_c such that CSP(Γ_c) is NP-complete but solvable in $O^*(c^{|V|})$ time. First, recall from Example 1 that the ternary relation $S = \{(x, x, y), (x, y, y) \mid x, y \in \mathbb{N}, x \neq y\}$ has an NP-complete CSP. We will show how S can be extended with additional arguments in order to decrease the time complexity of the resulting CSP. If $\mathbf{v} = (v_1, \ldots, v_k)$ and $\mathbf{w} = (w_1, \ldots, w_k)$ are two k-ary tuples of variables, x is a variable, and R is a binary relation, then we write $R(x, \mathbf{v})$ for $\bigwedge_{1 \leq i \leq k} R(x, v_i)$, $R(\mathbf{v}, \mathbf{w})$ for $\bigwedge_{1 \leq i, j \leq k} R(v_i, w_j)$, and $R(\mathbf{v})$ for $\bigwedge_{1 \leq i, j \leq k, i \neq j} R(v_i, v_j)$.

For arbitrary $k \geq 1$ now define

$$
\begin{aligned}
S_1^k(x, y, z, \mathbf{v}, \mathbf{w}) &\equiv \bigwedge_{s \in \{x,y,z\}, \mathbf{t} \in \{\mathbf{v}, \mathbf{w}\}} \mathrm{R}_{\neq}(s, \mathbf{t}) \wedge \mathrm{R}_{\neq}(\mathbf{v}, \mathbf{w}), \\
S_2^k(x, y, z, \mathbf{v}, \mathbf{w}) &\equiv x = y \wedge y \neq z \wedge \mathrm{Eq}(\mathbf{v}) \wedge \mathrm{R}_{\neq}(\mathbf{w}), \\
S_3^k(x, y, z, \mathbf{v}, \mathbf{w}) &\equiv x \neq y \wedge y = z \wedge \mathrm{R}_{\neq}(\mathbf{v}) \wedge \mathrm{Eq}(\mathbf{w}), \text{ and} \\
S^k(x, y, z, \mathbf{v}, \mathbf{w}) &\equiv S_1^k(x, y, z, \mathbf{v}, \mathbf{w}) \wedge \left(S_2^k(x, y, z, \mathbf{v}, \mathbf{w}) \vee S_3^k(x, y, z, \mathbf{v}, \mathbf{w}) \right)
\end{aligned}
$$

where $\mathbf{v} = (v_1, \ldots, v_k)$ and $\mathbf{w} = (w_1, \ldots, w_k)$ are two distinct k-ary tuples of variables.

The general idea behind the relation S^k is that we want to take an existing relation S yielding an NP-hard CSP and add a number of variables, depending on the given parameter k, so that these variables depend on the original variables from S but cannot be identified with each other. The latter point is important since it allows us to construct a branching algorithm with a sufficiently good branching factor.

It is straightforward to verify that the problem CSP($\{S^k\}$) is NP-complete with the aid of Theorem 1 since $S \in \langle \{S^k\} \rangle$. We will now prove that the fine-grained complexity of CSP($\{S^k\}$) decreases with increasing k, in the following sense.

Theorem 11 *Let $c > 1$. Then there exists k such that CSP($\{S^k\}$) is solvable in $O^*(c^{|V|})$ time.*

Proof We will present an algorithm Y for CSP($\{S^k\}$) which runs in $O^*(2^{\frac{n}{k}})$ time. The claim then follows from choosing a sufficiently large $k \geq \frac{1}{\log c}$. Thus, choose $k \geq 1$

[1] For technical reasons Δ contains all unary relations over A.

Algorithm $Y((V, C), L)$

1. If L is inconsistent, return no.
2. If L is consistent and $C = \emptyset$, return 'yes'.
3. Pick a constraint $S^k(x, y, z, \mathbf{v}, \mathbf{w}) \in C$ where $\mathbf{v} = (v^1, \ldots, v^k)$ and $\mathbf{w} = (w^1, \ldots, w^k)$.
4. Return 'no' if:
   (a) $|\{x, y, z\}| = 1$,
   (b) $\{x, y, z\} \cap \{v^1, \ldots, v^k, w^1, \ldots, w^k\} \neq \emptyset$,
   (c) $\{v^1, \ldots, v^k\} \cap \{w^1, \ldots, w^k\} \neq \emptyset$, or if
   (d) $|\{v^1, \ldots, v^k\}| < k$ and $|\{w^1, \ldots, w^k\}| < k$.
5. If $|\{v^1, \ldots, v^k\}| < k$ and $|\{w^1, \ldots, w^k\}| = k$ then we identify y with x, v^1 with every variable in $\{v^1, \ldots, v^k\} \setminus \{v^1\}$, remove $S^k(x, y, z, \mathbf{v}, \mathbf{w})$ from C, add
   $$\mathrm{R}_{\neq}(\mathbf{w}), \mathrm{R}_{\neq}(x, \mathbf{v}), \mathrm{R}_{\neq}(x, \mathbf{w}), \mathrm{R}_{\neq}(z, \mathbf{v}), \mathrm{R}_{\neq}(z, \mathbf{w}), \mathrm{R}_{\neq}(x, z)$$
   to L, and jump to step (2).
6. The case $|\{v^1, \ldots, v^k\}| = k$, $|\{w^1, \ldots, w^k\}| < k$, is handled analogously.
7. If none of the above cases apply we proceed as follows.
   (a) If $|\{x, y, z\}| = 2$ then no branching is necessary, and depending on whether $x = y$ or $x \neq y$ we jump to step (b) or step (c) below.
   (b) Identify y with x, v^j $(2 \leq j \leq k)$ with v^1, add
   $$\mathrm{R}_{\neq}(\mathbf{w}), \mathrm{R}_{\neq}(x, \mathbf{v}), \mathrm{R}_{\neq}(x, \mathbf{w}), \mathrm{R}_{\neq}(z, \mathbf{v}), \mathrm{R}_{\neq}(z, \mathbf{w}), \mathrm{R}_{\neq}(x, z)$$
   to L, remove $S^k(x, y, z, \mathbf{v}, \mathbf{w})$ from C, and let $a_1 = Y((V, C), L)$.
   (c) Identify z with y, w^j $(2 \leq j \leq k)$ with w^1, add
   $$\mathrm{R}_{\neq}(\mathbf{v}), \mathrm{R}_{\neq}(x, \mathbf{v}), \mathrm{R}_{\neq}(x, \mathbf{w}), \mathrm{R}_{\neq}(y, \mathbf{v}), \mathrm{R}_{\neq}(y, \mathbf{w}), \mathrm{R}_{\neq}(x, y)$$
   to L, remove $S^k(x, y, z, \mathbf{v}, \mathbf{w})$ from C, and let $a_2 = Y((V, C), L)$.
8. Answer 'yes' if a_1 ='yes' or a_2 ='yes', and otherwise 'no'.

Fig. 1 Algorithm for the Proof of Theorem 11

and let (V, C) be an instance of CSP($\{S^k\}$), where $|V| = n$. Say that a set of inequality constraints L is *consistent* if L, viewed as an instance of CSP($\{\mathrm{R}_{\neq}\}$), is satisfiable, and *inconsistent* otherwise. The consistency of a set of inequality constraints can be determined in polynomial time since CSP($\{\mathrm{R}_{\neq}\}$) is in P (from Example 1). Consider the algorithm Y in Fig. 1. The set L is used to keep track of inequality constraints induced by the constraints in the instance.

