Artificial Intelligence 296 (2021) 103505

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Acyclic orders, partition schemes and CSPs: Unified hardness proofs and improved algorithms ☆

Peter Jonsson, Victor Lagerkvist *, George Osipov

Department of Computer and Information Science, Linköpings Universitet, Campus Valla, Sweden

A R T I C L E   I N F O

Article history:
Received 29 June 2020
Received in revised form 25 January 2021
Accepted 12 April 2021
Available online 15 April 2021

Keywords:
Constraint satisfaction problems
Infinite domains
Partition schemes
Lower bounds

A B S T R A C T

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 an acyclic 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 properties of such orders which, when combined, are sufficient to establish NP-hardness of the CSP and strong lower bounds under the exponential-time hypothesis, even for degree-bounded problems. This result explains, in a uniform way, many existing hardness results from the literature, and shows that it is impossible to obtain subexponential time algorithms unless the exponential-time hypothesis fails. However, some of these problems (including several important temporal problems), despite likely not being solvable in subexponential time, admit non-trivial improved exponential-time algorithm, and we present a novel improved algorithm for RCC-8 and related formalisms.

1. Introduction

In this article we study the complexity of infinite-domain *constraint satisfaction problems* over *partition schemes*. In this framework one can formulate many naturally occurring reasoning problems in artificial intelligence such as *Allen's interval algebra* and the *region connection calculus*. We identify shared properties among these problems, based on the existence of *acyclic orders*, and use these properties to provide a general NP-hardness result and prove strong lower bounds under the *exponential-time hypothesis*. Importantly, to the best of our knowledge, this is the first lower bound under the exponential-time hypothesis for problems of this form. Motivated by these lower bounds we also turn to the problem of constructing improved algorithms for CSPs over partition schemes, with a particular focus on the region connection calculus.

1.1. Background

The *constraint satisfaction problem* over a constraint language Γ (CSP(Γ)) is the decision problem of verifying whether a set of constraints based on the relations in Γ admits a satisfying assignment. For finite domains the complexity of

☆ Parts of this article appeared in the proceedings of the 43rd International Symposium on Mathematical Foundations of Computer Science (MFCS-2018).

* Corresponding author.
E-mail addresses: peter.jonsson@liu.se (P. Jonsson), victor.lagerkvist@liu.se (V. Lagerkvist), george.osipov@liu.se (G. Osipov).

https://doi.org/10.1016/j.artint.2021.103505
0004-3702/

CSP(Γ) is well understood due to the recent dichotomy theorem separating tractable from NP-complete problems [9,42], 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 Γ, usually satisfying certain model-theoretic properties, and analyse the complexity of CSPs over first-order reducts of Γ. 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 [29] is a set of pairwise disjoint binary relations $\mathcal{B}$ over a domain D such that $\bigcup_{R \in \mathcal{B}} R = D^2$, the equality relation is a member of $\mathcal{B}$, and the converse of R is in $\mathcal{B}$ for every $R \in \mathcal{B}$. Due to their capability of modelling many different kinds of qualitative reasoning problems, partition schemes are the de facto standard for CSPs in the artificial intelligence community [13]. 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. Equivalently, each relation in $\mathcal{B}^{\vee=}$ can be defined as a disjunction of the form $B_1(x, y) \vee B_2(x, y) \vee \cdots \vee B_k(x, y)$ for some $\{B_1, \ldots, B_k\} \subseteq \mathcal{B}$, and the set $\mathcal{B}^{\vee=}$ thus contains all relations definable in this way.

Famous AI examples of formalisms based on partition schemes include *Allen's interval algebra*, the *region connection calculus* (RCC-8), and the *rectangle algebra*. For more examples, see e.g. the survey by Dylla et al. [14]. 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. Thus, does there exist *one* reduction applicable to *every* partition scheme? Or does one need separate proofs for (e.g.) RCC-8 and Allen's interval algebra, and then perform an exhaustive case analysis on all possible set of base relations $\mathcal{B}$?

Thus, while quite a bit is known about specific partition schemes, it is safe to say that we lack a more general understanding of *why* these problems are NP-hard. When turning to questions of more fine-grained complexity the situation is even more dire. For example, can CSPs over partition schemes be solved roughly as fast as Boolean satisfiability problems, i.e., in $O(2^n)$ time? Or at least as fast as a finite-domain CSP, i.e., $O(c^n)$ for some $c \geq 1$? Classifying partition schemes according to this highly "fine-grained" level of complexity is a very hard open problem which we should not hope to immediately resolve, and as a first sanity check it is common to first rule out the existence of *subexponential* algorithms, i.e., algorithms with a running time of $2^{o(n)}$. Naturally, this cannot be done unconditionally, and a popular assumption in this context is the *exponential-time hypothesis* (ETH) which states that the 3-SAT problem cannot be solved in subexponential time. Despite the fact that CSPs over partition schemes are among the most frequently studied CSPs in artificial intelligence, it is fair to say that such ETH-based lower bounds have largely been neglected by both the artificial intelligence community and the CSP community, and to the best of our knowledge there are *no* concrete lower bounds under the ETH for these problems. There are a few reasons for this. First, significant efforts have been made to solve hard reasoning problems with efficient heuristics [35], 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. 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 [25] which relates the complexity of Allen's interval algebra to the complexity of the Chromatic Number problem (but not under the more common assumption ETH).

1.2. Our results

Our first step (in Section 3) is to note that the majority of practically relevant partition schemes contain *acyclic orders* satisfying certain properties, which we in this article refer to as *unbounded total orders*, *in-forks*, *out-forks*, and *no-forks*. For brevity, we typically refer to these conditions as C_1, C_2, C_3 and C_4. We provide several examples from the literature of partition schemes containing acyclic orders of this form, e.g., Allen's interval algebra, the unit interval algebra, and RCC-8. While there is no general recipe for proving that a partition scheme contains an acyclic order of the above form, all the examples that we have tried show that it in practice is rather straightforward.

In Section 4 we turn to the problem of proving lower bounds for CSPs over partition schemes containing an acyclic order satisfying C_1–C_4. Through a sophisticated reduction we prove that no problem of this form can be solved in subexponential time (under the ETH), even in the structurally restricted case where a variable may occur in at most 3 constraints. The case when a variable may occur in at most 2 constraints is handled in Section 4.3 where we establish that all CSPs of this form are tractable if the template in question is *ω-categorical*.

In particular, our results imply that CSPs over partition schemes containing acyclic orders of this form are NP-hard. Hence, we succeed both in finding a uniform hardness proof, and with proving lower bounds under the ETH. Importantly, our lower bounds are the first ETH-based lower bounds for CSPs of this form. It might also be interesting to observe that we do not need any strong model-theoretic properties, e.g. ω-categoricity, which is otherwise common for infinite-domain CSPs. The reduction establishing NP-hardness is also interesting to compare to the procedure by Renz and Li [36] 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.

