# Why are CSPs Based on Partition Schemes Computationally Hard?

**Peter Jonsson**
Department of Computer and Information Science, Linköping University, Linköping, Sweden
peter.jonsson@liu.se

**Victor Lagerkvist**
Department of Computer and Information Science, Linköping University, Linköping, Sweden
victor.lagerkvist@liu.se

## Abstract

Many computational problems arising in, for instance, artificial intelligence can be realized as infinite-domain constraint satisfaction problems (CSPs) based on *partition schemes*: a set of pairwise disjoint binary relations (containing the equality relation) whose union spans the underlying domain and which is closed under converse. We first consider partition schemes that contain a strict partial order and where the constraint language contains all unions of the basic relations; such CSPs are frequently occurring in e.g. temporal and spatial reasoning. We identify three properties of such orders which, when combined, are sufficient to establish NP-hardness of the CSP. This result explains, in a uniform way, many existing hardness results from the literature. More importantly, this result enables us to prove that CSPs of this kind are not solvable in subexponential time unless the exponential-time hypothesis (ETH) fails. We continue by studying constraint languages based on partition schemes but where relations are built using disjunctions instead of unions; such CSPs appear naturally when analysing first-order definable constraint languages. We prove that such CSPs are NP-hard even in very restricted settings and that they are not solvable in subexponential time under the randomised ETH. In certain cases, we can additionally show that they cannot be solved in $O(c^n)$ time for any $c \geq 0$.

**2012 ACM Subject Classification** Theory of computation → Computational complexity and cryptography

**Keywords and phrases** Constraint satisfaction problems, infinite domains, partition schemes, lower bounds

**Digital Object Identifier** 10.4230/LIPIcs.MFCS.2018.43

**Acknowledgements** We thank the anonymous reviewers for several helpful comments. The authors are partially supported by the Swedish Research Council (VR) under grant 2017-04112. In addition, the second author has received funding from the DFG-funded project "Homogene Strukturen, Bedingungserfüllungsprobleme, und topologische Klone" (Project number 622397), and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 681988, CSP-Infinity).

## 1 Introduction

The *constraint satisfaction problem* over a constraint language $\Gamma$ (CSP($\Gamma$)) is the decision problem of verifying whether a set of constraints based on the relations in $\Gamma$ admits a satisfying assignment. For finite domains the complexity of CSP($\Gamma$) is well understood due to the recent dichotomy theorem separating tractable from NP-complete problems [6, 30], but for infinite domains the situation differs markedly. This class of problems includes both undecidable problems and NP-intermediate problems, and it is therefore common to impose additional assumptions on the allowed constraints. The predominant method has been to fix a constraint language $\Gamma$, usually satisfying certain model-theoretic properties, and analyse the complexity of CSPs over first-order reducts of $\Gamma$. Traditionally, this has also been the case for CSPs arising from artificial intelligence, e.g. temporal and spatial reasoning problems, albeit usually with weaker closure conditions.

Motivated by problems of this form, we study the complexity of infinite-domain CSPs over *partition schemes*. A partition scheme [20] is a set of pairwise disjoint binary relations $\mathcal{B}$ over a domain $D$ such that $\bigcup_{R \in \mathcal{B}} R = D^2$ and which for every relation contains its converse. Partition schemes are the de facto standard for CSPs in the artificial intelligence community [8], due to their capability of modelling many different kinds of reasoning problems. Given a partition scheme, the predominant way of forming new relations is to allow unions of the relations in $\mathcal{B}$, and we let $\mathcal{B}^{\vee=}$ denote this set. We will also study languages where each relation can be defined as a disjunction of constraints from $\mathcal{B}$ of arity at most $k \geq 1$, and let $\mathcal{B}^{\vee k}$ denote this set. Note that $\mathcal{B}^{\vee=} \subseteq \mathcal{B}^{\vee k}$ for sufficiently large $k$ but that $\mathcal{B}^{\vee k} \subseteq \mathcal{B}^{\vee=}$ does not necessarily hold for any $k > 1$. Languages of the form $\mathcal{B}^{\vee k}$ occur naturally in theoretical CSP research since such classification projects typically aim to understand the complexity of all first-order reducts of a base set $\mathcal{B}$ of relations.

Famous AI examples of formalisms based on partition schemes include *Allen's interval algebra*, the *region connection calculus*, and the *rectangle algebra*. For more examples, see e.g. the survey by Dylla et al. [9]. CSP($\mathcal{B}^{\vee=}$) problems have been proven to be NP-hard for many choices of $\mathcal{B}$. The proofs have utilised various reductions from various problems, but there has not been a clear explanation why the majority of them are NP-hard. We will try to obtain such an explanation in the sequel. Our first step (in Section 3) is to note that the majority of practically relevant partition schemes contain strict partial orders satisfying certain properties, which we in this paper refer to as *infinite height*, *in-forks*, and *out-forks*. In Section 4 we prove that these properties are sufficient to guarantee that CSP($\mathcal{B}^{\vee=}$) is NP-hard. It might be interesting to observe that we do not need any strong model-theoretic properties, e.g. $\omega$-categoricity, which is otherwise common for infinite-domain CSPs. This result is also interesting to compare to the procedure by Renz and Li [25] which takes a partition scheme as input and tries to prove NP-hardness. One important distinction is that our result provides a concrete source of NP-hardness while the algorithm in Renz and Li gives no such insight. Moreover, this procedure is not complete, and is due to computational constraints not applicable to e.g. the rectangle algebra, while it is a straightforward task to prove that this algebra falls within the scope of our result. Hence, our study offers a more theoretical explanation of why so many naturally occurring CSPs over partition schemes are computationally hard.

Having identified a natural class of NP-hard CSPs based on partition schemes, we turn, in Section 4.2 and Section 5, to the problem of showing lower bounds for problems of this form. Traditionally, it is fair to say that such investigations have largely been neglected by both the artificial intelligence community and the CSP community. There are a few reasons for this. First, significant efforts have been made to solve hard reasoning problems with efficient heuristics [24], which are typically difficult to analyse rigorously even if they work well for certain real-world instances. Second, existing lower bounds are typically based on size-preserving reductions from SAT-like problems where one needs the ability to express disjunctive clauses, which is difficult to express with partition schemes. In fact, to the best of our knowledge, the only concrete lower bounds for a CSP over a partition scheme is the bound by Jonsson and Lagerkvist [16] which relates the complexity of Allen's interval algebra to the complexity of the CHROMATIC NUMBER problem. We show that a size-preserving reduction from a SAT-like problem, perhaps contrary to intuition, is possible for certain CSPs over partition schemes, using ideas from Opatrny [23]. More precisely we prove that CSP($\mathcal{B}^{\vee =}$) cannot be solved in subexponential time unless the *exponential-time hypothesis* is false. One way of interpreting this result is that CSP($\mathcal{B}^{\vee =}$) is far from being polynomial-time solvable: there is a constant $c > 1$ such that the problem cannot be solved in $O(c^n)$ time. An immediate consquence of lower bounds of this form is that we can immediately rule out certain kinds of algorithms for CSP($\mathcal{B}^{\vee =}$), e.g. algorithms based on graph-decomposition and $k$-consistency, which typically run in subexponential or polynomial time. It is of course tempting to strengthen our lower bound even further since the current best known algorithm for CSP($\mathcal{B}^{\vee =}$) for an arbitrary partition scheme $\mathcal{B}$ runs in $2^{O(n^2)}$ time, if CSP($\mathcal{B}$) is polynomial-time solvable [16, 27]. While we do not succeed in doing this, we can provide stronger lower bounds for CSP($\mathcal{B}^{\vee k}$): we prove that CSP($\{\prec\}^{\vee 4}$), where $\prec$ is a strict partial order of infinite height, is not solvable in $O(c^n)$ time for any $c \geq 0$ assuming the complexity theoretical assumption known as the *randomised exponential-time hypothesis* (r-ETH). We also show that CSP($\mathcal{B}^{\vee 2}$) cannot be solved in subexponential time if we assume the r-ETH and that a non-empty relation $R \subseteq \{(x, y, z) \in D^3 \mid x \neq y, x \neq z, y \neq z\}$ can be defined in $\mathcal{B}^{\vee 2}$. Note that we do not require $\mathcal{B}$ to contain any partial orders in this case. We conclude the paper with some discussion in Section 6, where we point out some future research directions concerning both lower and upper bounds.