For correctness, the algorithm branches on a constraint $S^k(x, y, z, \mathbf{v}, \mathbf{w}) \in C$, and either identifies x with y, or y with z; in the process, it identifies variables and introduces inequality constraints according to the definition of S^k. Furthermore, the algorithm answers 'yes' if and only if it for each constraint $S^k(x, y, z, \mathbf{v}, \mathbf{w}) \in C$ is possible to identify x with y, or y with z, in a non-contradictory way, and answers 'no', otherwise. Concerning time complexity, note first that all variables in $\mathbf{v}$ and $\mathbf{w}$ are distinct, once step (7) is reached. This follows from the tests undertaken in step 4 where we systematically verify that $\{w^1, \ldots, w^k\}$ and $\{v^1, \ldots, v^k\}$ are disjoint and that $|\{w^1, \ldots, w^k\}| = |\{v^1, \ldots, v^k\}| = k$. Furthermore, if (7)(b) or (7)(c) is reached then $|\{x, y, z\}| = 3$, as otherwise the current instance is unsatisfiable ($|\{x, y, z\}| = 1$) or no branching was required ($|\{x, y, z\}| = 2$). Thus, in each branch in step 7 we eliminate k variables via variable identification, which implies that the time complexity is bounded by the recurrence $T(n) = 2T(n-k) + \mathrm{poly}(||I||)$. Thus, algorithm Y has total running time $O^*(2^{\frac{n}{k}})$, and therefore it solves CSP($\{S^k\}$) in $O^*(c^n)$ time for a sufficiently large k. □

We immediately obtain the following corollary.

Corollary 1 *Let $\mathcal{A} = (A; R_1, \ldots, R_k)$ be a relational structure over a countably infinite A. Assume that a first-order reduct Γ of $\mathcal{A}$ is NP-complete if and only if Γ pp-interprets 3-SAT. Let $\mathcal{G} = \{\Gamma \mid \Gamma$ is a first-order reduct of $\mathcal{A}$ and CSP(Γ) is NP-complete$\}$. If $\mathcal{G}$ has an easiest CSP, then the ETH is false.*

Proof For each $c > 1$ there exists a constraint language $\Gamma_c \in \mathcal{G}$ such that CSP(Γ_c) is NP-complete and solvable in $O^*(c^{|V|})$ time (Theorem 11). If $\mathcal{G}$ has an easiest NP-complete problem CSP(Γ) then (1) CSP(Γ) pp-interprets 3-SAT, and (2) CSP(Γ) is solvable in $O^*(c^{|V|})$ time for each $c > 1$. Thus, CSP(Γ) is solvable in subexponential time, but this violates the ETH by Theorem 3.1 in [22]. □

Observe that the class of relational structures considered in Corollary 1 includes the NP-hard cases of the CSP dichotomy conjecture over *finitely bounded homogeneous structures* [2]. It is worth noting that after the Feder-Vardi conjecture on finite-domain CSPs was settled (independently) by Bulatov [13] and Zhuk [41], a large part of the complexity-oriented CSP work has concentrated on homogeneous infinite-domain CSPs.

3.4 Algebra and Fine-Grained Complexity of Equality CSPs

Our lower bounds suggest a large difference in fine-grained complexity between equality CSPs and finite-domain CSPs. In this section we take a different viewpoint and investigate this difference through the lens of universal algebra and *partial clone theory*, with the aim of achieving an algebraic explanation of the results obtained in the previous section. We will see a correspondence to the non-existence of certain relations known as *weak bases*. Via the results from Sect. 3.3 we are first able to give a straightforward proof conditional to the ETH (Theorem 1) which we then strengthen to an unconditional proof (Theorem 14) but which requires more elaborate arguments.

3.4.1 Algebraic Background

The basic setting on the functional side is to consider *partial functions* over a universe A. We view a partial function as a mapping of the form $f : X \to A$ for a set $X \subseteq A^k$ called the *domain* of f, and denoted by domain$(f) = X$. Then a partial function f of arity k is said to be a *partial polymorphism* of an n-ary relation R over A if $f(t_1, \ldots, t_k) \in R$ for each sequence of tuples $t_1, \ldots, t_k \in R$ such that $(t_1[i], \ldots, t_k[i]) \in$ domain(f) for each $1 \leq i \leq n$. If f is total, i.e., f is always defined, then f is simply called a *polymorphism*. If we let pPol(R) be the set of all partial polymorphisms of a relation R, and pPol$(\Gamma) = \bigcap_{R \in \Gamma}$ pPol(R) be the set of all partial polymorphisms of the set of relations Γ, the resulting sets of partial functions are called *strong partial clones*. Similarly, we write Pol(Γ) for the set of all polymorphisms of Γ, and the resulting sets of functions are known as *clones*. If F is a set of (total or partial) functions then we write Inv(F) to denote the set of relations invariant under each function in F.

Let us also briefly mention some properties of strong partial clones. In this context the term strong means that if $f \in \mathrm{pPol}(\Gamma)$ then $f_{|X} \in \mathrm{pPol}(\Gamma)$ for each restriction of f on the domain $X \subseteq \mathrm{domain}(f)$, i.e., $\mathrm{domain}(f_{|X}) = X$ and $f(\mathbf{x}) = f_{|X}(\mathbf{x})$ for each $\mathbf{x} \in X$. More generally, if $X \nsubseteq \mathrm{domain}(f)$ then we let $f_{|X}$ be the restriction of f to the set $\mathrm{domain}(f) \cap X$. Then strong partial clones of the form $\mathrm{pPol}(\Gamma)$ are precisely the *local* strong partial clones over A [34], meaning that $f \in \mathrm{pPol}(\Gamma)$ for a k-ary (partial) function f if $f_{|X} \in \mathrm{pPol}(\Gamma)$ for each finite $X \subseteq A^k$. On the relational side strong partial clones $\mathrm{pPol}(\Gamma)$ correspond to sets of relations closed under pp-definitions without existential quantification, *quantifier-free pp-definitions* (qfpp-definitions). We write $\langle\Gamma\rangle_{\nexists}$ for the smallest set of relations containing Γ which is closed under qfpp-definitions. For ω-categorical structures we then have the following useful correspondence between $\mathrm{Inv}(\cdot)$ and $\mathrm{pPol}(\cdot)$. The theorem follows almost directly from Romov [33], but for completeness we include a proof sketch where the ω-categorical case differs.

Theorem 12 *Let Γ and Δ be two ω-categorical sets of relations over a domain A. Then (1)* $\mathrm{Inv}(\mathrm{pPol}(\Gamma)) = \langle\Gamma\rangle_{\nexists}$ *and (2)* $\Gamma \subseteq \langle\Delta\rangle_{\nexists}$ *if and only if* $\mathrm{pPol}(\Delta) \subseteq \mathrm{pPol}(\Gamma)$.

Proof Since Γ is ω-categorical there for each $n \geq 1$ only exists a finite number of first-order definable relations of arity n (see, e.g., Theorem 6.3.1 in Hodges [17]). The first claim then follows from Proposition 2 in Romov [33] since infinite intersections of relations and direct limits of relations can always be expressed via qfpp-definitions over a finite number of relations from Γ. The second claim then readily follows by standard arguments. □

There is a similar connection between $\mathrm{Pol}(\cdot)$ and $\langle\cdot\rangle$ which we omit since it is not directly useful for our purposes (see, e.g., the introductory textbook by Lau [28]). Theorem 12 then implies that partial polymorphisms determine the fine-grained complexity of CSPs in the following sense, as originally proved by Jonsson et al. for Boolean CSPs [21].