One way of interpreting these results is that CSP($\mathcal{B}^{\vee=}$), when $\mathcal{B}$ is a partition scheme containing an acyclic order satisfying the aforementioned properties, 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 consequence 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 [25,39]. Improvements to $2^{O(n \log n)}$ are possible for certain *temporal CSPs* and for Allen's interval algebra [25], so there is reason for being optimistic. We attack this question in Section 5 where we begin by presenting an $2^{O(n)}$ time algorithm for the degree-bounded case, and then present a novel $2^{O(n \log n)}$ time algorithm for RCC-8. This is the first example of a non-trivial spatial partition scheme whose CSP can be solved in $2^{o(n^2)}$ time. Classifying partition schemes admitting improved algorithms of this form is an interesting open question, which we discuss in greater detail in Section 6, among other unresolved questions.

2. Preliminaries

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

Definition 1. Let Γ be a set of finitary relations over some set D of values. The *constraint satisfaction problem* over Γ (CSP(Γ)) 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 Γ is called a *constraint language*, and the function f is sometimes called a *satisfying assignment*, or a *solution*. Importantly, the domain D is allowed to be infinite, and CSPs over constraint languages over infinite domains are typically called *infinite-domain CSPs*, to contrast them with *finite-domain* CSPs. Given an instance I of CSP(Γ) 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(Γ). If a variable x occurs in B (distinct) constraints in C, then we say that the degree of x is B. We let CSP(Γ)-B denote the CSP(Γ) problem where each variable in the input is restricted to have degree at most B. Note that if (V, C) is a CSP(Γ)-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 are now ready to introduce *partition schemes* [29]. Let $\mathcal{B} = \{B_1, \ldots, B_m\}$ be a constraint language consisting 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 jointly exhaustive and pairwise disjoint, (2) $\mathrm{eq}_D = \{(x, x) \mid x \in D\} \in \mathcal{B}$, 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 $\mathcal{B}$. Note that each such relation is still binary, i.e., of arity 2. 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(Γ) where $\Gamma \subseteq \mathcal{B}^{\vee=}$ are typical objects that are studied in the 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 CSP($\mathcal{B}^{\vee=}$) is a member of NP.

Example 1. *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. Note that CSP($\mathcal{A}^{\vee=}$) is an infinite-domain CSP since the underlying domain of real intervals is intrinsically infinite.

An extension of the interval algebra is the so-called *rectangle algebra* [19,32]. 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 $I_y(s) J_y$.

Example 2. *RCC-8* [34] is a formalism for qualitative spatial reasoning where the basic objects (referred to as *regions*) are non-empty regular closed subsets of a topological space. The regions do not have to be internally connected, that is, they

Table 1
The thirteen basic relations in Allen's interval algebra. The endpoint relations $I^- < I^+$ and $J^- < J^+$ that are valid for all relations have been omitted.

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

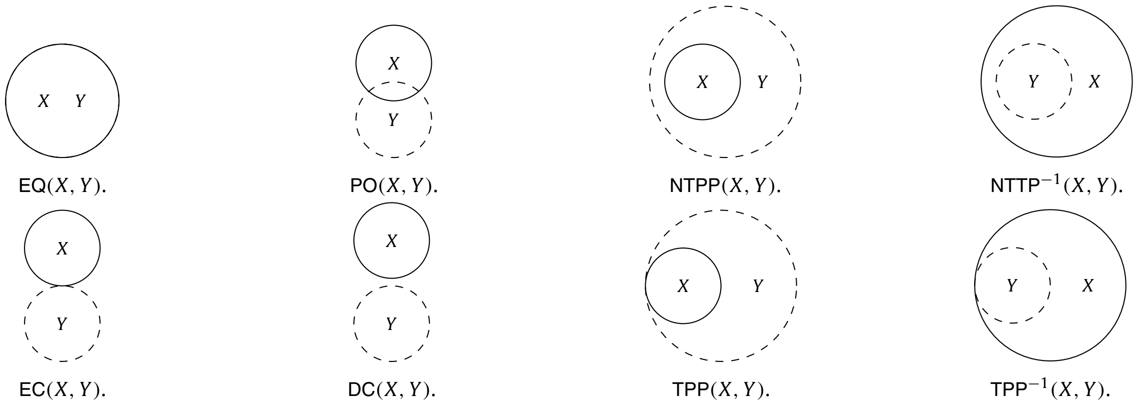

Fig. 1. Illustration of the basic relations of RCC-8 with two-dimensional disks.

may consist of different disconnected pieces. RCC-8 contains eight basic relations: EQ (equal), PO (partial overlap), DC (disconnected), EC (externally connected), NTPP (non-tangential proper part), its converse NTPP^{-1}, TPP (tangential proper part) and its converse TPP^{-1}. See Fig. 1 for examples. RCC-5 is a variant of RCC-8 where one is not able to distinguish regions from their topological closure, i.e. the distinction between boundary points and interior points is ignored. Thus, the disconnectedness relations DC and EC are replaced by $\mathsf{DR} = \mathsf{DC} \cup \mathsf{EC}$, the tangential and non-tangential proper part relations TPP and NTPP are replaced by $\mathsf{PP} = \mathsf{TPP} \cup \mathsf{NTPP}$, and PP^{-1} is defined analogously.

Last, we define two satisfiability problems useful in the context of lower bounds, which we will return to in Section 4.1. If V is a set of variables and f a function from V to $\{0, 1\}$, then we define the function h_f as $h_f(x) = f(x)$ and $h_f(\neg x) = 1 - f(x)$ for any $x \in V$. For $k \geq 1$ we define the *k-satisfiability* (k-SAT) problem as follows.

Instance: A set of variables V and a set clauses of the form $(\ell_1 \vee \ldots \vee \ell_k)$, where $\ell_i = \neg x$ or $\ell_i = x$ for some $x \in V$.
Question: Does there exist a function $f : V \to \{0, 1\}$ such that $h_f(\ell_1) + \ldots + h_f(\ell_k) \geq 1$ for each clause $(\ell_1 \vee \ldots \vee \ell_k)$?

The variant of k-SAT where we in addition require that the function f does not assign the same value to each literal in any clause is known as the NOT-ALL-EQUAL-k-SATISFIABILITY problem (NAE-k-SAT). NAE-3-SAT remains NP-hard even if it is restricted to clauses containing only positive literals, and we invite the reader to verify that this restricted problem can be formulated as a Boolean CSP over the template $N = \{0, 1\}^3 \setminus \{(0, 0, 0), (1, 1, 1)\}$.

3. Acyclic orders

CSPs based on partition schemes are 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 [39, p. 161] write

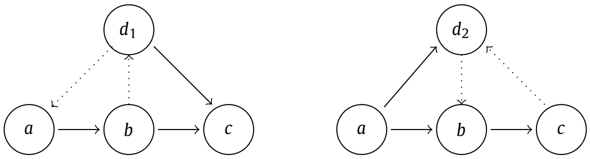

Fig. 2. Illustration of in-fork (left) and out-fork (right). Solid arrows denote the $\prec$ relation and dotted arrows the $\sqcap$ relation.