## 2 Preliminaries

In this section we introduce the necessary prerequisites concerning constraint satisfaction problem, disjunctive relations, and partition schemes. We begin by defining the CSP problem when it is parameterized by a set of relations.

▶ **Definition 1.** Let $\Gamma$ be a set of finitary relations over some set $D$ of values. The *constraint satisfaction problem* over $\Gamma$ (CSP($\Gamma$)) is defined as follows:

*Instance:* A set $V$ of variables and a set $C$ of *constraints* of the form $R(v_1, \ldots, v_k)$, where $k$ is the arity of $R$, $v_1, \ldots, v_k \in V$ and $R \in \Gamma$.

*Question:* Is there a function $f : V \to D$ such that $(f(v_1), \ldots, f(v_k)) \in R$ for every $R(v_1, \ldots, v_k) \in C$?

The set $\Gamma$ is called a *constraint language*. Given an instance $I$ of CSP($\Gamma$) we write $||I||$ for the number of bits required to represent $I$. We will occasionally encounter *bounded-degree* CSP instances. Let $(V, C)$ denote an instance of CSP($\Gamma$). If a variable $x$ occurs in $B$ constraints in $C$, then we say that the degree of $x$ is $B$. We let CSP($\Gamma$)-$B$ denote the CSP($\Gamma$) problem where each variable in the input is restricted to have degree at most $B$. Note that if $(V, C)$ is a CSP($\Gamma$)-$B$ instance, then $|C| \leq B \cdot |V|$, implying that the number of constraints is linearly bounded with respect to the number of variables.

We continue by describing how to use disjunctions for combining relations.

▶ **Definition 2.** Let $D$ be a set of values and let $\mathcal{B} = \{B_1, \ldots, B_m\}$ denote a finite set of relations over $D$, i.e. $B_i \subseteq D^j$ for some $j \geq 1$.

1. A *disjunctive formula* over $\mathcal{B}$ is of the form $B_1(\mathbf{x}_1) \vee \cdots \vee B_t(\mathbf{x}_t)$ where $\mathbf{x}_1, \ldots, \mathbf{x}_t$ are sequences of variables from $\{x_1, \ldots, x_p\}$ such that the length of $\mathbf{x}_j$ equals the arity of $B_j$, and $B_1, \ldots, B_t \in \mathcal{B}$. The arity of a disjunctive formula $B_1(\mathbf{x}_1) \vee \cdots \vee B_t(\mathbf{x}_t)$ is $t$.
2. $\mathcal{B}^{\vee k} = \{R \mid R$ is definable by a disjunctive formula over $\mathcal{B}$ of arity $l \leq k\}$.

For simplicity we represent relations in $\mathcal{B}^{\vee k}$ by their defining disjunctive formulas. Two syntactially distinct disjunctive formulas may now denote the same relation, implying that

**Table 1** The thirteen basic relations in Allen's interval algebra. The endpoint relations $x^s < x^e$ and $y^s < y^e$ that are valid for all relations have been omitted.

| Basic relation | | Example | Endpoints |
|---|---|---|---|
| $x$ precedes $y$ | $\mathsf{p}$ | xxx | $I^+ < J^-$ |
| $y$ preceded by $x$ | $\mathsf{p}^{-1}$ | yyy | |
| $x$ meets $y$ | $\mathsf{m}$ | xxxx | $I^+ = J^-$ |
| $y$ met-by $x$ | $\mathsf{m}^{-1}$ | yyyy | |
| $x$ overlaps $y$ | $\mathsf{o}$ | xxxx | $I^- < J^- < I^+$, |
| $y$ overl.-by $x$ | $\mathsf{o}^{-1}$ | yyyy | $I^+ < J^+$ |
| $x$ during $y$ | $\mathsf{d}$ | xxx | $I^- > J^-$, |
| $y$ includes $x$ | $\mathsf{d}^{-1}$ | yyyyyyy | $I^+ < J^+$ |
| $x$ starts $y$ | $\mathsf{s}$ | xxx | $I^- = J^-$, |
| $y$ started by $x$ | $\mathsf{s}^{-1}$ | yyyyyyy | $I^+ < J^+$ |
| $x$ finishes $y$ | $\mathsf{f}$ | xxx | $I^+ = J^+$, |
| $y$ finished by $x$ | $\mathsf{f}^{-}1$ | yyyyyyy | $I^- > J^-$ |
| $x$ equals $y$ | $\equiv$ | xxxx | $I^- = J^-$, |
| | | yyyy | $I^+ = J^+$ |

this representation is not unique. To avoid tedious technicalities we ignore this issue and whenever convenient view constraint languages as multisets.

We are now ready to introduce *partition schemes* [20]. Let $\mathcal{B} = \{B_1, \ldots, B_m\}$ be a set of binary relations over a domain $D$. We say that $\mathcal{B}$ is *jointly exhaustive* if $\bigcup \mathcal{B} = D^2$ and that $\mathcal{B}$ is *pairwise disjoint* if $B_i \cap B_j = \emptyset$ whenever $i \neq j$. We say that $\mathcal{B}$ is a *partition scheme* if (1) $\mathcal{B}$ is jointy exhaustive and pairwise disjoint, (2) $\mathsf{eq}_D = \{(x,x) \mid x \in D\} \in \mathcal{B}$, and (3) for every $B_i \in \mathcal{B}$, the converse relation $B_i^{\smile}$ (i.e. $B_i^{\smile} = \{(y,x) \mid (x,y) \in B_i\}$) is in $\mathcal{B}$. We define $\mathcal{B}^{\vee=}$ to be the set of all unions of relations from $\mathcal{B}$. Equivalently, each relation in $\mathcal{B}^{\vee=}$ can be viewed as a disjunction $B_1(x,y) \vee B_2(x,y) \vee \cdots \vee B_k(x,y)$ for some $\{B_1, \ldots, B_k\} \subseteq \mathcal{B}$. We sometimes abuse notation and write $(B_1, \ldots, B_k)$ to denote the relation $B_1 \cup \cdots \cup B_k$. The set $\mathcal{B}^{\vee=}$ and the problem CSP($\Gamma$) where $\Gamma \subseteq \mathcal{B}^{\vee=}$ are typical objects that are studied in artificial intelligence literature. For example, it has been common to use a *relation algebra* $\mathcal{A}$ as a starting point and then define a *network satisfaction problem* over $\mathcal{A}$, which in our notation is nothing else than the CSP over a set of binary relations. Note that if CSP($\mathcal{B}$) is polynomial-time solvable, then both CSP($\mathcal{B}^{\vee k}$) and CSP($\mathcal{B}^{\vee=}$) are members of NP.

▶ **Example 3.** *Allen's interval algebra* [2] is a well-known formalism for temporal reasoning where one considers relations between intervals of the form $I = [I^+, I^-]$, where $I^+, I^- \in \mathbb{R}$ is the start and end point, respectively. In Allen's algebra one can for instance describe that one interval begins before another interval, and one express such relations in terms of a partition scheme consisting of 13 basic relations (see Table 1), and then form more complicated relations by taking the union of the basic relations. If we let $\mathcal{A}$ denote the set of 13 basic relations in Allen's algebra, then CSP($\mathcal{A}^{\vee=}$) is an alternative formulation of the network consistency problem over Allen's algebra.

An extension of the interval algebra is the so-called *rectangle algebra* [13, 22]. Here, one considers relations between rectangles in the plane by extending the basic relations in the interval algebra to the projections of a rectangle onto the $x$- and $y$-axis, respectively. In other words, given $r, s \in \mathcal{A}$ and two rectangles represented by the intervals $I_x, I_y$, $J_x, J_y$ we may define the relation $r \oplus s$ in the rectangle algebra holding if $I_x(r)J_x$ and $I_y(s)J_y$.