Theorem 13 *Let Γ and Δ be two finite ω-categorical languages. If* $\mathrm{pPol}(\Delta) \subseteq \mathrm{pPol}(\Gamma)$ *then there exists a polynomial-time many-one reduction f from CSP(Γ) to CSP(Δ) such that $f((V, C)) = (V', C')$, $|V'| \leq |V|$, for each instance (V, C) of CSP(Γ).*

The lattice of strong partial clones is uncountable in the Boolean domain [16] and it is to a large extent unexplored. Quite naturally, even less is known for arbitrary finite domains or infinite domains. However, we can simplify the task of analysing strong partial clones by restricting our attention to strong partial clones $\mathrm{pPol}(\Gamma)$ where $\mathrm{Pol}(\Gamma) = C$ for a fixed clone C. It is then of particular interest to determine whether the set of strong partial clones of this form, sometimes called an *interval*, has a largest element.

Definition 2 Let C be a clone over a finite or countably infinite domain A. If there exists a set of relations Γ_w over A such that $\mathrm{Pol}(\Gamma_w) = C$ and $\mathrm{pPol}(\Gamma_w) = \bigcup\{\mathrm{pPol}(\Delta) \mid \mathrm{Pol}(\Delta) = C\}$ then we say that Γ_w is a *weak base* of $\mathrm{Inv}(C)$.

Thus, $\mathrm{pPol}(\Gamma_w)$ is the largest element in $\{\mathrm{pPol}(\Delta) \mid \mathrm{Pol}(\Delta) = C\}$, which on the relational side means that $\Gamma_w \subseteq \langle\Delta\rangle_{\not\exists}$ for every set of relations Δ such that $\mathrm{Pol}(\Delta) = C$. Hence, Γ_w is minimally expressive with respect to qfpp-definitions among the generating sets of $\mathrm{Inv}(C)$, which explains the name "weak base". If A is finite and C can be generated by a finite set of functions over A, then it is known that $\mathrm{Inv}(C)$ has a weak base [36]. For infinite domains the situation differs, and weak bases do not necessarily exist. For both negative and positive examples, see Romov [35].

Might it then be possible that $\langle\Gamma\rangle$ admits a weak base whenever Γ is an equality language? And which implications would that have if $\mathrm{CSP}(\Gamma)$ is NP-complete? Let $\mathcal{E}$ be the set of all first-order definable relations over $(\mathbb{N}; =)$. Now, recall the definition of the relation S from Example 1. It is then known that $S \in \langle\Gamma\rangle$ (and thus, $\mathrm{CSP}(\Gamma)$ is NP-complete) for an equality language Γ if and only if $\langle\Gamma\rangle = \mathcal{E}$ [8]. Thus, we are interested in determining whether $\mathcal{E}$ has a weak base, and we may now observe that the existence of a weak base would have far-reaching implications.

Proposition 1 *If $\mathcal{E}$ has a weak base then the ETH is false.*

Proof Assume that $\mathcal{E}$ has a weak base Γ_w. Assume first that Γ_w is infinite. It is then known that there exists a finite set $\Delta \subseteq \Gamma_w$ such that $\langle\Gamma_w\rangle = \langle\Delta\rangle$, implying also that $\langle\Gamma_w\rangle_{\not\exists} = \langle\Gamma\rangle_{\not\exists}$ (see, e.g., the second condition of Theorem 7.4.2 in Bodirsky [4].

Thus, assume that Γ_w is finite. But then Theorem 13 together with the relations constructed in Theorem 11 implies that $\mathrm{CSP}(\Gamma_w)$ is solvable in $O(c^{|V|})$ time for every $c > 1$. However, then Theorem 3 implies that 3-SAT is solvable in subexponential time, thus contradicting the ETH. □

3.4.2 The Non-Existence of a Weak Base

Due to Proposition 1 we strongly suspect that $\mathcal{E}$ does not have a weak base, but we will see that one can *unconditionally* prove that $\mathcal{E}$ does not have a weak base. In fact, we will prove a fairly general condition which determines the non-existence of a weak base, which is particularly poignant in the relationship of NP-hard CSPs. For a universe A, let $R^A_{\neq} = \{(x, y) \in A^2 \mid x \neq y\}$ be the inequality relation over A.

Lemma 1 *Let Γ be a finite, ω-categorical set of relations over an infinite domain A. If $R^A_{\neq} \in \langle\Gamma\rangle$ then $\langle\Gamma\rangle$ does not admit a weak base.*

Proof Let f be an arbitrary k-ary function over A. Our goal is to show that there for every finite $X \subset \mathrm{domain}(f) = A^k$ exists an equality constraint language Γ such that $\langle\Gamma\rangle = \mathcal{E}$ and such that $f_{|X}$ preserves Γ. If $\langle\Gamma\rangle$ admits a weak base Γ_w then, clearly, $f_X \in \mathrm{pPol}(\Gamma_w)$ for every finite $X \subset A^k$, which implies that $f \in \mathrm{pPol}(\Gamma_w)$ for *every* function f (since $\mathrm{pPol}(\Gamma_w)$ is local). This contradicts the assumption that $R^A_{\neq} \in \langle\Gamma\rangle$ since $R^A_{\neq}$, for example, is not preserved by any constant function over D.

Hence, let $X \subset A^k$ be finite. Let $N = \{d_1, \ldots, d_k \mid (d_1, \ldots, d_k) \in X\}$ be the set of values occuring in tuples in X, and let $|N| = n$. Let $\Gamma = \{R_1, \ldots, R_l\}$ and define the relation R to be the Cartesian product of all relations in Γ, i.e., $R = R_1 \times \ldots \times R_l$. Let m be the arity of the relation R. Clearly, $\langle\{R\}\rangle = \langle\Gamma\rangle$, since Γ can pp-define R

via a conjunction, and R can pp-define each relation in Γ by projecting away every other argument. Define the $(m+n+1)$-ary relation R^n such that

$$R^n(x_1,\ldots,x_m,y_1,\ldots,y_n,y_{n+1}) \equiv$$
$$R(x_1,\ldots,x_m) \wedge \bigwedge_{i,j\in\{1,\ldots,n+1\},i\neq j} R^A_{\neq}(y_i,y_j).$$

This relation is pp-definable by R since we assumed that R can pp-define the inequality relation $R^A_{\neq}$, and since

$$R(x_1,\ldots,x_m) \equiv \exists y_1,\ldots,y_{n+1}\colon R^n(x_1,\ldots,x_m,y_1,\ldots,y_n,y_{n+1}),$$

we also have that $\langle\{R\}\rangle = \langle\{R^n\}\rangle = \langle\Gamma\rangle$. Next, we claim that $f_{|X}$ preserves R^n. Consider any sequence of tuples $t_1,\ldots,t_k \in R^n$. Due to the definition of R^n we then have that $t[m+i] \neq t[m+j]$ for any distinct $i,j \in \{1,\ldots,n+1\}$. Hence, $|\{t[i] \mid 1 \leq i \leq m+n+1\}| > N$, meaning that $f(t_1,\ldots,t_k)$ is undefined, and that f preserves R^n.