> 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 orders that satisfy certain additional properties. 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. Inspired by this, we present (in Section 3.1) a collection of four properties that capture some common aspects of "part-of" relations. Many other relations (that are not necessarily "part-of" relations) satisfy these properties, too: one example is the precedes relation p in Allen's algebra. In fact, relations of this kind appear very frequently in CSPs for qualitative reasoning and we present a number of examples in Section 3.2. Naturally, while our conditions on acyclic orders are sufficient to establish hardness, there exists CSPs over other types of partition schemes which are of independent interest. For example, if we leave the realm of qualitative reasoning, then the *graph interval sandwich problem* [17] can be phrased as a CSP over the two relations $\mathsf{p} \cup \mathsf{p}^{-1}$ and $\bigcup_{R \in \mathcal{A} \setminus \{\mathsf{p},\mathsf{p}^{-1}\}} R$, neither of which is acyclic. Another relevant example is the partition scheme $\{\mathsf{eq}_D, \mathsf{neq}_D\}$ over a countably infinite D. While the tractable problem CSP($\{\mathsf{eq}_D, \mathsf{neq}_D\}$) might not be terribly interesting, the optimisation variant of this problem where one is interested in maximising the number of satisfied constraints, MAX-CSP($\{\mathsf{eq}_D, \mathsf{neq}_D\}$), can be viewed as an alternative formulation of *correlation clustering* [5].

3.1. Conditions on acyclic orders

Let $\prec \subseteq D^2$ denote a binary relation and let $\succ$ denote its converse $\prec^{\smile}$. 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. It is easy to verify that strict partial orders are acyclic orders but that there exist acyclic orders that are not strict partial orders. 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. We will now define additional properties of acyclic orders particularly relevant in the context of CSP($\mathcal{B}^{\vee=}$). Here, we invite the reader to view the relation $\sqcap$ which holds between two elements if they are not comparable with respect to $\prec$, although all that is needed is that $\sqcap$ satisfies the following properties.

Definition 2. Let $\prec \subseteq D^2$ be an acyclic order and $\sqcap \subseteq D^2$ a relation. We define the following properties.

C_1. (unbounded total orders) 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,
C_2. (in-fork) if $a, b, c \in D$, $a \prec b \prec c$, and $a \prec c$, then there exists $d_1 \in D$ such that $d_1 \sqcap a$, $d_1 \sqcap b$, and $d_1(\prec, \succ)c$,
C_3. (out-fork) if $a, b, c \in D$ and $a \prec b \prec c$, and $a \prec c$, then there exists $d_2 \in D$ such that $d_2(\prec, \succ)a$, $d_2 \sqcap b$, and $d_2 \sqcap c$, and
C_4. (no-fork) if $a, b, c \in D$ and $a \prec b \prec c$, and $a \prec c$, then there does not exist any $d_3 \in D$ such that $d_3 \sqcap a$, $d_3(\prec, \succ)b$, and $d_3 \sqcap c$.

Relations satisfying these properties are abundant in the artificial intelligence literature, but they have to the best of our knowledge not been explicitly formalized before. The conditions in-fork and out-fork are illustrated in Fig. 2. Given a relation R it is typically easy to check if it is an acyclic order that contains an infinite total order, but checking C_2–C_4 may need additional work.

Many partition schemes contain strict partial orders, and these are slightly easier to work with than acyclic relations, since it is always possible to define the relation $\sqcap$ in a canonical way. Given an order $\prec \subseteq D^2$, we let $\curlywedge$ denote its *incomparability relation* $D^2 \setminus \bigcup\{\prec, \succ, \mathsf{eq}_D\}$. If $\mathcal{B}$ is a partition scheme and $\mathcal{B}^{\vee=}$ contains an order $\prec$, then $\curlywedge$ is included in $\mathcal{B}^{\vee=}$, too. We obtain the following characterization.

Theorem 3. *Let $\mathcal{B}$ be a partition scheme with domain D such that $\mathcal{B}^{\vee=}$ contains a strict partial order $\prec$. If $\prec$ and the incomparability relation $\curlywedge$ satisfy C_1–C_3 then they satisfy C_4.*

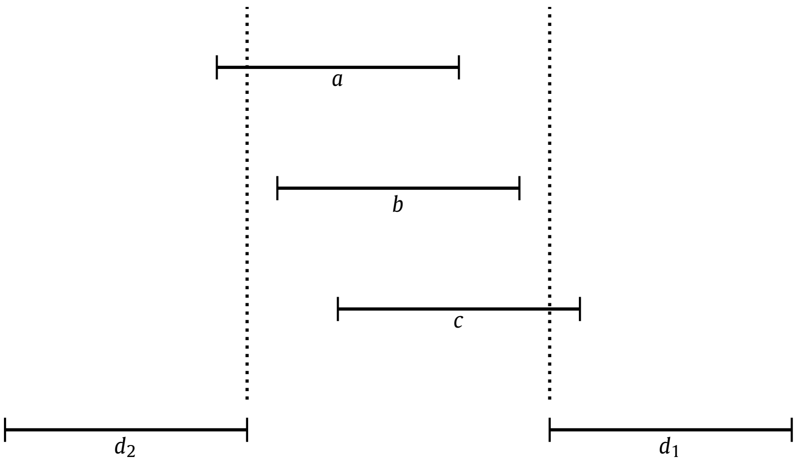

Fig. 3. The choice of d_1 and d_2 in the unit interval example.

Proof. Assume to the contrary that there exist $a, b, c, d_3 \in D$ such that $d_3 \curlywedge a$, $d_3(\prec, \succ)b$, and $d_3 \curlywedge c$. The relation $\prec$ is a strict partial order so it is transitive. If $d_3 \prec b$, then $d_3 \prec c$. But then $d_3 \curlywedge c$ cannot hold since the relations $\prec$ and $\curlywedge$ are disjoint. Similarly, if $d_3 \succ b$, then $a \prec d_3$ and $d_3 \curlywedge a$ cannot hold. $\square$

3.2. Examples

Consider Allen's algebra and the relation p, i.e. the relation stating that one interval appears strictly before another interval. In this case, $\sqcap$ can be chosen to be the relation that holds if and only if two distinct intervals have at least one point in common. The relation p is clearly acyclic (in fact, it is a strict partial order) and it contains many infinite strict total orders such as $T = \{[0, 1], [2, 3], [4, 5], \ldots\}$. Pick three intervals $I_j = [I_j^-, I_j^+] \in T$, $1 \le j \le 3$, such that $I_1(\mathrm{p})I_2(\mathrm{p})I_3$. The transitivity of p implies that $I_1(\mathrm{p})I_3$, too. 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 $I_5 = [I_2^-, I_3^+]$. Concerning the no-fork property, simply note that an interval I_6 which precedes I_2 (or is preceded by I_2) cannot share a point with both I_1 and I_3.