## 3 Partial Orders

CSPs based on partition schemes are very often used for *qualitative reasoning*. We acknowledge that it is not obvious how to define "qualitative reasoning" rigorously, but the concept seems to have an informal meaning that is generally accepted. Renz and Nebel [27, p. 161] write

> Qualitative reasoning is an approach for dealing with commonsense knowledge without using numerical computation. Instead, one tries to represent knowledge using a limited vocabulary such as qualitative relationships between entities or qualitative categories of numerical values, ...

Abstraction is the defining feature of qualitative reasoning: qualitative reasoning is about disregarding unnecessary and uninteresting details. With this in mind, it is clear that an important kind of qualitative relationships between objects are "part-of" relations. One may argue that such relations are strict partial orders that satisfy certain additional properties. We will now define three properties of strict partial orders, *infinite height*, *in-fork*, and *out-fork*, that appear to be relevant in the pursuit of classifying the complexity of CSP$(\mathcal{B}^{\vee=})$. A typical example of such a relation is the NTPP relation in RCC-8 – this can be viewed as an archetypical example of a "part-of" relation. Many other relations that are not "part-of" relations satisfy these properties, too: one example is the precedes relation $\mathsf{p}$ in Allen's algebra. In fact, relations of this kind appear very frequently in CSPs for qualitative reasoning.

Let $\prec \subseteq D^2$ denote a binary relation let $\succ$ denote its converse $\prec^{\smile}$. We say that $\prec$ is a *strict partial order* if there is no $d \in D$ such that $d \prec d$ (irreflexivity) 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 will now define three additional properties of strict partial orders. First define $\sqcap = D^2 \setminus \bigcup\{\prec, \succ, \mathsf{eq}_D\}$, and note that $x \sqcap y$ holds if and only if $x$ and $y$ are incomparable with respect to $\prec$.

▶ **Definition 4.** Let $\prec \subseteq D^2$ be a strict partial order over a domain $D$. We define the following properties.

**C1.** (infinite height) for every $k \geq 1$, there exists a sequence of elements $d_1, d_2, \ldots, d_k$ in $D$ such that $d_1 \prec d_2 \prec \cdots \prec d_k$,

**C2.** (in-fork) if $a, b, c \in D$ and $a \prec b \prec c$, then there exists $d_1 \in D$ such that $d_1 \sqcap a$, $d_1 \sqcap b$, and $d_1 \prec c$, and

**C3.** (out-fork) if $a, b, c \in D$ and $a \prec b \prec c$, then there exists $d_2 \in D$ such that $d_2 \succ a$, $d_2 \sqcap b$, and $d_2 \sqcap c$.

Partial orders satisfying these three properties are abundant in the artificial intelligence literature, but has to the best of our knowledge not been explicitly formalized before. The conditions in-fork and out-fork are illustrated in Figure 1. Given a binary relation $\prec$ it is typically easy to check if it is a strict partial order of infinite height, but checking if it also satisfies in-fork and out-fork may need additional work. Consider Allen's algebra and the relation $\mathsf{p}$, i.e. the relation stating that one interval appears strictly before another interval. In this case, $\sqcap$ is the relation that holds if and only if two distinct intervals have at least one point in common. Pick three intervals $I_j = [I_j^-, I_j^+]$, $1 \leq j \leq 3$, such that $I_1(\mathsf{p})I_2(\mathsf{p})I_3$. For in-fork, we choose $I_4 = [I_1^-, I_2^+]$ so that $I_4 \sqcap I_1$, $I_4 \sqcap I_2$, and $I_4 \prec I_3$. For out-fork, one may choose $I_5 = [I_2^-, I_3^+]$.

Let us consider another example where the domain contains the closed disks in $\mathbb{R}^2$ and the relation $\prec$ is the strict subset relation. Pick three disks $d_1, d_2, d_3 \in D$ such that $d_1 \prec d_2 \prec d_3$. How to choose suitable disks for verifying in-fork and out-fork is illustrated in Figure 2. This

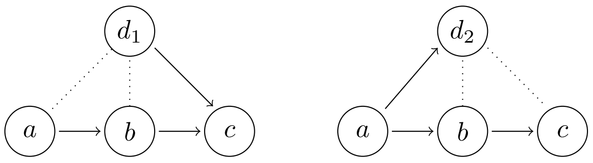

**Figure 1** Illustration of in-fork (left) and out-fork (right). Arrows denote the $\prec$ relation and dotted lines the $\sqcap$ relation.

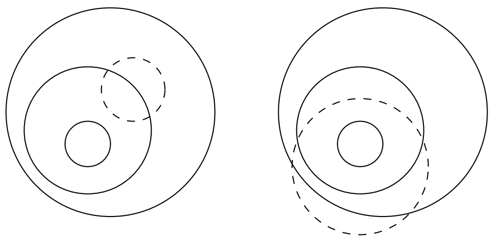

**Figure 2** The dashed circles show possible choices of disks for in-fork (left) and out-fork (right).

example can easily be adapted to relations such as (PP) in RCC-5 and (NTPP) in RCC-8, and the relation $\mathsf{d} \oplus \mathsf{d}$ in the rectangle algebra. Many additional examples can be found in the survey by Dylla et al. [9], e.g. Goyal & Egenhofer's Cardinal Direction Calculus and Ragni & Scivos' Dependency Calculus. Last, let us remark that there are examples of strict partial orders that do not have in- or out-forks. Well-known examples are the less-than relation $<$ in the (1-dimensional) point algebra and in the branching time algebra. Interestingly, CSP($\mathcal{B}^{\vee=}$) is polynomial-time solvable in these two cases and we will come back to this observation at the end of Section 4.1.

## 4 Lower Bounds for CSP($\mathcal{B}^{\vee=}$)

We will now study the computational complexity of CSP($\mathcal{B}^{\vee=}$) when $\mathcal{B}$ contains a strict partial order of infinite height with in- and out-forks. In Section 4.1, we prove that CSP($\mathcal{B}^{\vee=}$) is NP-hard and we use this result in Section 4.2 for proving that CSP($\mathcal{B}^{\vee=}$) cannot be solved in subexponential time (given that the ETH holds).

### 4.1 NP-hardness

NP-hardness of CSP($\mathcal{B}^{\vee=}$) for specific partition schemes $\mathcal{B}$ containing a strict partial order of infinite height with in- and out-forks has been proven many times in the literature. Examples where this connection is quite pronounced can be found in, for instance, Grigni et al. [12], Renz and Nebel [26], Moratz et al. [21], and Krokhin et al. [18] The basis for our reduction is the NP-complete problem BETWEENNESS.

*Instance:* A finite set $A$ and a collection $T$ of ordered triples $(a, b, c)$ of distinct elements from $A$.
*Question:* Is there a total ordering $<$ on $A$ such that for each $(a, b, c) \in T$, we have either $a < b < c$ or $c < b < a$?

Our hardness result requires two steps that are presented in Lemma 5 and Theorem 6.

▶ **Lemma 5.** *Let $\mathcal{B}$ be a set of binary relations over a domain $D$ containing a strict partial order $\prec$ of infinite height. Let $G(a, b, c, x_1, \ldots, x_m)$ be an instance $(V, C)$ of CSP($\mathcal{B}^{\vee=}$), where $V = \{a, b, c, x_1, \ldots, x_k\}$, and having the following properties.*

**G1.** *For arbitrary elements $d_a, d_b, d_c \in D$ such that $d_a \prec d_b$ and $d_b \prec d_c$, there exist elements $d_1, \ldots, d_m \in D$ such that the function $s : V \to \{d_a, d_b, d_c, d_1, \ldots, d_m\}$ defined by $s(a) = d_a$, $s(b) = d_b$, $s(c) = d_c$, and $s(x_i) = d_i$, $1 \leq i \leq m$, is a solution to the instance $(V, C \cup \{a \prec b, b \prec c\})$.*

**G2.** *For arbitrary elements $d_a, d_b, d_c \in D$ such that $d_c \prec d_b$ and $d_b \prec d_a$, there exist elements $d_1, \ldots, d_m \in D$ such that the function $s : V \to \{d_a, d_b, d_c, d_1, \ldots, d_m\}$ defined by $s(a) = d_c$, $s(b) = d_b$, $s(c) = d_a$, and $s(x_i) = d_i$, $1 \leq i \leq m$, is a solution to the instance $(V, C \cup \{c \prec b, b \prec a\})$.*

**G3.** *$(V, C \cup \{b \prec a, b \prec c, a(\prec, \succ)c\})$ is not satisfiable.*

**G4.** *$(V, C \cup \{a \prec b, c \prec b, a(\prec, \succ)c\})$ is not satisfiable.*

*Let $\Gamma$ be the set of relations that appear in $G$. Then, CSP$(\Gamma \cup \{\prec, \succ\})$ is NP-hard.*

**Proof.** Let $\Gamma' = \Gamma \cup \{(\prec, \succ)\}$. We present a polynomial-time reduction from Betweenness to CSP$(\Gamma')$. Arbitrarily choose an instance $(A, T)$ of Betweenness and construct an instance $I$ of CSP$(\Gamma')$ as follows:

1. for each pair of distinct elements $a, b \in A$, add the constraint $a(\prec, \succ)b$ to $I$, and
2. for each triple $(a, b, c) \in T$, introduce $m$ fresh variables $x_1, \ldots, x_m$ and add $G(a, b, c, x_1, \ldots, x_m)$ to $I$.