Last, assume there exists Γ_w such that $\mathrm{pPol}(\Gamma_w) = \bigcup_{\langle\Delta\rangle=\langle\Gamma\rangle} \mathrm{pPol}(\Delta)$, i.e., that Γ_w is a weak base of $\langle\Gamma\rangle$. Then, by the above construction, $f_{|X} \in \mathrm{pPol}(\Gamma_w)$ for every finite X since $\mathrm{pPol}(\{R^n\}) \subseteq \mathrm{pPol}(\Gamma_w)$, which then implies that $f \in \mathrm{pPol}(\Gamma_w)$ since $\mathrm{pPol}(\Gamma_w)$ is local. Hence, the strong partial clone $\mathrm{pPol}(\Gamma_w)$ would need to contain *all* total functions over A. But then Γ_w cannot be a weak base of $\langle\Gamma\rangle$ since the assumption that $R^A_{\neq} \in \langle\Gamma\rangle = \langle\Gamma_w\rangle$ e.g. implies that Γ_w cannot be preserved by any constant function over A. □

This condition is sufficient to establish non-existence of weak bases in the context of both equality languages and temporal languages.

Theorem 14 *$\mathcal{E}$ does not have a weak base.*

Proof Since $\mathcal{E}$ is the set of *all* first-order definable relations it clearly follows that $R_{\neq} \in \mathcal{E}$. But since all equality languages are ω-categorical, and since $\mathcal{E} = \langle\{S\}\rangle$, the result then directly follows from Lemma 1. □

Observe that Theorem 14 together with Theorem 12 implies that

$$\bigcap\{\langle\Gamma\rangle_{\not\exists} \mid \Gamma \text{ is an equality constraint language}, \langle\Gamma\rangle = \mathcal{E}\} = \langle\mathrm{Eq}\rangle_{\not\exists}.$$

To see this, assume otherwise, i.e., that there exists $R \notin \langle\mathrm{Eq}\rangle_{\not\exists}$ such that $R \in \langle\Delta\rangle_{\not\exists}$ for Γ such that $\langle\Gamma\rangle = \mathcal{E}$. This, however, would imply that $\mathrm{pPol}(\mathrm{Eq}) \supset \mathrm{pPol}(R) \supseteq \bigcup_{\langle\Gamma\rangle=\mathcal{E}} \mathrm{pPol}(\Gamma)$. This contradicts the Proof of Theorem 14 since it is shown that $\bigcup_{\langle\Gamma\rangle=\mathcal{E}}$ contains all (total and partial) functions. One interpretation of this result is that equality languages resulting in NP-hard CSPs have rather little in common with regards to qfpp-definability. For example, we may conclude that not all such languages can qfpp-define the inequality relation $\mathrm{Neq}_{\mathbb{N}}$.

Last, we will show that the non-existence of weak bases is not solely a property of equality CSPs, and that an analogous property can be proven also for temporal CSPs,

i.e., CSP(Γ) where each relation in Γ has a first-order definition in the structure $(\mathbb{Q}; <)$.

This class of CSPs is a strict generalisation of equality CSPs and includes many natural problems, e.g., the *betweenness* problem and the *cyclic ordering problem*. For many other examples, see Bodirsky & Kára [9]. Let $\mathcal{T}$ be the set of all first-order definable relations over $(\mathbb{Q}; <)$. For $x_1, \ldots, x_k \in \mathbb{Q}$, we write $\overrightarrow{x_1 \ldots x_k}$ when $x_1 < \cdots < x_k$. The following dichotomy holds for temporal CSPs.

Theorem 15 (Bodirsky and Kára [9]) *Let $\Gamma \subseteq \mathcal{T}$ be a temporal constraint language. If there is a primitive positive definition of* Betw*,* Cycl*,* Sep*, T_3, $-T_3$, or S in Γ, where*

1. $\text{Betw} = \{(x, y, z) \in \mathbb{Q}^3 \mid \overrightarrow{xyz} \vee \overrightarrow{zyx}\}$,
2. $\text{Cycl} = \{(x, y, z) \in \mathbb{Q}^3 \mid \overrightarrow{xyz} \vee \overrightarrow{yzx} \vee \overrightarrow{zxy}\}$,
3. $\text{Sep} = \{(x_1, y_1, x_2, y_2) \in \mathbb{Q}^4 \mid \overrightarrow{x_1x_2y_1y_2} \vee \overrightarrow{x_1y_2y_1x_2} \vee \overrightarrow{y_1x_2x_1y_2} \vee \overrightarrow{y_1y_2x_1x_2} \vee \overrightarrow{x_2x_1y_2y_1} \vee \overrightarrow{x_2y_1y_2x_1} \vee \overrightarrow{y_2x_1x_2y_1} \vee \overrightarrow{y_2y_1x_2x_1}\}$,
4. $T_3 = \{(x, y, z) \in \mathbb{Q}^3 \mid x = y < z \vee x = z < y\}$, *and*
5. $-T_3 = \{(-x, -y, -z) \mid (x, y, z) \in T_3\}$,

then CSP(Γ) is NP-complete. Otherwise, CSP(Γ) is tractable.

With the help of this classification we can then prove that $\langle\Gamma\rangle$ cannot admit a weak base whenever CSP(Γ) is NP-complete (assuming P $\neq$ NP).

Theorem 16 *Let $\Gamma \subseteq \mathcal{T}$ be a finite temporal language. If Γ pp-defines* Betw*,* Cycl*,* Sep*,* T_3*,* $-T_3$*, or S, then $\langle\Gamma\rangle$ does not admit a weak base.*

Proof We want to apply Lemma 1, and thus need to show that Γ can pp-define the inequality relation $R_{\neq}^{\mathbb{Q}}$ over $\mathbb{Q}$. To prove this it is sufficient to show that Betw, Cycl, Sep, T_3, $-T_3$, and S, can all pp-define the inequality relation, which can be done with straightforward arguments. For example, $R_{\neq}^{\mathbb{Q}}(x, y) \equiv \exists z\colon \text{Betw}(x, y, z)$, and $R_{\neq}^{\mathbb{Q}}(x, y) \equiv \exists z\colon \text{Cycl}(x, y, z)$. The result then directly follows from Lemma 1. □

4 Upper Bounds for Equality CSPs and Reducts of Unary Structures

The lower bounds established in Sect. 3 suggest that we cannot construct an $O(c^{|V|})$ time algorithm ($c > 1$) which is applicable to arbitrary equality languages. However, if we fix a finite equality language Γ, this still leaves the possibility of constructing an $O(c^{|V|})$ time algorithm for a constant c depending on Γ. In this section we tackle this problem, and the more general problem of constructing faster exponential-time algorithms for CSP(Γ) whenever Γ is a finite unary reduct. We begin in Sect. 4.1 by constructing an improved algorithm for the case when Γ is a finite equality language, and in Sect. 4.2 consider the more involved case of reducts of unary structures.

Algorithm $A((V, C))$:

1. Assume $V = \{x_1, \ldots, x_n\}$.
2. Define $s\colon V \to \{1, \ldots, n\}$ such that $s(x_i) = i$.
3. If s is a solution to I, then return 'yes'.
4. If s is not a solution to I and $|V| = 1$, then return 'no'.
5. Arbitrarily choose a constraint $R(x_{i_1}, \ldots, x_{i_p})$ that is not satisfied by s.
6. For each $1 \leq j < k \leq p$, let $I_{j,k}$ denote the instance obtained by identifying x_{i_j} with x_{i_k} in I. If $A(I_{j,k}) =$ 'yes' for some $I_{j,k}$, then return 'yes'.
7. Return 'no'.