We continue with a more complex example that is based on acyclic orders instead of strict partial orders. The *Unit Interval Algebra* (UIA) is Allen's interval algebra restricted to intervals of length one. The UIA has important applications in, for instance, bioinformatics [33]. Consider the overlaps relation o. This relation is irreflexive but it is not transitive in general: the unit intervals [0, 1], [0.9, 1.9], and [1.8, 2.8] are examples of this. Hence, it is not a partial order. We show that o is an acyclic order that satisfies C_1–C_4. The fact that o is an acyclic order is easy to verify. Choose the relation $\prec$ to equal o and let $\sqcap$ hold if and only if two intervals do not have a common point, i.e., $\sqcap = (\mathrm{p}, \mathrm{p}^{-1})$.

C_1. We claim that the order (L, o) where $L = \{[x, x+1] \mid 0 < x < 1 \text{ and } x \in \mathbb{Q}\}$ is an infinite strict total order—this implies property C_1. Irreflexivity is obvious so we continue with the connexity property. We see that if $[a^-, a^+], [b^-, b^+]$ are distinct members of L, then either $-1 < a^- - b^- < 0$ or $0 < a^- - b^- < 1$. In the first case, it follows that $a^- < b^-$ and $1 + a^- > b^-$ which implies that $a^+ = 1 + a^- > b^- > a^-$ and $a(\mathrm{o})b$. The other case is analogous. To verify that (L, o) is transitive, we arbitrarily pick three distinct unit intervals $a, b, c = [a^-, a^+], [b^-, b^+], [c^-, c^+]$ in L such that $a(\mathrm{o})b(\mathrm{o})c$. We need to verify that $a^- < c^-$ and $c^- < a^+ < c^+$. The fact that $a(\mathrm{o})b(\mathrm{o})c$ implies that $a^- < b^- < c^-$ and, consequently, that $a^+ < c^+$ since $a^+ = a^- + 1$ and $c^+ = c^- + 1$. Finally, note that $c^- - a^- < 1$ so $c^- - (a^+ - 1) < 1$ and $c^- < a^+$.

C_2/C_3. We begin by arbitrarily choosing three distinct unit intervals $a, b, c = [a^-, a^+], [b^-, b^+], [c^-, c^+]$ in D such that $a(\mathrm{o})b(\mathrm{o})c$ and $a(\mathrm{o})c$. Let $d_1 = [\frac{c^+ - b^+}{2}, \frac{c^+ - b^+}{2} + 1]$. We see that $a(\mathrm{p})d_1$, $b(\mathrm{p})d_1$, and $c(\mathrm{o})d_1$ so $d_1 \sqcap a$, $d_1 \sqcap b$, and $d_1(\prec, \succ)c$. Let $d_2 = [\frac{b^- - a^-}{2} - 1, \frac{b^- - a^-}{2}]$. We see that $d_2(\mathrm{o})a$, $d_2(\mathrm{p})b$, and $d_2(\mathrm{p})c$. This implies that $d_2(\prec, \succ)a$, $d_2 \sqcap b$, and $d_2 \sqcap c$. The choice of d_1 and d_2 is illustrated in Fig. 3.

C_4. Pick three unit intervals $a, b, c = [a^-, a^+], [b^-, b^+], [c^-, c^+]$ in D such that $a(\mathrm{o})b(\mathrm{o})c$ and $a(\mathrm{o})c$. Assume to the contrary that there is a $d_3 \in D$ such that $d_3 \sqcap a$, $d_3(\prec, \succ)b$, and $d_3 \sqcap c$. If $d_3(\mathrm{p})a$, then d_3 cannot satisfy $d_3(\prec, \succ)b$ so $d_3(\mathrm{p}^{-1})a$. Similarly $d_3(\mathrm{p})c$. This contradicts the fact that $a \prec c$.

Let us consider another example where the domain contains the closed disks in $\mathbb{R}^2$, the relation $\prec$ is the strict subset relation, and where $\sqcap$ holds between two regions if and only if neither region included in the other. For each $k \ge 1$ it is then clear that there exists a set of k regions where $\prec$ induces a strict total order, e.g., k disks $c_1, c_2, \ldots, c_k$ where $c_1 \prec c_2 \ldots \prec c_k$. 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 Fig. 4. For the no-fork property, simply observe that any region containing (or is contained by)

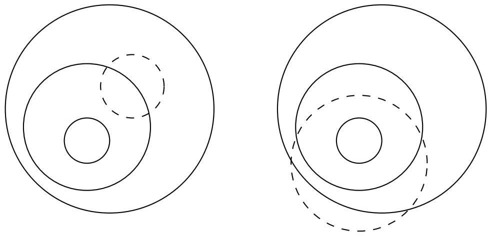

Fig. 4. The dashed circles show possible choices of disks for in-fork (left) and out-fork (right).

d_2 also contains d_1 (or is contained by d_3). This example can easily be adapted to relations such as PP in RCC-5, NTPP in RCC-8, and the relation $\mathsf{d} \oplus \mathsf{d}$ in the rectangle algebra.

The examples presented above are just a small selection of partition schemes that satisfy properties C_1–C_4 and many additional examples can be found, for instance, in the survey by Dylla et al. [14]. Last, let us remark that there are examples of strict partial orders that do not have in- and/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 $\mathcal{B}^{\vee=}$ contains an acyclic order $\prec$ and a relation $\sqcap$ that satisfy C_1–C_4. To avoid lengthy formulations of this kind we introduce the following set of templates.

Definition 4. We let $\mathcal{H}$ be the set of partition schemes $\mathcal{B}$ such that (1) CSP($\mathcal{B}$) is solvable in polynomial time, and (2) $\mathcal{B}$ contains an acyclic order $\prec$ and a relation $\sqcap$ that satisfy C_1–C_4.

Note that it is sufficient that the partition scheme contains a single acyclic order with these properties: the other relations are not relevant as long as CSP($\mathcal{B}$) is tractable. Examples where the connection between acyclic orders and the complexity of the resulting CSPs is quite pronounced can be found in, for instance, Grigni et al. [18], Renz and Nebel [37], Moratz et al. [31], and Krokhin et al. [27]. Thus, is CSP($\mathcal{B}^{\vee=}$) always NP-hard when $\mathcal{B} \in \mathcal{H}$, or can there exist tractable cases? If CSP($\mathcal{B}^{\vee=}$) is indeed NP-hard, how fast can it be solved? Might there exist some particularly "easy" partition scheme $\mathcal{B}$ where CSP($\mathcal{B}^{\vee=}$) is solvable in $O(c^n)$ for a very small constant c? Or even in $O(c^n)$ time for *every* constant $c > 1$, i.e., in *subexponential* time[1]? Naturally, we cannot hope to unconditionally prove that a CSP($\mathcal{B}^{\vee=}$) problem is not subexponential, and it is instead common to prove lower bounds subjected to the assumption that a specific problem is not solvable in subexponential time. For this purpose the 3-SAT problem, i.e., satisfiability of clauses of length at most 3, has turned out to be a very useful starting point.