We refer to the variables in $I$ that correspond to the set $A$ as basic variables and the other variables as auxiliary variables. We first assume that $s$ is a solution to $I$. Let $S = \{s(a) \mid a \in A\}$. The constraints introduced in step (1) implies that the $|S| = |A|$ and the relation $\prec$ induces a total order on the set $S$. Assume to the contrary that there, for example, exists a triple $(a, b, c) \in T$ such that $s(b) \prec s(a) \prec s(c)$. Then, the instance $(V, C \cup \{b(\prec)a, b(\prec)c, a(\prec, \succ)c\})$ introduced in step (2) is satisfiable and this contradicts our assumptions. Analogously, we can rule out all orderings except $s(a) \prec s(b) \prec s(c)$ and $s(c) \prec s(b) \prec s(a)$. We conclude that there is a solution to the instance $(A, T)$: for all $a, b \in A$, set $a < b$ if and only if $s(a) \prec s(b)$.

Assume now that there exists a solution $<$ to $(A, T)$. We show how to construct a solution to the instance $I$. We rename the members of $A$ such that $A = \{a_1, \ldots, a_n\}$ and $a_1 < a_2 < \cdots < a_n$. Arbitrarily choose elements $d_1, \ldots, d_n \in D$ such that $d_1 \prec d_2 \prec \cdots \prec d_n$. Such elements exist since $\prec$ is a strict partial order of infinite height. Let $s(a_i) = d_i$, $1 \leq i \leq n$, and note that $s$ satisfies all constraints introduced in step 1.

Arbitrarily choose a triple $(a, b, c) \in T$ and consider the gadget $G(a, b, c, x_1, \ldots, x_k)$ that is introduced in step 2. If $a < b < c$, then $s(a) \prec s(b) \prec s(c)$ and $x_1, \ldots, x_k$ can be assigned values that satisfy the gadget by condition (1). If $c < b < a$, then $s(c) \prec s(b) \prec s(a)$ and $x_1, \ldots, x_k$ can be assigned values that satisfy the gadget by condition (2). Thus, for every triple $(a, b, c) \in T$, we can find values for the auxiliary variables that satisfy all $G$-gadgets. Note that two distinct $G$-gadgets do not have any auxiliary variables in common. We conclude that $I$ is satisfiable. ◀

▶ **Theorem 6.** *Let $\mathcal{B}$ be a partition scheme with domain $D$ containing a strict partial order $\prec$ of infinite height with in- and out-forks. Then CSP$(\mathcal{B}^{\vee=})$ is NP-hard.*

**Proof.** First observe that the relation $\sqcap = D^2 \setminus \bigcup\{\prec, \succ, \mathsf{eq}_D\}$ is a member of $\mathcal{B}^{\vee=}$ since $\mathcal{B}$ is a partition scheme. We will now define the following gadget: $G(a, b, c, x_1, x_2) = (\{a, b, c, x_1, x_2\}, \{x_1 \sqcap a, x_1 \sqcap b, x_1(\prec, \succ)c, x_2(\prec, \succ)a, x_2 \sqcap b, x_2 \sqcap c\})$. We demonstrate that $G$ satisfies the preconditions of Lemma 5. We first consider the following condition:

**C4.** if $a \prec b \prec c$, then there does not exist $d_3 \in D$ such that $d_3 \sqcap a$, $d_3(\prec, \succ)b$, and $d_3 \sqcap c$.

We verify that C4 always holds under the assumptions stated in the theorem. Assume to the contrary that $a \prec b \prec c$ and $d_3 \in D$ satisfies $d_3 \sqcap a$, $d_3(\prec, \succ)b$, and $d_3 \sqcap c$. The relation $\prec$ is a strict partial order so it is transitive. If $d_3 \prec b$, then $d_3 \prec c$ and $d_3 \sqcap c$ cannot hold since the relations $\prec$ and $\sqcap$ are disjoint. Similarly, if $d_3 \succ b$, then $a \prec d_3$ and $d_3 \sqcap a$ cannot hold.

Next, we consider conditions G1 and G2 and show that they are satisfied: we see that proper assignments to variables $x_1$ and $x_2$ exist due to in-fork and out-fork. Assume to the contrary that G3 does not hold, i.e. $\{x_1 \sqcap a, x_1 \sqcap b, x_1(\prec, \succ)c, x_2(\prec, \succ)a, x_2 \sqcap b, x_2 \sqcap c\} \cup \{b(\prec)a, b(\prec)c, a(\prec, \succ)c\}$ is satisfiable. Under these constraints, two orderings of $a, b, c$ are possible: $b \prec a \prec c$ and $b \prec c \prec a$. We consider the case $b \prec a \prec c$; the other case is analogous. Note now that $x_2 \sqcap b$, $x_2(\prec, \succ)a$, and $x_2 \sqcap c$. These constraints do not have a solution due to C4, and we conclude that G3 holds. That G4 holds can be shown analogously. The result then follows from Lemma 5. ◀

Hence, the properties in Definition 4 are sufficient for establishing NP-hardness of CSP($\mathcal{B}^{\vee=}$), and it is thus natural to ask to which extent they are also necessary. Although a complete answer seems difficult to obtain, we may at least observe that if $\prec \in \mathcal{B}$ is a strict partial order of finite height, then CSP($\mathcal{B}^{\vee=}$) is NP-hard, regardless of whether $\prec$ have in- and out-forks or not. This can be seen via a polynomial-time reduction from $k$-Colourability to CSP($\mathcal{B}^{\vee=}$) for some constant $k \geq 1$. Let $(V, E)$ be an arbitrary undirected graph. Introduce variables $c_1, \ldots, c_k$ for each colour, and constrain them as $c_1(\prec)c_2(\prec)\ldots(\prec)c_k$. For each vertex $v \in V$, introduce a variable $w$ and the constraints $w(\prec, \succ, \mathtt{eq}_D)c_i$, $1 \leq i \leq k$, and observe that $\succ, \mathtt{eq}_D \in \mathcal{B}$ since $\mathcal{B}$ is a partition scheme. Note that these constraints imply that $w$ equals exactly one colour variable in any satisfying assignment. Finally, introduce the constraint $w(\prec, \succ)w'$ for each edge $(v, v')$ in $E$. It is easy to verify that the resulting CSP($\mathcal{B}^{\vee=}$) instance has a solution if and only if $(V, E)$ is $k$-colourable. It is also easy to verify that the reduction can be computed in polynomial time since $k$ is a constant that only depends on the choice of $\mathcal{B}$. Since $k$-Colourability is NP-hard whenever $k \geq 3$, NP-hardness of CSP($\mathcal{B}^{\vee=}$) follows.