Fig. 2 Algorithm for equality languages

4.1 An Algorithm for Finite Equality Languages

We begin by describing a novel algorithm for CSP(Γ), where Γ is a finite equality language with maximum arity α, with a running time of $O^*((\frac{\alpha(\alpha-1)}{2})^{|V|})$. Thus, the algorithm runs in $O^*(c^{|V|})$ time for a constant c depending on Γ, which is a significant improvement over the algorithm proposed by [18] which solves CSP(Γ) in $O^*(2^{|V| \cdot \log(\frac{0.792|V|}{\ln(|V|+1)})})$ time.

Theorem 17 *The CSP of an arbitrary finite equality language Γ can be solved in $O^*\left(\left(\frac{\alpha(\alpha-1)}{2}\right)^{|V|}\right)$ time where $\alpha = \max\{ar(R) \mid R \in \Gamma\}$.*

Proof Consider the algorithm A for instances of CSP(Γ) presented in Fig. 2. We begin by proving correctness by induction over $|V| = n$. If $n = 1$, then the tests in steps (3) and (4) provide the correct answer. Assume the algorithm is correct when $n > 1$. Let $I = (V, C)$ be an instance where $|V| = n + 1$. First, assume that I has an injective solution. Then it is readily verified that $f\colon V \to \{1, \ldots, |V|\}$ defined as $f(x_i) = i$ for each $x_i \in V = \{x_1, \ldots, x_{|V|}\}$, is a solution to I as well (in technical terms this follows from the well-known fact that the automorphisms of Γ is the full symmetric group [8]). Hence, the algorithm answers 'yes' via step (3). Otherwise I does not have an injective solution and at least one constraint $c = R(x_{i_1}, \ldots, x_{i_p}) \in C$ is not satisfied by the function s. This implies that (at least) two variables in $\{x_{i_1}, \ldots, x_{i_p}\}$ must be assigned the same value. This is systematically tested in step (6), and the correctness follows from the inductive hypothesis.

Concerning the time complexity, it is bounded from above by the recurrence $T(n) = \frac{\alpha(\alpha-1)}{2} \cdot T(n-1) + \text{poly}(||I||)$ since $i_p \leq \alpha$ for each possible choice of constraint $R(x_{i_1}, \ldots, x_{i_p})$. Thus, $T(n) \in O^*((\frac{\alpha(\alpha-1)}{2})^n)$, and we get the desired bound on the time complexity. □

4.2 An Algorithm for Finite Reducts of Unary Structures

We recall that a structure $\mathbf{A} = (A; U_1, U_2, \ldots, U_k)$ is *unary* if $U_1, U_2, \ldots, U_k$ are unary relations. The classical complexity of the constraint satisfaction problem for finite first-order reducts of unary structures has been thoroughly analysed by Bodirsky & Mottet [10] and Bodirsky & Bodor [5]: they prove that such problems are either

polynomial-time solvable or NP-complete. We refer the reader to their articles for more background information about unary structures and their reducts.

Throughout this section we let $\Theta = (\mathbb{N}; U_1, \ldots, U_k)$, $k \geq 1$, be an arbitrary unary structure where each $U_i \subseteq \mathbb{N}$, and we let $\Gamma = \{R_1, \ldots, R_m\}$ be a finite first-order reduct of Θ. We can (without loss of generality) focus on structures with a countably infinite domain since every reduct of a unary structure has the same CSP as a reduct of a structure on a countably infinite domain. Since Θ admits quantifier-elimination, and since Γ is finite, we may without loss of generality assume that each R_i is defined via a DNF formula where an atom consists of either a unary relation from Θ or an equality constraint. Let α denote the maximum arity of Γ, i.e. $\alpha = \max\{ar(R_1), \ldots, ar(R_m)\}$.

Our algorithm for CSP(Γ) is based on the following steps. First, we show that there for each instance $I = (V, C)$ of CSP(Γ) exists a particular set of functions F with $c^{|V|}$ elements (where c is a constant that only depends on Γ). These functions can be viewed as "high-level descriptions" of the solution we are searching for. Second, we prove that for each $f \in F$, one can construct an instance I_f of CSP(Γ_{eq}) where Γ_{eq} is a finite equality language that only depends on the choice of Γ. The instances I_f are constructed in such a way that I is satisfiable if and only if I_f is satisfiable for some $f \in F$.

We proceed with a few definitions. For every set $S \subseteq \mathbb{N}$ we denote the complement $\mathbb{N} \setminus S$ of S by $\bar{S}$. Define $U(S)$, $S \subseteq \{1, \ldots, k\}$, such that

$$U(S) = \bigcap_{i \in S} U_i \cap \bigcap_{i \notin S} \bar{U}_i,$$

and let $\mathbf{S} = \{U(S) \mid S \subseteq \{1, \ldots, k\}\}$. One may view the set $\mathbf{S}$ as a "basis" for $U_1, \ldots, U_k$ in the sense that each U_i is the union of some elements in $\mathbf{S}$. Let

$$K = \bigcup \{U(S) \mid S \subseteq \{1, \ldots, k\} \text{ and } U(S) \text{ is finite}\}$$

and

$$\mathbf{U} = \{\{e\} \mid e \in K\} \cup \{U(S) \mid S \subseteq \{1, \ldots, k\} \text{ and } U(S) \text{ is infinite}\}.$$

The set $\mathbf{U}$ can be viewed as a refinement of $\mathbf{S}$: it is still a basis for $U_1, \ldots, U_k$ but it contains more elements. The functions in the set F that we briefly discussed earlier have the set $\mathbf{U}$ as their codomain.

Lemma 2 *The following statements are true.*

1. $\mathbf{U}$ *is a partitioning of* $\mathbb{N}$*,*
2. $\mathbf{U}$ *is finite, and*
3. *for every* U_i, $1 \leq i \leq k$, *there exist* $S_1, \ldots, S_p \in \mathbf{U}$ *such that* $U_i = \bigcup_{j=1}^{p} S_j$.

Proof We first make the following claims concerning the set $\mathbf{S}$.

1. $\mathbf{S}$ is a partitioning of $\mathbb{N}$,
2. $\mathbf{S}$ is finite, and

3. for every U_i, $1 \leq i \leq k$, there exist $S_1, \ldots, S_p \in \mathbf{S}$ such that $U_i = \bigcup_{j=1}^{p} S_j$.

We prove each case in turn.