Definition 5. The conjecture that 3-SAT is not solvable in subexponential time is known as the *exponential-time hypothesis* (ETH) [23].

The general idea behind a non-subexponentiality lower bound subjected to the ETH is then similar to a typical NP-hardness proof: one needs to provide a suitable reduction from 3-SAT to the problem in question. The complicating factor, of course, is that one needs reductions preserving subexponential complexity, which can sometimes be much more difficult to construct than ordinary polynomial-time many-one reductions. More information about the ETH and its consequences can be found in the survey by Lokshtanov et al. [30].

Example 3. Consider the classical gadget reduction from 4-SAT to 3-SAT which replaces a clause of the form $(x_1 \vee x_2 \vee x_3 \vee x_4)$ with $(x_1 \vee x_2 \vee y) \wedge (x_3 \vee x_4 \vee \neg y)$, where y is a fresh variable. If one is not careful then one might be led to believe that this reduction preserves subexponential complexity since we for each clause in the original instance only introduces one fresh variable. But what if the instance contains a superlinear amount of clauses with respect to the number of variables, e.g., a quadratic number? Assuming that 3-SAT is solvable in $O(c^n)$ time for some $c > 0$, where n is the number of variables, this reduction would then only say that 4-SAT is solvable in $O(c^{n^2})$ time, and in particular would not imply that 4-SAT is subexponential if 3-SAT is subexponential.

However, it is known that the degree-bounded k-SAT problem (for some fixed $B > 0$) is solvable in subexponential time if and only if k-SAT is solvable in subexponential time, using the powerful idea of *sparsification* [23]. If one then reduces

[1] Here, and in the rest of this section, n will always denote the number of variables in an instance of CSP($\mathcal{B}^{\vee=}$).

from a degree-bounded problem the total number of clauses is linear with respect to the number of variables, meaning that the above reduction preserves subexponential complexity.

Impagliazzo et al. [23] introduce a more general theory of reductions preserving subexponential complexity but for our purposes, it is sufficient with polynomial-time many-one reductions which given an instance with n variables produce an instance with $O(n)$ variables.

4.1. ETH-based lower bounds and NP-hardness

Arbitrarily choose $\mathcal{B}$ in $\mathcal{H}$. We have two challenges to overcome: first, is it possible to find a *uniform* reduction applicable to every CSP($\mathcal{B}^{\vee=}$) problem; second, can such a reduction, if it even exists, be used to obtain lower bounds under the ETH? We will reach an affirmative answer to both of these questions in this section, and will see that it is possible to obtain lower bounds even for degree-bounded CSP($\mathcal{B}^{\vee=}$)-B problems, indeed, even for the severely restricted problem CSP($\mathcal{B}^{\vee=}$)-3 where a variable may occur in at most 3 constraints. However, before we turn to the details we consider an example which shows a large difference between CSP($\mathcal{B}^{\vee=}$) problems and related CSPs, and highlight the involved difficulty.

Example 4. For a partition scheme $\mathcal{B}$ over the domain D, let $\mathcal{B}^{\vee k}$ be the set of all relations definable by disjunctions of length at most k, where each atom is a constraint over $\mathcal{B}$. It is easy to see that $\mathcal{B}^{\vee=} \subseteq \mathcal{B}^{\vee k}$ for some k, but that the converse is not necessarily true. In fact, $\mathcal{B}^{\vee k}$ is in general much more expressive than $\mathcal{B}^{\vee=}$ in the context of CSPs. To see this, consider the following reduction from 3-SAT to CSP($\mathcal{B}^{\vee 3}$). We only sketch the details since they are not important for the subsequent results.

1. Assume that the domain D contains at least two elements.
2. Since $\mathcal{B}$ is a partition scheme it always contains the equality relation eq_D and the inequality relation neq_D over D.
3. Introduce two fresh variables x_f and x_t and constrain them as $\mathrm{neq}_D(x_f, x_t)$.
4. For each 3-clause, e.g., $(x_1 \vee x_2 \vee \neg x_3)$ introduce the constraint $\mathrm{eq}_D(x_1, x_t) \vee \mathrm{eq}_D(x_2, x_t) \vee \mathrm{eq}_\mathrm{D}(x_3, x_f)$.

In other words this reduction is a standard gadget reduction from 3-SAT to CSP($\mathcal{B}^{\vee 3}$) which replaces each 3-clause by the corresponding disjunction over $\mathcal{B}$. Moreover, since it only introduces 2 fresh variables in total, it immediately follows that CSP($\mathcal{B}^{\vee 3}$) cannot be solved in subexponential time without violating the ETH (recall Example 3). Note that we do not even require any additional assumptions on $\mathcal{B}$: it is sufficient that it is a partition scheme.

For CSP($\mathcal{B}^{\vee=}$) the situation is *much* more difficult since it (in general) is not possible to represent disjunctions of the required form. We will soon see that while it is possible to obtain a suitable reduction to CSP($\mathcal{B}^{\vee=}$) when $\mathcal{B} \in \mathcal{H}$, it is significantly more complicated than the reduction in Example 4. Before we proceed with the actual reduction we define a useful gadget.

Lemma 6. *Assume that $\mathcal{B} \in \mathcal{H}$. Then there exists an instance of CSP($\mathcal{B}^{\vee=}$) with variables $\{a, b, c, x_1, x_2\}$ which satisfies the following properties:*

G_1. *For arbitrary elements $d_a, d_b, d_c \in D$ such that $d_a \prec d_b \prec d_c$ and $d_a \prec d_c$, there exist elements $d_1, d_2 \in D$ such that the function $s\colon V \to \{d_a, d_b, d_c, d_1, d_2\}$ defined by $s(a) = d_a$, $s(b) = d_b$, $s(c) = d_c$, $s(x_1) = d_1$ and $s(x_2) = d_2$ is a solution to the instance $(V, C \cup \{a \prec b, b \prec c\})$.*

G_2. *For arbitrary elements $d_a, d_b, d_c \in D$ such that $d_c \prec d_b \prec d_a$ and $d_c \prec d_a$, there exist elements $d_1, d_2 \in D$ such that the function $s\colon V \to \{d_a, d_b, d_c, d_1, d_2\}$ defined by $s(a) = d_c$, $s(b) = d_b$, $s(c) = d_a$, $s(x_1) = d_1$ and $s(x_2) = d_2$ is a solution to the instance $(V, C \cup \{c \prec b, b \prec a\})$.*

G_3. *$(V, C \cup \{b \prec a, b \prec c, a(\prec, \succ)c\})$ is not satisfiable.*

G_4. *$(V, C \cup \{a \prec b, c \prec b, a(\prec, \succ)c\})$ is not satisfiable.*

Proof. Define the gadget $G(a, b, c, x_1, x_2)$ to be the CSP($\mathcal{B}^{\vee=}$) instance

$$(\{a, b, c, x_1, x_2\}, \{x_1 \sqcap a, x_1 \sqcap b, x_1(\prec, \succ)c, x_2(\prec, \succ)a, x_2 \sqcap b, x_2 \sqcap c\}).$$

We demonstrate that G satisfies G_1-G_4. Properties G_1 and G_2 follow immediately from in-fork and out-fork. To prove G_3 and G_4, we need to show that whenever a, b, c is totally ordered in a way different from $a \prec b \prec c$ or $c \prec b \prec a$, then the gadget is not satisfied. Assume for instance that $b \prec a \prec c$. Then, $\{b \prec a, a \prec c, x_2(\prec, \succ)a, x_2 \sqcap b, x_2 \sqcap c\}$ must be satisfiable and this violates property C_4. The remaining three cases can be ruled out analogously. We conclude that G has the properties G_1–G_4. $\square$

Informally, the gadget $G(a, b, c, x_1, x_2)$ constrains b to be between a and c. Equipped with this lemma, we are now ready to prove a weaker version of the main result of the article.