Similarly, it is natural to ask what happens if $\prec$ is a strict partial order of infinite height which does not have in- and/or out-forks. We have seen that this sometimes leads to tractability, as in the case of e.g. the point algebra and the branching time algebra, but this is not always the case. For a simple counter example, let $D = \{(0, i), (1, i), (2, i) \mid i \in \mathbb{N}\}$ and define $\prec \subseteq D^2$ such that $(a, b) \prec (c, d)$ if and only if $a = c$ and $b < d$. It is easy to verify that $\prec$ is a strict partial order of infinite height and that it does not have in- or out-forks. Let $\mathcal{B} = \{\prec, \succ, \sqcap, \mathtt{eq}_D\}$ where $\sqcap = D^2 \setminus \bigcup\{\prec, \succ, \mathtt{eq}_D\}$, and observe that $\mathcal{B}$ is a partition scheme. We show that CSP($\mathcal{B}^{\vee=}$) is an NP-hard problem via a polynomial-time reduction from 3-Colourability. Let $(V, E)$ be an arbitrary undirected graph. For each vertex $v \in V$, introduce a variable $w$, and for each edge $(w, w') \in E$, introduce the constraint $w \sqcap w'$. Note that $((a, b), (c, d)) \in \sqcap$ if and only if $a \neq c$ and that $a$ and $c$ are restricted to the three-element set $\{0, 1, 2\}$. Given this, it is easy to verify that the resulting CSP($\mathcal{B}^{\vee=}$) instance has a solution if and only if $(V, E)$ is 3-colourable.

## 4.2 ETH-based Lower Bound

Based on the results presented in the previous section, we will now show that CSP($\mathcal{B}^{\vee=}$) cannot be solved in subexponential time if $\mathcal{B}$ contains a strict partial order of infinite height with in- and out-forks, unless the exponential-time hypothesis (ETH) does not hold. If CSP($\Gamma$) is solvable in $O(c^n)$ time by a deterministic algorithm for every $c > 1$ (where $n$ denotes the number of variables) then CSP($\Gamma$) is said to be *subexponential.* The exponential-time hypothesis is the conjecture that 3-SAT is not solvable in subexponential time [15].

The NP-hardness proof of Betweenness by Opatrny [23] is based on a reduction from the Rank-3 hypergraph 2-colourability problem. A *hypergraph* is a pair $H = (V, \mathcal{E})$ such that $V$ is a non-empty finite set and $\mathcal{E}$ is a non-empty finite set of subsets of $V$. The elements of $V$ are called the *nodes* of $H$ and the elements of $\mathcal{E}$ are the *edges* of $H$. The *rank* of $H$ is $\max\{|e| \mid e \in \mathcal{E}\}$, and the Rank-$k$ hypergraph 2-colourability problem is defined as follows.

*Instance:* A rank-$k$ hypergraph $H = (V, \mathcal{E})$.
*Question:* Do there exist sets $V_0, V_1 \subseteq V$ such that $V_0 \cap V_1 = \emptyset$ and $V_0 \cap e \neq \emptyset$, $V_1 \cap e \neq \emptyset$ for every $e \in \mathcal{E}$?

Define relations $R_1 = \{(0,1),(1,0)\}$ and $R_2 = \{0,1\}^3 \setminus \{(0,0,0),(1,1,1)\}$ and note that CSP($\{R_1, R_2\}$) is an obvious reformulation of the Rank-3 hypergraph 2-colourability problem. Lemma 2 in Opatrny [23] immediately implies the following result.

▶ **Lemma 7.** *Let $I = (V, C)$ denote an arbitrary instance of CSP($\{R_1, R_2\}$). It is possible to construct an instance $(A, T)$ of the* Betweenness *problem in polynomial time with the following properties.*

1. *$I$ has a solution if and only if $(A, T)$ has a solution,*
2. *$|A| \leq |V| + 1 + |C|$, and*
3. *$|T| \leq 2|C|$.*

▶ **Theorem 8.** *Assume the ETH holds. If $\mathcal{B}$ is a partition scheme such that $\prec \in \mathcal{B}$ and $\prec$ is a strict partial order of infinite height with in- and out-forks, then CSP($\mathcal{B}^{\vee=}$) is not solvable in subexponential time.*

**Proof.** Results by Jonsson et al. [17] imply that CSP($\{R_1, R_2\}$)-$B$ cannot be solved in subexponential time for some $B \geq 1$. Let $I = (V, C)$ denote an arbitrary instance of CSP($\{R_1, R_2\}$)-$B$. Recall that $|C| \leq B \cdot |V|$ since each variable can occur in at most $B$ constraints. Lemma 7 shows that we can (in polynomial time) construct an instance $(A, T)$ of Betweenness such that

1. $I$ has a solution if and only if $(A, T)$ has a solution,
2. $|A| \leq K \cdot |V|$, and
3. $|T| \leq L \cdot |C| \leq L \cdot B \cdot |V|$.

for some universal constants $K, L$. Lemma 5 combined with the standard gadget shows that we can (in polynomial time) construct an instance $I' = (V', C')$ of CSP($\mathcal{B}^{\vee=}$) such that

1. $I'$ has a solution if and only if $(A, T)$ has a solution and
2. $|V'| \leq |A| + 2|T|$.

Note that $|V'| \leq |A| + 2|T| \leq K|V| + 2L|C| \leq K|V| + 2LB|V| = (K + 2LB)|V|$. If CSP($\mathcal{B}^{\vee=}$) is solvable in subexponential time, then CSP($\{R_1, R_2\}$)-$B$ is solvable in subexponential time, too, and this leads to a contradiction. ◀

In summary, we may rule out subexponential time algorithms for CSP($\mathcal{B}^{\vee=}$) for partition schemes $\mathcal{B}$ containing a strict partial order of infinite height with in- and out-forks. However, the best general algorithm for CSP($\mathcal{B}^{\vee=}$) runs in $O(2^{O(n^2)})$ time (if CSP($\mathcal{B}$) is tractable) [16, 27]. Hence, there is a large discrepancy between the upper and lower bound for this problem, suggesting that (at least) one of these bounds can be strengthened.

## 5 Lower Bounds for CSP($\mathcal{B}^{\vee k}$)

In this section, we will make use of the randomised version of ETH, and need a few additional definitions. First, let $\Gamma_{d,k}$, $d,k \geq 1$, denote the set of relations with arity at most $k$ over the domain $\{1,\ldots,d\}$. A CSP algorithm $A$ is said to be a $2^{c \cdot n}$-randomised algorithm if its running time is bounded by $2^{c \cdot n} \cdot poly(||I||)$ (where $n$ is the number of variables) and its error probability is at most $1/3$. Let $c_{d,k} = \inf\{c \mid \exists\ 2^{c \cdot n}\text{-randomised algorithm for CSP}(\Gamma_{d,k})\}$. The variant of the ETH that we will use in the forthcoming lower bound states that $c_{2,3} > 0$, i.e., 3-SAT cannot be solved in subexponential time even if we are allowed to use randomised algorithms. We let r-ETH denote this hypothesis. Traxler [29] has shown the following result.

▶ **Theorem 9.** *If r-ETH holds, then there exists a universal constant $\alpha > 0$ such that for all $d \geq 3$, $\alpha \cdot \log(d) \leq c_{d,2}$.*

We begin by proving a result for $\mathcal{B}^{\vee 2}$ that is analogous to Theorems 6 and 8. Let $\mathcal{B}$ be a partition scheme over a domain $D$. Assume that $\mathcal{B}$ admits a gadget that forces three variables to be assigned distinct values, i.e., it is possible to define a non-empty ternary relation $R$ such that $R \subseteq \{(x,y,z) \in D^3 \mid x \neq y, x \neq z, y \neq z\}$. This gadget can be defined for all examples considered in this paper, and in particular it can be defined by any strict partial order relating at least three elements (via $R(x,y,z) \equiv x \prec y \prec z$). Let $S(x,y,z) \equiv \mathsf{eq}_D(x,y) \vee \mathsf{eq}_D(x,z)$. Note that the relation $S$ is a member of $\mathcal{B}^{\vee 2}$ since $\mathcal{B}$ is a partition scheme.

▶ **Theorem 10.** *Assume that $\mathcal{B}$ is a partition scheme admitting a gadget as described above. Then, CSP($\mathcal{B}^{\vee 2}$) is NP-hard, and if the r-ETH holds, there is no $2^{\frac{c_{3,2}}{5} \cdot n}$-randomised algorithm for CSP($\mathcal{B}^{\vee 2}$).*

**Proof.** We present a polynomial-time reduction from CSP($\Gamma_{3,2}$) to CSP($\mathcal{B}^{\vee 2}$). If the given CSP($\Gamma_{3,2}$) instance contains $n$ variables, then the CSP($\mathcal{B}^{\vee 2}$) instance will contain at most $5n + K$ variables where $K$ is a constant. By Theorem 9, CSP($\Gamma_{3,2}$) cannot be solved in $2^{c_{3,2}n}$ time, so CSP($\mathcal{B}^{\vee 2}$) cannot be solved in $2^{\frac{c_{3,2}}{5}n}$ time.