1. Arbitrarily choose $p \in \mathbb{N}$. Let $S = \{i \mid p \in U_i\}$ and note that $p \in U(S)$. In particular, if $S = \emptyset$, then $U(\emptyset) = \bigcap_{i=1}^{k} \bar{U}_i$ and $p \in U(\emptyset)$. We conclude that every element in $\mathbb{N}$ appears in at least one of the sets $U(S)$. Assume, with the aim of getting a contradiction, that $p \in U(S)$, $p \in U(S')$, and $S \neq S'$ where $S' \subseteq \{1, \ldots, k\}$. It is clear that the only sets among $U_1, \ldots, U_k, \bar{U}_1, \ldots, \bar{U}_k$ that contain p are U_i, $i \in S$, and $\bar{U}_j$, $j \in \{1, \ldots, k\} \setminus S$. Thus, there is at least one set in

$$X = \{U_i \mid i \in S'\} \cup \{\bar{U}_j \mid j \in \{1, \ldots, k\} \setminus S'\}$$

that does not contain p. We know that $U(S') = \bigcap X$ so $p \notin U(S')$ and this leads to a contradiction.
2. $\mathbf{S}$ contains at most 2^k elements.
3. Arbitrarily choose U_i, $1 \leq i \leq k$. Let $T = \bigcup\{X \mid X \subseteq U_i,\ X \in \mathbf{S}\}$ and note that $T \subseteq U_i$. We show that $U_i \subseteq T$ and conclude that there exists a set of elements in $\mathbf{S}$ whose union equals U_i. Arbitrarily choose $e \in U_i$ and assume to the contrary that $e \notin T$. There exists exactly one set $E \in \mathbf{S}$ that contains e since $\mathbf{S}$ is a partitioning of $\mathbb{N}$. We know that $E = U(S)$ for some $S \subseteq \{1, \ldots, k\}$ by the definition of $\mathbf{S}$. If $i \in S$, then $E \subseteq U_i$ and $\{e\} \subseteq E \subseteq U_i \subseteq T$ which leads to a contradiction. Hence, $i \notin S$ and $E = E \cap \bar{U}_i$ by the definition of $U(S)$. This implies that $e \notin E$ since $e \in U_i$ and this contradicts the choice of E.

The statements for the set $\mathbf{U}$ now become straightforward consequences. The family of sets $\mathbf{U}$ is still a partitioning of $\mathbb{N}$ since we have only "refined" the finite sets of $\mathbf{S}$ into single-element sets. Since $\mathbf{S}$ is a finite set, $\mathbf{U}$ is finite, too. Finally, every U_i, $1 \leq i \leq k$, can be expressed as a union of elements in $\mathbf{U}$ since this is possible in $\mathbf{S}$. □

Let us remark that a partition where every part is either infinite or one-element (such as $\mathbf{U}$) is called a *stabilised partition* in the terminology of Bodirsky & Mottet [10]. We define the algorithm D (see Fig. 3) for instances (V, C) of CSP(Γ) and functions $f\colon V \to \mathbf{U}$. Algorithm D checks whether a given instance (V, C) has a solution that *respects* the function f: we say that a solution $g\colon V \to \mathbb{N}$ to (V, C) respects f if $g(x) \in f(x)$ for all $x \in V$. If a conjunct becomes empty, then we view it (as usual) as satisfiable and it can be removed. If a disjunction becomes empty, then it is not satisfiable and the algorithm can immediately report that the instance is not satisfiable. The algorithm A that appears within D is the algorithm for equality languages presented in Sect. 4.1. It will only be applied to constraints that are based on equality relations with arity at most α. We let Γ_{eq} denote this set of equality relations. The language Γ_{eq} is finite so algorithm A solves CSP(Γ_{eq}) in time $O^*\left(\left(\frac{\alpha(\alpha-1)}{2}\right)^{|V|}\right)$ by Theorem 17.

Our aim is now to prove that an instance $I = (V, C)$ of CSP(Γ) is satisfiable if and only if there exists a function $f\colon V \to \mathbf{U}$ such that $D(I, f)$ answers 'yes'. First of all, we verify that the computation of the instance (V, C'') is an instance of CSP(Γ_{eq}),

Algorithm $D((V, C), f)$:

1. For every $(\ell_1 \vee \ldots \vee \ell_p) \in C$, where $\ell_i = \kappa_{i_1} \wedge \ldots \wedge \kappa_{i_k}$:
   (a) Remove ℓ_i, $1 \leq i \leq p$, if there exists $i_1 \leq i_j \leq i_k$ such that $\kappa_{i_j} = U_q(x)$ where $f(x) \cap U_q = \emptyset$,
   (b) Remove ℓ_i, $1 \leq i \leq p$, if there exists $i_1 \leq i_j \leq i_k$ such that $\kappa_{i_j} = \neg U_q(x)$ where $f(x) \subseteq U_q$.
   (c) Remove κ_{i_j} from ℓ_i if $\kappa_{i_j} = U_q(x)$ and $f(x) \subseteq U_q$.
   (d) Remove κ_{i_j} from ℓ_i if $\kappa_{i_j} = \neg U_q(x)$ and $f(x) \cap U_q = \emptyset$.
2. Let C' denote the new set of formulas.
3. Let $C'' = C' \cup \{x \neq y \mid f(x) \neq f(y), \text{ and } x, y \in V\}$
   $\cup \{x = y \mid f(x) = f(y) = S, \ |S| = 1, \text{ and } x, y \in V\}$.
4. Return $A((V, C''))$.

Fig. 3 Algorithm for reducts of unary structures

implying that the call to algorithm A in step (4) is valid. For this purpose it is sufficient to show that $S \subseteq U_i$ or $S \cap U_i = \emptyset$ for each $U_i \in \{U_1, \ldots, U_k\}$ and $S \in \mathbf{U}$, since the filtering in step (1) then guarantees that any constraint involving U_i is replaced by a constraint over Γ_{eq}.

Lemma 3 *Arbitrarily choose U_i, $1 \leq i \leq k$, and a set $S \in \mathbf{U}$. Either $S \subseteq U_i$ or $S \cap U_i = \emptyset$.*

Proof There exist $S_1, \ldots, S_p \in \mathbf{U}$ such that $U_i = \bigcup_{j=1}^{p} S_j$ by the third statement of Lemma 2. Since $\mathbf{U}$ is a partitioning of $\mathbb{N}$ (by the first statement of Lemma 2), this decomposition is unique. If $S \in \{S_1, \ldots, S_p\}$, then $S \subseteq U_i$. Otherwise, $S \cap U_i = \emptyset$. □

We continue the correctness proof by establishing a close connection between (V, C) and (V, C'').

Lemma 4 *Let (V, C) be an instance of CSP(Γ), let $f : V \to \mathbf{U}$, and let (V, C'') be the instance computed in step (3) of the algorithm $D((V, C), f)$. Then (V, C) has a solution $g : V \to \mathbb{N}$ that respects f if and only if the instance (V, C'') has such a solution.*

Proof We begin by showing that (V, C) has a solution $g : V \to \mathbb{N}$ which respects f if and only if the instance (V, C') computed in step 1 of the algorithm has such a solution. Therefore, first assume that (V, C) has a solution g that respects f. If a formula in C contains the atom $U_i(x)$ (respectively, $\neg U_i(x)$) and $f(x) \cap U_i = \emptyset$ (respectively, $f(x) \subseteq U_i$), then we can safely remove the entire conjunction containing $U_i(x)$ since it cannot be satisfied by a solution that respects f (such as g). Furthermore, every atom $U_i(x)$ (respectively, $\neg U_i(x)$) such that $f(x) \subseteq U_i$ (respectively, $f(x) \cap U_i = \emptyset$) is vacuously satisifed by any solution that respects f so such atoms can be removed. We conclude that g is a solution to (V, C').

Second, assume that (V, C') has a solution $g : V \to \mathbb{N}$ that respects f. First note that the atoms that are removed in step 1(c) and 1(d) are satisfied by the solution f. Since g respects f, these atoms are satisfied by g, too. Thus, if we take all constraints in C' and extend them with the conjuncts and atoms that were removed in step 1, then g is a solution to this set of constraints. Note here that adding back the removed

conjuncts only makes the instance easier in the sense that it is satisfied by a potentially larger set of variable assignments. Clearly, the new set of constraints equals C and we conclude that g is a solution to (V, C).