Lemma 7. *Let $\mathcal{B} \in \mathcal{H}$. Then CSP($\mathcal{B}^{\vee=}$) is NP-hard and it is not solvable in subexponential time, unless the ETH is false.*

Proof. Recall that NAE-3-SAT restricted to positive literals may be viewed as CSP($\{N\}$) where $N = \{0,1\}^3 \setminus \{(0,0,0),(1,1,1)\}$, and that this problem is NP-complete. We will show that there exists a polynomial-time many-to-one reduction f from CSP($\{N\}$) to CSP($\mathcal{B}^{\vee=}$).

Arbitrarily choose an instance (A, T) of CSP($\{N\}$) and construct an instance $I = f((A, T))$ of CSP($\mathcal{B}^{\vee=}$) as follows:

F_1: add the variable M to I,
F_2: for each $a \in A$, add a variable a and the constraint $a(\prec,\succ)M$ to I,
F_3: for each triple $N(a,b,c) \in T$, introduce five variables z, x_1, x_2, x_3, x_4 and add $a(\prec,\succ)b$, $a(\prec,\succ)c$, $b(\prec,\succ)c$, $G(a, M, z, x_1, x_2)$, and $G(b, z, c, x_3, x_4)$ to I, where G is the gadget from Lemma 6.

Clearly, the reduction above can be carried out in polynomial time. We proceed with the correctness proof.

First, assume that s is a solution to I. For each $a \in A$, either $s(a) \prec s(M)$ or $s(a) \succ s(M)$. We define a solution $s' \colon A \to \{0,1\}$ such that $s'(a) = 0$ if and only if $s(a) \prec M$. We continue by proving that s' satisfies each $N(a,b,c) \in T$. Assume to the contrary that $s'(a) = s'(b) = s'(c) = 0$, i.e. $s(a), s(b), s(c) \prec s(M)$. We analyse the gadgets $G(a, M, z, x_1, x_2)$, $G(b, z, r, x_3, x_4)$ and the four orderings that they allow.

1. $s(a) \prec s(M) \prec s(z)$ and $s(b) \prec s(z) \prec s(c)$. We see that $s(M) \prec s(z) \prec s(c)$ so $s'(c) = 1$ and this is not possible.
2. $s(a) \succ s(M) \succ s(z)$ and $s(b) \prec s(z) \prec s(r)$. This is not possible since $s(a) \prec s(M)$.
3. $s(a) \prec s(M) \prec s(z)$ and $s(b) \succ s(z) \succ s(r)$. We see that $s(M) \prec s(z) \prec s(b)$ so $s'(b) = 1$ and this is not possible.
4. $s(a) \succ s(M) \succ s(z)$ and $s(b) \succ s(z) \succ s(r)$. This is not possible since we know that $s(a) \prec s(M)$.

The case when $s'(a) = s'(b) = s'(c) = 1$ can be ruled out similarly. We conclude that at least one variable is assigned 0, at least one variable is assigned 1, and the constraint $N(a,b,c)$ is satisfied.

For the other direction, assume that there exists a solution $s' \colon A \to \{0,1\}$ to (A, T). We show how to construct a solution to the instance I. Let $A_0 \subseteq A$ be the variables that are assigned 0 by s' and let $A_1 \subseteq A$ that are assigned 1. Let $(L, \prec)$ denote a strict total order in $(D, \prec)$ that contains $2|A| + 2$ elements

$$d_1 \prec e_1 \prec d_2 \prec e_2 \prec \cdots \prec e_{|A|} \prec d_{|A|+1} \prec e_{|A|+1}.$$

Construct $s \colon A \cup \{M\} \to \{d_1, \ldots, d_{|A|+1}\}$ such that $s(a) \prec s(M)$ if $a \in A_0$ and $s(a) \succ s(M)$ if $a \in A_1$. The function s satisfies all constraints introduced in step F_2. We continue by the constraints introduced in step F_3. Consider an arbitrary constraint $N(a,b,c) \in T$ and the corresponding constraints in I: we have introduced five fresh variables z, x_1, x_2, x_3, x_4 and the constraints: (1) $a(\prec,\succ)b$, (2) $a(\prec,\succ)c$, (3) $b(\prec,\succ)c$, (4) $G(a, M, z, x_1, x_2)$, and (5) $G(b, z, c, x_3, x_4)$. The constraints (1) − (3) are clearly satisfied by s. We will now show how to choose $s(z)$ in order to satisfy constraints (3) and (4). Have in mind that, for instance in constraint (4), it is sufficient to choose $s(z)$ such that $s(a) \prec s(M) \prec s(z)$ or $s(a) \succ s(M) \succ s(z)$; suitable values always exist for x_1 and x_2 due to G_1 and G_2.

Let e_+ be the e-element in $(L, \prec)$ that is the immediate larger neighbour to the element $f(M)$ and define e_- analogously. Given two distinct $a, b \in A$, let e_{ab} be an arbitrary e-element in $(L, \prec)$ that lies between $s(a)$ and $s(b)$. The following table summarises how $s(z)$ should be chosen.

$s'(a)$	$s'(b)$	$s'(c)$	$s(z)$
0	0	1	e_+
0	1	0	e_+
0	1	1	e_{bc}
1	0	0	e_{bc}
1	0	1	e_-
1	1	0	e_-

We conclude that the function s can be extended to a solution to I. □

We need the last stepping stone to prove the stronger version of the lemma above (recall that CSP($\mathcal{B}^{\vee=}$)-3 is the degree-bounded problem where a variable may occur in at most 3 constraints).