Let $(V,C)$ be an arbitrary instance of CSP($\Gamma_{3,2}$) where $V = \{x_1,\ldots,x_n\}$. To construct our CSP($\mathcal{B}^{\vee 2}$) instance, we perform the following steps.

1. Introduce three variables $d_1, d_2, d_3$ and the gadget that makes them distinct. These variables will be used to denote the three domain elements.
2. For each variable $x_i$, $1 \leq i \leq n$, we introduce the variable $x'_i$.
3. For each variable $x_i$, $1 \leq i \leq n$, introduce the variable $y_i$ together with the constraints $S(y_i, d_2, d_3)$ and $S(x_i, d_1, y_i)$. These constraints imply that $x_i$ is equal to $d_1$, $d_2$, or $d_3$.
4. For each variable $x_i$, $1 \leq i \leq n$, introduce variables $x_i^{\neq 1}, x_i^{\neq 2}, x_i^{\neq 3}$ together with the constraints $S(x_i^{\neq 1}, d_2, d_3)$, $S(x_i^{\neq 2}, d_1, d_3)$, and $S(x_i^{\neq 3}, d_1, d_2)$. These variables are used for "simulating" inequalities in step 5.
5. For each constraint $R(x_i, x_j) \in C$ and each tuple $(a,b) \in \{1,2,3\}^2$ that is not in $R$, introduce the constraint $\mathsf{eq}_D(x_i, x_i^{\neq a}) \vee \mathsf{eq}_D(x_j, x_j^{\neq b})$.

The resulting CSP($\mathcal{B}^{\vee 2}$) instance $(V', C')$ can obviously be constructed in polynomial time. It contains $5n$ variables plus the constant number of variables needed for the gadget. We claim that $(V', C')$ has a solution if and only if $(V,C)$ has a solution. Assume that $f : V \to \{1,2,3\}$ is a solution to $(V,C)$. Let $c_1, c_2, c_3 \in D$ be three distinct values that are permitted by the gadget. Let $U$ denote the set of other values used by the gadget. Define $f' : V' \to U \cup \{c_1, c_2, c_3\}$ as follows.

- $f'$ assigns suitable values from $U$ to the gadget,
- $f'(d_i) = c_i$, $1 \leq i \leq 3$,
- $f'(y_i) = c_2$ if $f(x_i) = 2$ and $f'(y_i) = c_3$ otherwise,
- $f'(x'_i) = c_{f(x_i)}$
- $f'(x_i^{\neq 1}) = c_{f(x_i)}$ if $f(x_i) \neq 1$ and $f'(x_i^{\neq 1}) = c_2$, otherwise,
- $f'(x_i^{\neq 2}) = c_{f(x_i)}$ if $f(x_i) \neq 2$ and $f'(x_i^{\neq 2}) = c_1$, otherwise,
- $f'(x_i^{\neq 3}) = c_{f(x_i)}$ if $f(x_i) \neq 3$ and $f'(x_i^{\neq 3}) = c_2$, otherwise.

The function $f'$ can easily be seen to satisfy the constraints introduced in steps 1, 3 and 4. We consider the constraints introduced in step 5. Pick a constraint $R(x_i, x_j) \in C$ and a tuple $(a, b) \in \{1, 2, 3\}^2$ that is not in $R$. We assume without loss of generality that $a = 1$ and $b = 2$. The corresponding constraint in $C'$ is now $\mathtt{eq}_D(x'_i, x_i^{\neq 1}) \vee \mathtt{eq}_D(x'_j, x_j^{\neq 2})$. We know that $f(x_i) \neq 1$ or $f(x_j) \neq 2$. Assume, for example, that $f(x_i) = 2$ and $f(x_j) = 2$. We see that $f'(x_i^{\neq 1}) = c_2$ and $f'(x_j^{\neq 2}) = c_2$ so $f'$ satisfies this constraint. The other cases can be verified analogously.

Assume that $f' : V' \to U \cup \{c_1, c_2, c_3\}$ is a solution to $(V', C')$ where $c_i$, $1 \leq i \leq 3$, is the value assigned to variable $d_i$. Define $f : V \to \{1, 2, 3\}$ such that $f(x_i) = p$ when $f(x'_i) = c_p$. Arbitrarily choose a constraint $R(x_i, x_j) \in C$ and assume to the contrary that $(f(x_i), f(x_j)) = (a, b) \notin R$. This implies that $f'$ satisfies the constraint $\mathtt{eq}_D(x'_i, x_i^{\neq a}) \vee \mathtt{eq}(x'_j, x_j^{\neq b})$ that was introduced in step 5. In order to do so, either $f'(x'_i) = f'(x_i^{\neq a})$ and $f'(x'_i) \neq c_a$ or $f'(x'_j) = f'(x_j^{\neq b})$ and $f'(x'_j) \neq c_b$. In both cases, $(f(x_i), f(x_j)) \neq (a, b)$ and this leads to a contradiction. ◀

If we consider $\mathcal{B}^{\vee k}$ with larger $k$ and require that certain relations are members of $\mathcal{B}$, then stronger lower bounds can be obtained.

▶ **Theorem 11.** *Let $\prec \subseteq D^2$ be a strict partial order of infinite height over a domain $D$. If the r-ETH holds, then there is no $2^{c \cdot n}$-randomised algorithm for CSP($\{\prec\}^{\vee 4}$) for any $c \geq 0$.*

**Proof.** Assume there exists a $2^{c \cdot n}$-randomised algorithm for CSP($\{\prec\}^{\vee 4}$). Arbitrarily choose $d \geq 3$ such that $c_{d,2} > c$. We show how to polynomial-time reduce CSP($\Gamma_{d,2}$) to CSP($\{\prec\}^{\vee 4}$) in a way such that only a constant number of new variables are introduced. This implies that CSP($\Gamma_{d,2}$) can be solved by a $2^{c \cdot n}$-randomised algorithm where $c < c_{d,2}$ which contradicts the r-ETH due to Traxler's result.

Let $I = (V, C)$ be an arbitrary instance of CSP($\Gamma_{d,2}$). We assume (without loss of generality) that the variable domain is $\{1, \ldots, d\}$. Introduce $d + 1$ fresh variables $V_1 = \{a_1, \ldots, a_{d+1}\}$ and define $C_1 = \{a_1(\prec)a_2, a_2(\prec)a_3, \ldots, a_d(\prec)a_{d+1}\}$. Since $\prec$ is a strict partial order of infinite height, we know that $I_1 = (V_1, C_1)$ is satisfiable. In every solution $s$, it holds that $s(a_i) \prec s(a_j)$ when $1 \leq i < j \leq d + 1$ by the transitivity of $\prec$. We then constrain each $x \in V$ as follows: $a_1(\prec)x$, $x(\prec)a_i \vee a_i(\prec)x$ for $2 \leq i \leq d$, and $x(\prec)a_{d+1}$. Let $C_2$ denote the corresponding set of constraints and let $I_2 = (V \cup V_1, C_1 \cup C_2)$. It is easy to verify that in every solution $s$ to $I_2$, each variable $x \in V$ satisfies $s(a_i) \prec s(x) \prec s(a_{i+1})$ for exactly one $1 \leq i \leq d$. For each constraint $S(x, y)$ in $C$, we finally introduce the following set of constraints $\{x(\prec)a_e \vee a_{e+1}(\prec)x \vee y(\prec)a_{e'} \vee a_{e'+1}(\prec)y \mid (e, e') \notin S\}$.