Now, assume that $g\colon V \to \mathbb{N}$ is a solution to (V, C') that respects f. The additional constraints $\{x \neq y \mid f(x) \neq f(y)\}$ are always satisfied when we are only interested in solutions that respect f—this follows from the fact that $\mathbf{U}$ is a partitioning of $\mathbb{N}$ by the first statement of Lemma 2. The constraints $\{x = y \mid f(x) = f(y) = S \text{ and } |S| = 1\}$ are always satisfied when the domain of a variable consists of a single element. Thus, g is a solution to (V, C'').

Last, assume that (V, C'') has a solution $g\colon V \to \mathbb{N}$ that respects f. Since $C' \subseteq C''$, C' can be viewed as a relaxation of C''. Consequently, g is a solution to (V, C') which respects f. □

Lemma 4 gives us a straightforward way of proving the correctness of algorithm D.

Lemma 5 *Let (V, C) be an instance of CSP(Γ), and let $f\colon V \to \mathbf{U}$. Then the algorithm D accepts $((V, C), f)$ if and only if (V, C) has a solution that respects f.*

Proof For the first direction, assume that D accepts the instance $((V, C), f)$. This implies that there exists a solution $g\colon V \to \mathbb{N}$ to the instance (V, C''). Let $D_S = \{g(x) \mid f(x) = S, x \in V\}$ for every $S \in \mathbf{U}$, i.e. D_S contains the values that g assigns to the variables satisfying $f(x) = S$. We make two observations concerning the sets D_S.

1. $D_S \cap D_{S'} = \emptyset$ whenever $S \neq S'$. This a consequence of the construction of C'': the constraint $x \neq y$ is in C'' whenever $f(x) \neq f(y)$.
2. $|D_S| \leq 1$ if $|S| = 1$. Once again, this a consequence of the construction of C'': the constraint $x = y$ is in C'' whenever $f(x) = f(y) = S$ and $|S| = 1$.

These two observations imply that there exist injective functions h_S from D_S to S for all $S \in \mathbf{U}$ (recall that a set in $\mathbf{U}$ is either infinite or one-element). The sets in $\{D_S \mid S \in \mathbf{U}\}$ are pairwise disjoint and so are the sets in $\mathbf{U}$. Hence, there exists an injective function $h\colon \mathbb{N} \to \mathbb{N}$ such that $\{h(d) \mid d \in D_S\} \subseteq S$ for all $S \in \mathbf{U}$. We see that the function $g'\colon V \to \mathbb{N}$ defined by $g'(x) = h(g(x))$ is a solution to (V, C'') that respects f. By Lemma 4, there is a solution to (V, C) that respects f.

For the other direction, assume that D does not accept the instance $((V, C), f)$. This implies that there does not exist any solution to the instance (V, C''). By Lemma 4, there is no solution $g\colon V \to \mathbb{N}$ to (V, C) that respects f. □

We can now state and prove the main result by combining the results presented in this section.

Theorem 18 *CSP(Γ) can be solved in $O^*((|\mathbf{U}| \cdot \frac{\alpha(\alpha-1)}{2})^{|V|})$ time.*

Proof We begin by proving that algorithm D runs in $O^*((\frac{\alpha(\alpha-1)}{2})^{|V|})$ time. Let $I = ((V, C), f)$ denote an arbitrary input instance. First of all, each test performed in step 1 can be performed in constant time since the constraint language Γ is fixed and $\mathbf{U}$ is

finite: the information needed for verifying if $f(x) \cap U_i = \emptyset$ and $f(x) \subseteq U_i$ can be precomputed and stored in a finite table. Furthermore, the operations in step 1 do not increase the arity of the formulas in I, and the formulas added in step 3 all have arity 2. Thus, the algorithm D runs in $O^*(c^{|V|})$ time where $c = \max\{2, \frac{\alpha(\alpha-1)}{2}\}$. However, if the arity of the formulas in C are at most 2, then the algorithm runs in polynomial time since C'' only contains formulas of arity at most 2—such a formula is either $x = y$ or $x \neq y$.

We continue by proving the main result. Let $I = (V, C)$ denote an arbitrary instance of CSP(Γ). Let F denote the set of functions from V to $\mathbf{U}$ and note that $|F| = |\mathbf{U}|^{|V|}$ is finite since $\mathbf{U}$ is a finite set by the second statement of Lemma 2. If (V, C) has a solution g, then there exists an $f \in F$ such that g respects f since $\mathbf{U}$ is a partitioning of $\mathbb{N}$ by the first statement of Lemma 2. We can thus check the satisfiability of I by applying the algorithm D (which is correct by Lemma 5) to the set of input instances $\{((V, C), f) \mid f \in F\}$. The time complexity is consequently $O^*((|\mathbf{U}| \cdot \frac{\alpha(\alpha-1)}{2})^{|V|})$. □

5 Concluding Remarks

We have studied the fine-grained complexity of infinite-domain equality CSPs, and have proven that this class of problems differ from finite-domain CSPs in almost every way conceivable. Despite the disarray of this complexity landscape, it is possible to outline several concrete future research directions. First, since we know that all finite equality languages can be solved in $O(c^{|V|})$ time and that there exists infinite equality languages not solvable in $O(c^{|V|})$ time for any $c > 1$, is it possible to prove a complete dichotomy separating the equality language CSPs that are solvable in $O(c^{|V|})$ time from those that are not?

More generally, one may ask the following question: which infinite-domain CSPs are solvable in $O(c^{|V|})$ time? This is naturally a question that is too broad so it needs to be narrowed down. An interesting starting point is the class of *temporal CSPs*, i.e., CSPs over first-order reducts of $(\mathbb{Q}; <)$. Temporal languages are well-behaved from a model theoretic viewpoint (they are *ω-categorical*), admit a dichotomy between P and NP-complete, and are always solvable in $O^*(2^{|V| \log |V|})$ time, so one would expect similarities between equality CSPs and temporal CSPs when it comes to fine-grained complexity. Thus, which temporal CSPs are solvable in $O(c^{|V|})$ time? Despite the aforementioned similarities there are still large differences to equality CSPs. For example, there exists a finite first-order reduct Γ of $(\mathbb{Q}; <)$ such that CSP(Γ) is not solvable in $2^{o(|V| \log |V|)}$ time without violating the r-ETH [19].

Last, we have seen that the class of NP-complete equality CSPs does not admit an "easiest problem" unless the ETH is violated, contrary to satisfiability problems [21] and finite-domain CSPs [22]. This discrepancy stems from the constructions in Sect. 3.3 where we proved that one can construct NP-hard equality CSPs with arbitrarily low fine-grained complexity. Furthermore, we gave an algebraic explanation of this difference, namely the non-existence of a weak base for the set of all equality relations. Here, it is important to stress that whether a set of relations admits a weak base or not is a purely algebraic property and it can be formulated entirely without mentioning either CSPs or complexity theory. Interestingly, we first gave a conditional

proof under the ETH (Proposition 1), and later strengthened this to an unconditional proof (Theorem 14). Furthermore, the conditional proof turned out to be simpler and more straightforward than the algebraic proof. To the best of our knowledge, proofs of algebraic properties under the ETH are exceedingly rare, if not non-existent, and this raises the question on whether this is an isolated incidence, or a fragment of a larger phenomena.

Acknowledgements The authors are partially supported by the Swedish Research Council (VR) under Grants 2017-04112, 2019-03690, and 2021-04371.