Lemma 8. *Let $\mathcal{B}$ be a partition scheme and assume (V, C) is an instance of CSP($\mathcal{B}^{\vee=}$). If $|C| \le c|V|$ for some constant c, then (V, C) can be reduced to an instance of CSP($\mathcal{B}^{\vee=}$)-3 with at most $2c|V|$ variables in polynomial time.*

Proof. First, recall that $\mathrm{eq}_D \in \mathcal{B}$ since $\mathcal{B}$ is a partition scheme. Then pick a variable $x \in V$ occurring in constraints $c_1, \ldots, c_k$ for $k > 3$. Introduce k fresh variables $x_1, \ldots, x_k$ together with the constraints $x_1(\mathrm{eq}_D)x_2$, $x_2(\mathrm{eq}_D)x_3$, …, $x_{k-1}(\mathrm{eq}_D)x_k$. Next, replace each occurrence of x in c_i by the corresponding variable x_i. Clearly, the degree of each x_i variable is at most 3, and the equality constraints enforce that $x_1, \ldots, x_k$ are always assigned the same value in any satisfying assignment. Moreover, each constraint contains two variables, so the total number of variables introduced by this reduction is bounded from above by $2|C| \leq 2c|V|$. □

With this lemma at hand, we are now ready to prove the main result by carefully analysing the reduction in Lemma 7.

Theorem 9. *Let $\mathcal{B} \in \mathcal{H}$. Then CSP($\mathcal{B}^{\vee=}$)-3 is NP-hard and it is not solvable in subexponential time, unless the ETH is false.*

Proof. As before, let $N = \{0,1\}^3 \setminus \{(0,0,0),(1,1,1)\}$. There exists a constant $B \geq 1$ such that CSP($\{N\}$)-B is NP-complete and solvable in subexponential time if and only if the ETH is false [26].

Take an arbitrary instance (A,T) of CSP($\{N\}$)-B with $|A| = n$ (note that $|T| \leq Bn$) and apply the reduction f from Lemma 7 to obtain an instance $(V,C) = f((A,T))$ of CSP($\mathcal{B}^{\vee=}$). Observe that $|C| = |A| + 5|T| \leq (1+5B)n$, since we introduce a constraint $a(\prec,\succ)M$ for each $a \in A$ and five constraints for each $N(a,b,c) \in T$. The term $1+5B$ is constant. Hence, combining f with the reduction from Lemma 7 yields an instance of CSP($\mathcal{B}^{\vee=}$)-3 with $O(n)$ variables in polynomial time. □

Naturally, Theorem 9 also implies that CSP($\mathcal{B}^{\vee=}$)-3 (and, hence, CSP($\mathcal{B}^{\vee=}$)) is NP-complete. For strict partial orders we may combine Theorem 9 with the observation in Theorem 3 to obtain the following corollary.

Corollary 10. *Let $\mathcal{B}$ be a partition scheme with domain D such that $\mathcal{B}^{\vee=}$ contains a strict partial order $\prec$. Assume $\prec$ together with the incomparability relation $\curlywedge$ satisfy C_1–C_3. Then CSP($\mathcal{B}^{\vee=}$)-3 is NP-hard and not solvable in subexponential time, unless the ETH is false.*

In summary, we may rule out subexponential time algorithms for CSP($\mathcal{B}^{\vee=}$)-3 for partition schemes $\mathcal{B} \in \mathcal{H}$. However, the best general algorithm for CSP($\mathcal{B}^{\vee=}$) runs in $O(2^{O(n^2)})$ time (if CSP($\mathcal{B}$) is tractable) [25,39]. 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. We return to this question in Section 5.

4.2. Consequences

The properties in Definition 2 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 an acyclic order such that every strict total order in it contains at most k elements and there is at least one strict total order with three or more elements, then CSP($\mathcal{B}^{\vee=}$) is NP-hard, regardless of whether $\prec$ has properties C_2–C_4 or not. This can be seen via a polynomial-time reduction from k-Colourability (i.e. the problem CSP($\{R_k\}$) where $R_k = \{(x,y) \in \{1,\ldots,k\}^2 \mid x \neq y\}$) to CSP($\mathcal{B}^{\vee=}$). 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 $w(\prec,\succ,\mathrm{eq}_D)c_i$, $1 \leq i \leq k$. Recall that $\succ, \mathrm{eq}_D \in \mathcal{B}$ since $\mathcal{B}$ is a partition scheme so the relation $(\prec,\succ,\mathrm{eq}_D)$ is a member of $\mathcal{B}^{\vee=}$. 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 an acyclic order that contains infinite strict total orders but 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 counterexample, 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 an acyclic order (in fact, it is a strict partial order), it contains infinite strict partial orders (such as $(0,0) \prec (0,1) \prec (0,2) \prec \ldots$), and that it does not have in- or out-forks. Let $\mathcal{B} = \{\prec,\succ,\sqcap,\mathrm{eq}_D\}$ where $\sqcap = D^2 \setminus \bigcup\{\prec,\succ,\mathrm{eq}_D\}$, 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.

4.3. A tractable subclass of degree-bounded problems

Given that CSP($\mathcal{B}^{\vee=}$)-3 is NP-hard whenever $\mathcal{B} \in \mathcal{H}$, it is interesting to see whether the degree bound can be further lowered with retained NP-hardness or not. We will not be able to answer this question in its full generality but for *ω-categorical* partition schemes $\mathcal{B}$, CSP($\mathcal{B}^{\vee=}$)-2 is, to the contrary, always solvable in polynomial time. We begin by recapitulating the concept of ω-categoricity and its connections to qualitative CSPs. A *first-order theory* is a set of first-order sentences and the first-order theory of a constraint language Γ is the set of first-order sentences that are logically entailed by Γ. We say that a satisfiable first-order theory T is ω-categorical if all countable models of Γ are isomorphic, and a constraint language is *ω-categorical* if its first-order theory is ω-categorical. It is known that a large number of qualitative CSPs can be captured via ω-categorical constraint languages: well-known examples include Allen's algebra [21] and RCC-8 [8]. Many more examples can be found in Section 1 of Bodirsky & Jonsson [7].

The basic idea behind our tractability result is to study the tree-width of CSP($\mathcal{B}^{\vee=}$)-2 instances and exploit a result by Bodirsky & Dalmau [6]. A *tree-decomposition* of a graph $G = (V, E)$ is a pair $\mathcal{D} = (\mathcal{T}, f)$, where $\mathcal{T}$ is a rooted tree with vertex set T and $f : T \to 2^V$ is a function such that the following properties hold:

(T_1) $\bigcup_{t \in T} f(t) = V$,
(T_2) for every $u \in V$, the set $\{t \in T \mid u \in f(t)\}$ induces a connected subtree of $\mathcal{T}$, and
(T_3) for each edge $(u, v) \in E$, there exists a $t \in T$ such that $\{u, v\} \subseteq f(t)$.

The *width* of the tree-decomposition $\mathcal{D}$ is $\max_{t \in T} |f(t)| - 1$, and the *treewidth* of G is the minimum width over all tree-decompositions of G.

We will consider the treewidth of *Gaifman graphs*. The Gaifman graph (or the *primal graph*) of a CSP instance (V, C) is the graph on vertex set V where two distinct vertices v_i and v_j are adjacent if and only if v_i and v_j simultaneously appear in the scope of some constraint in C. The following result is a direct consequence of Corollary 1 in Bodirsky & Dalmau [6].