Let $C_3$ denote the resulting set of constraints and let $I_3 = (V \cup V_1, C_1 \cup C_2 \cup C_3)$. We claim that $I_3$ is satisfiable if and only if $I$ is satisfiable. Assume that $I_3$ has the solution $s_3$. We know that every variable $v$ in $V$ satisfies $s(a_i) \prec s(v) \prec s(a_{i+1})$ for exactly one $1 \leq i \leq d$. The constraints in $C_3$ assure that $s_3$ assigns values to the variables in $V$ that are consistent with the constraints in $(V, C)$. Thus, the function $s : V \to D$ defined by $s(v) = i$ where $v \in V$ and $s_3(a_i) \prec s_3(v) \prec s_3(a_{i+1})$ is a solution to $I$.

Assume that $I$ has the solution $s$. We construct a solution $s_3$ to $I_3$ as follows. Arbitrarily choose $e_1, \ldots, e_{d+1}, e'_1, \ldots, e'_d$ in $D$ such that $e_i \prec e'_i \prec e_{i+1}$, $1 \leq i \leq d$; such elements exists since $\prec$ has infinite height. Let $s_3(a_i) = e_i$, $1 \leq i \leq d+1$. This choice satisfies all constraints in $C_1$. Let $s_3(v) = e'_i$, $v \in V$, when $s(v) = i$. It follows from the choice of $e_1, \ldots, e_{d+1}, e'_1, \ldots, e'_d$ that all constraints in $C_2$ are satisfied. Finally, $s_3$ satisfies the constraints in $C_3$: this is an immediate consequence of $s$ being a solution to the instance $I$ combined with the restrictions imposed by the constraints in $C_1 \cup C_2$.

Last, we verify that $I_3$ can be computed in polynomial time. The constraints in $C_1$ and $C_2$ can be computed in constant time since $d$ is fixed, and each constraint in $C$ gives rise to at most $d^2$ new constraints in $C_3$, so this set can trivially be computed in polynomial time. ◀

The bound in Theorem 11 is substantially stronger than the bounds that we have been able to prove for CSP($\mathcal{B}^{\vee=}$). We may also observe that CSP($\{\prec\}^{\vee k}$), $k \geq 1$, is solvable in $O(|V|! \cdot \text{poly}(||I||)) = 2^{O(|V| \log |V|)} \cdot \text{poly}(||I||)$ time, implying that the lower bound in Theorem 11 does not admit large improvements (unless r-ETH fails).

▶ **Theorem 12.** *Let $\prec \subseteq D^2$ be a strict partial order of infinite height over a domain $D$, and let $k \geq 1$. Then CSP($\{\prec\}^{\vee k}$) is solvable in $O(|V|! \cdot \text{poly}(||I||))$ time.*

**Proof.** Let $(V, C)$ be an instance of CSP($\{\prec\}^{\vee k}$). For each total order $\sqsubset$ over $V$, we answer yes if there for every disjunctive clause in $C$ exists a disjunct $x \prec y$ such that $x \sqsubset y$. The time complexity of this algorithm is clear, and we now turn to correctness. Assume first that $f$ is a satisfying assignment to $(V, C)$. Let $C'$ denote the set of all disjuncts satisfied by $f$. This set induces a strict partial order which can be extended into a total order by topological sorting. For the other direction, assume that $\sqsubset$ satisfies at least one disjunct in every clause. Let $i_1, \ldots, i_{|V|} \subseteq \{1, \ldots, |V|\}$ be indices such that $x_{i_1} \sqsubset \ldots \sqsubset x_{i_{|V|}}$ and $|\{i_1, \ldots, i_{|V|}\}| = |V|$. Since $\prec$ is of infinite height there then exists $d_1, \ldots, d_{|V|} \in D$ such that $d_1 \prec \ldots \prec d_{|V|}$, and we can form a satisfying assignment $f$ by letting $f(x_{i_j}) = d_j$ for every $i_j \in \{i_1, \ldots, i_{|V|}\}$. ◀

## 6 Discussion

Our main focus has been to study the complexity of CSPs over partition schemes $\mathcal{B}$, with a particular emphasis on CSP($\mathcal{B}^{\vee=}$) when $\mathcal{B}$ contains a strict partial order. We have identified three properties resulting in NP-hardness, which explains the NP-hardness for many different CSP problems. Towards a better understanding of the time complexity of these problems we have also proven lower bounds under complexity-theoretic assumptions. We have studied lower bounds for CSP($\mathcal{B}^{\vee k}$), too, and obtained general bounds for this kind of problems. At this stage it is worth to yet again point out that none of our results require model-theoretic assumptions such as $\omega$-categoricity, i.e., that the first-order theory of $\mathcal{B}$ admits only one model up to isomorphism. A large amount of research on infinite-domain CSPs has concentrated on $\omega$-categorical constraint languages. However, there are interesting problems that are not amenable using this approach.

▶ **Example 13.** Bodirsky and Jonsson [5, Sec. 4.2] present a partition scheme $\mathcal{B}$ with domain $\mathbb{R}^3$ that demonstrate how to integrate arithmetics into partition schemes. They show that $\mathcal{B}$ is not $\omega$-categorical and there does not exist any $\omega$-categorical constraint language $\Gamma$ such that CSP($\Gamma$) and CSP($\mathcal{B}$) is the same computational problem. We will not define $\mathcal{B}$ explicitly, but remark that the relation $\text{Less} = \{((a, b, p), (c, d, q)) \subseteq (\mathbb{R}^3)^2 \mid a < c \wedge p \neq q\}$ is a member of $\mathcal{B}^{\vee=}$. Obviously, the constraints Less($d_1, d_2$) and Less($d_2, d_3$) force $d_1, d_2, d_3$

to be assigned distinct values. By definition, there exists relations $B_1, \ldots, B_k \in \mathcal{B}$ such that Less $= B_1 \cup \cdots \cup B_k$ so there exist (not necessarily distinct) $1 \leq i, j \leq k$ such that the constraints $B_1(d_1, d_2)$ and $B_2(d_2, d_3)$ force $d_1, d_2, d_3$ to be assigned distinct values, too. We know that $\mathsf{eq}_D \in \mathcal{B}$ since $\mathcal{B}$ is a partition scheme. We conclude (by Theorem 10) that CSP($\mathcal{B}^{\vee 2}$) cannot be solved in $O(2^{\frac{c_{3,2}}{5}n})$ time.

One important consequence of lower bound results is that they can be used to rule out certain types of algorithms. First of all, $k$-consistency algorithms are not applicable since they run in polynomial time for arbitrary fixed $k$. The powerful generalisation of $k$-consistency, the *Datalog* framework [11, 4], is not applicable either since every Datalog program runs in polynomial time, too. Another example is provided by graph-decomposition algorithms for CSPs (for instance, algorithms that exploit treewidth). Such algorithms have been highly influential in the CSP context [1, 3, 7], but they typically result in polynomial-time or subexponential algorithms and are therefore unlikely to be usable for CSP($\mathcal{B}^{\vee =}$) problems. Even more can be said if we take a detour via degree-bounded problems.

▶ **Lemma 14.** *Let $\mathcal{B}$ be a constraint language such that CSP($\mathcal{B}$) is solvable in polynomial time. For arbitrary constants $B$ and $k$, CSP($\mathcal{B}^{\vee k}$)-$B$ can be solved in $2^{B \cdot \log k \cdot |V|} \cdot poly(||I||)$ time and CSP($\mathcal{B}^{\vee =}$)-$B$ can be solved in $2^{B \cdot \log(|\mathcal{B}|-1) \cdot |V|} \cdot poly(||I||)$ time.*

**Proof.** Let $I = (V, C)$ be an arbitrary instance of CSP($\mathcal{B}^{\vee k}$)-$B$. Pick one disjunct out of each constraint in $C$, put the disjuncts into the set $S$, and check whether $S$ is satisfiable or not. This check can be performed in polynomial time. There are at most $k^{B \cdot |V|}$ different sets $S$ since each constraint contains at most $k$ disjuncts and there are at most $B \cdot |V|$ constraints in $C$. Furthermore, $(V, C)$ is satisfiable if and only if at least one of them is satisfiable. We conclude that $(V, C)$ can be solved in $k^{B \cdot |V|} \cdot poly(||I||)$ time. The proof for CSP($\mathcal{B}^{\vee =}$)-$B$ is essentially identical, with the difference that we never need to consider a relation containing all relations in $\mathcal{B}$, explaining $|\mathcal{B}| - 1$ in the exponent. ◀

Thus, both CSP($\mathcal{B}^{\vee k}$)-$B$ and CSP($\mathcal{B}^{\vee =}$)-$B$ can be solved in $2^{O(n)}$ time. We know (from Theorem 11) that there is no $2^{cn}$-randomised algorithm for the CSP($\{\prec\}^{\vee 4}$) problem. This shows that techniques used for transforming CSP instances into sparse instances, e.g. *linear kernelisations* [19], are unlikely to be applicable to CSP($\mathcal{B}^{\vee k}$). We cannot rule out linear kernelisations for CSP($\mathcal{B}^{\vee =}$), though, since we do not have a sufficiently strong lower bound in this case.