Funding Open access funding provided by Linköping University.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Barrett, C.W., Sebastiani, R., Seshia, S.A., Tinelli, C.: Satisfiability modulo theories. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185, pp. 825–885. IOS Press, Amsterdam (2009)
2. Barto, L., Pinsker, M.: The algebraic dichotomy conjecture for infinite domain constraint satisfaction problems. In: Proceedings of 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS-2016) (2016)
3. Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.): Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 185. IOS Press (2009)
4. Bodirsky, M.: Complexity of Infinite-Domain Constraint Satisfaction. Cambridge University Press, Cambridge (2021)
5. Bodirsky, M., Bodor, B.: Canonical polymorphisms of Ramsey structures and the unique interpolation property. In: Proceedings of 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS-2021), pp. 1–13 (2021)
6. Bodirsky, M., Grohe, M.: Non-dichotomies in constraint satisfaction complexity. In: Proceedings of 35th International Colloquium on Automata, Languages and Programming (ICALP-2008), pp. 184–196 (2008)
7. Bodirsky, M., Jonsson, P.: A model-theoretic view on qualitative constraint reasoning. J. Artificial Intelligence Res. **58**, 339–385 (2017)
8. Bodirsky, M., Kára, J.: The complexity of equality constraint languages. Theory Comput. Syst. **43**(2), 136–158 (2008)
9. Bodirsky, M., Kára, J.: The complexity of temporal constraint satisfaction problems. J. ACM **57**(2), 9:1-9:41 (2010)
10. Bodirsky, M., Mottet, A.: A dichotomy for first-order reducts of unary structures. Logical Methods in Computer Science **14**(2) (2018)
11. Bojańczyk, M., Klin, B., Lasota, S.: Automata theory in nominal sets. Logical Methods in Computer Science **10**(3) (2014)
12. Bojańczyk, M., Klin, B., Lasota, S., Toruńczyk, S.: Turing machines with atoms. In: Proceedings of 28th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS-2013), pp. 183–192 (2013)
13. Bulatov, A.: A dichotomy theorem for nonuniform CSPs. In: Proceedings of 58th Annual Symposium on Foundations of Computer Science (FOCS-2017) (2017)

14. Burch, J.R., Dill, D.L.: Automatic verification of pipelined microprocessor control. In: Proceedings of 6th International Conference on Computer Aided Verification (CAV-1994), pp. 68–80 (1994)
15. Dylla, F., Lee, J.H., Mossakowski, T., Schneider, T., Delden, A.V., Ven, J.V.D., Wolter, D.: A survey of qualitative spatial and temporal calculi: Algebraic and computational properties. ACM Comput. Surveys **50**(1), 7:1-7:39 (2017)
16. Freivald, R.V.: A completeness criterion for partial functions of logic and many-valued logic algebras. Sov. Phys. Doklady **11**, 288 (1966)
17. Hodges, W.: A Shorter Model Theory. Cambridge University Press, New York (1997)
18. Jonsson, P., Lagerkvist, V.: An initial study of time complexity in infinite-domain constraint satisfaction. Artificial Intelligence **245**, 115–133 (2017)
19. Jonsson, P., Lagerkvist, V.: Why are CSPs based on partition schemes computationally hard? In: 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS-2018), pp. 43:1–43:15 (2018)
20. Jonsson, P., Lagerkvist, V.: Lower bounds and faster algorithms for equality constraints. In: Proceedings of 29th International Joint Conference on Artificial Intelligence (IJCAI-2020), pp. 1784–1790 (2020)
21. Jonsson, P., Lagerkvist, V., Nordh, G., Zanuttini, B.: Strong partial clones and the time complexity of SAT problems. J. Comput. Syst. Sci. **84**, 52–78 (2017)
22. Jonsson, P., Lagerkvist, V., Roy, B.: Fine-grained time complexity of constraint satisfaction problems. ACM Trans. Comput. Theory **13**(1), 2:1-2:32 (2021)
23. Jonsson, P., Lagerkvist, V., Schmidt, J., Uppman, H.: The exponential-time hypothesis and the relative complexity of optimization and logical reasoning problems. Theor. Comput. Sci. **892**, 1–24 (2021)
24. Klin, B., Lasota, S., Ochremiak, J., Toruńczyk, S.: Turing machines with atoms, constraint satisfaction problems, and descriptive complexity. In: Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (CSL-LICS-2014), pp. 58:1–58:10 (2014)
25. Krokhin, A., Jeavons, P., Jonsson, P.: Reasoning about temporal relations: The tractable subalgebras of Allen's interval algebra. J. ACM **50**(5), 591–640 (2003)
26. Lagerkvist, V.: Precise upper and lower bounds for the monotone constraint satisfaction problem. In: Proceedings of the Mathematical Foundations of Computer Science (MFCS-2015), pp. 357–368 (2015)
27. Lagerkvist, V., Wahlström, M.: Sparsification of SAT and CSP problems via tractable extensions. ACM Trans. Comput. Theory **12**(2), 13:1-13:29 (2020)
28. Lau, D.: Function Algebras on Finite Sets: Basic Course on Many-Valued Logic and Clone Theory. Springer, New York (2006)
29. Lokshtanov, D., Marx, D., Saurabh, S.: Slightly superexponential parameterized problems. SIAM J. Comput. **47**(3), 675–702 (2018)
30. Niebert, P., Mahfoudh, M., Asarin, E., Bozga, M., Maler, O., Jain, N.: Verification of timed automata via satisfiability checking. In: Proceedings of 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT-2002), pp. 225–244 (2002)
31. Renz, J., Nebel, B.: On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. Artificial Intelligence **108**(1–2), 69–123 (1999)
32. Rodeh, Y., Strichman, O.: Building small equality graphs for deciding equality logic with uninterpreted functions. Inf. Comput. **204**(1), 26–59 (2006)
33. Romov, B.: The algebras of partial functions and their invariants. Cybernetics **17**(2), 157–167 (1981)
34. Romov, B.A.: Extendable local partial clones. Discrete Math. **308**(17), 3744–3760 (2008)
35. Romov, B.A.: Endpoints of associated intervals for local clones on an infinite set. Algebra Universalis **79**(4), 82 (2018)
36. Schnoor, H., Schnoor, I.: Partial polymorphisms and constraint satisfaction problems. In: Creignou, N., Kolaitis, P.G., Vollmer, H. (eds.) Complexity of Constraints. Lecture Notes in Computer Science, vol. 5250, pp. 229–254. Springer, Berlin (2008)
37. Schutt, A., Stuckey, P.J.: Incremental satisfiability and implication for UTVPI constraints. INFORMS J. Comput. **22**(4), 514–527 (2010)
38. Seshia, S.A., Subramani, K., Bryant, R.E.: On solving Boolean combinations of UTVPI constraints. J. Satisfiability Boolean Model. Comput. **3**(1–2), 67–90 (2007)
39. Traxler, P.: The time complexity of constraint satisfaction. In: Proceedings of 3rd International Workshop on Parameterized and Exact Computation (IWPEC-2008), pp. 190–201 (2008)

40. Volk, M., Junges, S., Katoen, J.: Fast dynamic fault tree analysis by model checking techniques. IEEE Trans. Industrial Inform. **14**(1), 370–379 (2018)
41. Zhuk, D.: A proof of the CSP dichotomy conjecture. J. ACM **67**(5), 30:1-30:78 (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.