Naturally, there are approaches that are not directly ruled out by our lower bounds. Jonsson and Lagerkvist [16] have presented general results for obtaining algorithms based on enumeration of domain values. These algorithms are sometimes much faster than the branching algorithms that are typically used for infinite-domain CSPs: the branching algorithm for CSP($\mathcal{A}^{\vee =}$) runs in $2^{O(n^2)}$ time while the enumeration-based algorithm runs in $2^{O(n \log n)}$ time. The range of applicability for enumeration-based algorithms is unfortunately not well understood, and more work is needed to clarify this. Another viable approach is to use methods that have been successful in solving finite-domain CSPs. Einarson [10] demonstrates how the finite-domain version of the PPSZ algorithm [14] can be applied to infinite-domain CSPs. His results are inconclusive: the algorithm is faster than previously known algorithms for certain CSP($\mathcal{B}^{\vee k}$) problems but it is, for instance, not competetive for Allen's interval algebra CSP($\mathcal{A}^{\vee =}$).

These examples suggest that it may be worthwhile to strengthen the subexponential lower bound for CSP($\mathcal{B}^{\vee =}$) even further – if possible. One possible way of doing this is to exploit the *strong exponential-time hypothesis*, i.e. the conjecture that SAT is not solvable in $O^*(c^n)$

time for any $c < 2$, The challenge here is that the SETH intrinsically requires reductions where one can "simulate" clauses of arbitrary high arity with a very small overhead. This seems difficult for CSP($\mathcal{B}^{\vee=}$) and in Theorem 8 we could only produce a reduction from a SAT problem with a linear number of constraints. This assumption cannot be made for SAT since sparsification, the process of reducing an instance to a subexponential number of instances with a linear number of constraints, is not possible for SAT [28]. Another possibility is to use bounds based on the Chromatic Number problem: Jonsson and Lagerkvist [16, Th. 21] have related the time complexity of Allen's interval algebra with the time complexity of the Chromatic Number problem and obtained concrete lower bounds of the form $O^*(c^n)$ for a constant $c > 1$ depending on the complexity of Chromatic Number. Thus, we ask the following: should stronger lower bounds for CSP($\mathcal{B}^{\vee=}$) be pursued in the setting of CNF-SAT and the SETH, or are problems of this kind fundamentally closer to e.g. colouring problems?

## References

1 J. Alber, H. Fernau, and R. Niedermeier. Parameterized complexity: exponential speed-up for planar graph problems. *J. Algorithms*, 52(1):26–56, 2004.

2 J. F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11):832–843, 1983.

3 S. Arora, B. Barak, and D. Steurer. Subexponential algorithms for unique games and related problems. *J. ACM*, 62(5):42:1–42:25, 2015.

4 M. Bodirsky and V. Dalmau. Datalog and constraint satisfaction with infinite templates. *J. Comput. Syst. Sci.*, 79(1):79–100, 2013.

5 M. Bodirsky and P. Jonsson. A model-theoretic view on qualitative constraint reasoning. *J. Artif. Intell. Res. (JAIR)*, 58:339–385, 2017.

6 A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS-2017)*, pages 319–330, 2017.

7 R. de Haan, I. A. Kanj, and S. Szeider. On the subexponential-time complexity of CSP. *J. Artif. Intell. Res.*, 52:203–234, 2015.

8 I. Düntsch. Relation algebras and their application in temporal and spatial reasoning. *Artif. Intell. Rev*, 23(4):315–357, Jun 2005.

9 F. Dylla, J. H. Lee, T. Mossakowski, T. Schneider, A. van Delden, J. van de Ven, and D. Wolter. A survey of qualitative spatial and temporal calculi: Algebraic and computational properties. *ACM Comput. Surv.*, 50(1):7:1–7:39, 2017.

10 C. Einarson. An extension of the PPSZ algorithm to infinite-domain constraint satisfaction problems. Master's thesis report, Department of Computer and Information Science, Linköpings Universitet, 2017.

11 T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1998.

12 M. Grigni, D. Papadias, and C. H. Papadimitriou. Topological inference. In *Proc. 14th International Joint Conference on Artificial Intelligence (IJCAI-1995)*, pages 901–907, 1995.

13 H. Güsgen. Spatial reasoning based on Allen's temporal logic. Technical report ICSI TR89-049, International Computer Science Institute, 1993.

14 T. Hertli, I. Hurbain, S. Millius, R. A. Moser, D. Scheder, and M. Szedlák. The PPSZ algorithm for constraint satisfaction problems on more than two colors. In *Proc. 22nd International Conference on Principles and Practice of Constraint Programming (CP-2016)*, pages 421–437, 2016.

15 R. Impagliazzo and R. Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

**16** P. Jonsson and V. Lagerkvist. An initial study of time complexity in infinite-domain constraint satisfaction. *Artif. Intell.*, 245:115–133, 2017.

**17** P. Jonsson, V. Lagerkvist, G. Nordh, and B. Zanuttini. Strong partial clones and the time complexity of SAT problems. *J. Comput. Syst. Sci.*, 84:52–78, 2017.

**18** A. Krokhin, P. Jeavons, and P. Jonsson. Reasoning about temporal relations: The tractable subclasses of Allen's interval algebra. *J. ACM*, 50(5):591–640, 2003.

**19** V. Lagerkvist and M. Wahlström. Kernelization of constraint satisfaction problems: A study through universal algebra. In *Proc. 23rd International Conference on Principles and Practice of Constraint Programming (CP-2017)*, pages 157–171, 2017.

**20** G. Ligozat and J. Renz. What is a qualitative calculus? A general framework. In *Proc. 8th Pacific Rim International Conference on Artificial Intelligence (PRICAI-2004)*, pages 53–64, 2004.

**21** R. Moratz, J. Renz, and D. Wolter. Qualitative spatial reasoning about line segments. In *Proc. 14th European Conference on Artificial Intelligence (ECAI-2000)*, pages 234–238, 2000.

**22** A. Mukerjee and G. Joe. A qualitative model for space. In *Proc. 8th National Conference on Artificial Intelligence (AAAI-1990)*, pages 721–727, 1990.

**23** J. Opatrny. Total ordering problem. *SIAM J. Comput.*, 8(1):111–114, 1979.

**24** J. Renz. Qualitative spatial and temporal reasoning: Efficient algorithms for everyone. In *Proc. 20th International Joint Conference on Artifical Intelligence (IJCAI-2007)*, pages 526–531, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

**25** J. Renz and J. J. Li. Automated complexity proofs for qualitative spatial and temporal calculi. In *Proc. Principles of Knowledge Representation and Reasoning (KR-2008)*, pages 715–723, 2008.

**26** J. Renz and B. Nebel. On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artif. Intell.*, 108(1–2):69–123, 1999.

**27** J. Renz and B. Nebel. Qualitative spatial reasoning using constraint calculi. In Marco Aiello, Ian Pratt-Hartmann, and Johan van Benthem, editors, *Handbook of Spatial Logics*, pages 161–215. Springer, 2007.

**28** R. Santhanam and S. Srinivasan. On the limits of sparsification. In *Proc. 39th International Colloquium on Automata, Languages, and Programming (ICALP-2012)*, pages 774–785, 2012.

**29** P. Traxler. The time complexity of constraint satisfaction. In *Proc. 3rd International Workshop on Parameterized and Exact Computation (IWPEC-2008)*, pages 190–201, 2008.

**30** D. Zhuk. A proof of CSP dichotomy conjecture. In *Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS-2017)*, pages 331–342, 2017.