Proposition 11. *Let Γ be a finite ω-categorical constraint language. Then CSP(Γ) restricted to instances whose Gaifman graphs have tree-width bounded by some constant is solvable in polynomial time.*

Theorem 12. *If $\mathcal{B}$ is an ω-categorical partition scheme, then CSP($\mathcal{B}^{\vee=}$)-2 is solvable in polynomial time.*

Proof. If $\mathcal{B}$ is ω-categorical, then $\mathcal{B}^{\vee=}$ is ω-categorical, too, since this property is preserved under first-order definitions [22, Theorem 7.3.8]. Let (V, C) be an arbitrary instance of CSP($\mathcal{B}^{\vee=}$)-2. Since the relations in $\mathcal{B}^{\vee=}$ are binary, it is easy to see that the Gaifman graph of (V, C) is the disjoint union of simple paths and cycles. It is well-known (and not difficult to verify) that such a graph has tree-width at most 2. The result follows from Proposition 11 since $\mathcal{B}^{\vee=}$ is a finite set of relations. $\square$

5. Faster exponential-time algorithms for CSP($\mathcal{B}^{\vee=}$)

We have established that CSP($\mathcal{B}^{\vee=}$) for $\mathcal{B} \in \mathcal{H}$ is unlikely to be solvable in subexponential time, so we focus our efforts on constructing faster exponential-time algorithms. Algorithm 1 (that we present below) is an abstract description of the "classical" backtracking algorithm for solving CSPs over partition schemes. It is not so difficult to see that this algorithm solves CSP($\mathcal{B}^{\vee=}$) in $2^{O(n^2)}$ time (cf. Jonsson & Lagerkvist [25] or Renz & Nebel [38]), and we will soon see that this algorithm applied to the degree-bounded problem runs in $2^{O(n)}$ time. The algorithm is well-known but we give a detailed account of it since it makes the presentation of the degree-bounded case much simpler. We note that the general upper bound can be improved for several specific partition schemes, for instance:

- the CSP over Allen's interval algebra admits a $2^{O(n \log n)}$ time algorithm [25], and
- the CSP over Allen's interval algebra restricted to intervals of unit length admits a $2^{O(n \log \log n)}$ time algorithm [11].

Importantly, as we will prove in this section, the CSP problem over RCC-8 also admits a $2^{O(n \log n)}$ time algorithm. This immediately implies, for instance, that the CSPs for RCC-5 and the partial-order time algebra can be solved within this time bound, too [3].

5.1. The branching algorithm

When presenting the branching algorithm we for simplicity assume that the constraint language $\mathcal{B}$ is a partition scheme where CSP($\mathcal{B}$) is solvable in polynomial time. One can show that an instance $I = (V, C)$ of CSP($\mathcal{B}^{\vee=}$) is satisfiable by providing a *certificate* defined as a satisfiable instance $I' = (V, C')$ of CSP($\mathcal{B}$) obtained by removing all but one relation from each constraint in C. Note that I is satisfiable if and only if it has a certificate: a satisfying assignment to I satisfies I', and vice versa.

Lemma 13. *Let $\mathcal{B}$ be a partition scheme where CSP($\mathcal{B}$) is solvable in polynomial time. Then Algorithm 1 solves an instance $I = (V, C)$ of CSP($\mathcal{B}^{\vee=}$) in $2^{|C| \log(|\mathcal{B}|-1)} \cdot poly(||I||)$ time.*

Algorithm 1 Branching procedure for CSP($\mathcal{B}^{\vee=}$).

```
1: procedure BRANCH(I = (V, C))
2:     if all constraints in C have at most one basic relation then
3:         return SOLVEBASECASE(I)                          // I is an instance of CSP(B)
4:     take any constraint c = x(R_1, ..., R_m)y in C with m > 1
5:     for R_i ∈ {R_1, ..., R_m} do
6:         I' ← I with c replaced by R_i(x, y)
7:         if BRANCH(I') then
8:             return Yes
9:     return No
```

Proof. To confirm that Algorithm 1 is correct, observe that it considers every possible instance of CSP($\mathcal{B}$) that can be obtained from I by removing relations from the constraints. The algorithm returns yes if and only if there is a certificate among these instances.

To determine the running time, note that we may assume without loss of generality that $m < |\mathcal{B}|$ on line 5 of the algorithm (otherwise the constraint c is trivially satisfiable). Then, for each constraint the algorithm considers at most $|\mathcal{B}| - 1$ branches. The size of the recursion tree is at most $(|\mathcal{B}| - 1)^{|C|} = 2^{|C|\log(|\mathcal{B}|-1)}$, while the computation in the base case requires polynomial time in $||I||$. Hence, the running time of the algorithm is bounded by $2^{|C|\log(|\mathcal{B}|-1)} \cdot poly(||I||)$. $\square$

Corollary 14. *CSP($\mathcal{B}^{\vee=}$) is solvable in $2^{O(n^2)}$ time.*

Proof. Since the relations in $\mathcal{B}$ are binary, there are at most $\binom{n}{2}$ constraints in C. The size of the constraint language $\mathcal{B}$ is fixed, so $|C|\log(|\mathcal{B}| - 1) \in O(n^2)$. $\square$

Corollary 15. *CSP($\mathcal{B}^{\vee=}$)-B is solvable in $2^{B\log(|\mathcal{B}|-1)n} \cdot poly(||I||)$ time.*

Proof. Each variable appears in at most B constraints so $|C| \leq Bn$. $\square$

Hence, if $\mathcal{B} \in \mathcal{H}$ then CSP($\mathcal{B}^{\vee=}$)-B is solvable in $2^{O(n)}$ time but not in $2^{o(n)}$ time, unless the ETH fails (by Theorem 9). One interpretation of these results is that backtracking algorithms, which in practice are used in solvers for many temporal and spatial reasoning problems, are close to being optimal in the degree-bounded case.

5.2. A faster algorithm for RCC-8

Let $\mathcal{R}$ denote the set of basic relations in RCC-8 (as defined in Example 2). To convey the intuition behind our algorithm, we start by describing a maximal tractable fragment $\widehat{\mathcal{H}}_8$ of $\mathcal{R}^{\vee=}$ identified in [37]. It contains 108 out of $2^8 = 256$ relations of the full algebra. The missing relations can be classified into six sets:

$$\mathcal{N}_1 = \{R \mid \mathsf{PO} \not\subseteq R, \mathsf{TPP} \subseteq R, \mathsf{TPP}^{-1} \subseteq R\}$$
$$\mathcal{N}_2 = \{R \mid \mathsf{PO} \not\subseteq R, \mathsf{NTPP} \subseteq R, \mathsf{NTPP}^{-1} \subseteq R\}$$
$$\mathcal{N}_3 = \{R \mid \mathsf{PO} \not\subseteq R, \mathsf{TPP} \subseteq R, \mathsf{NTPP}^{-1} \subseteq R\}$$
$$\mathcal{N}_4 = \{R \mid \mathsf{PO} \not\subseteq R, \mathsf{NTPP} \subseteq R, \mathsf{TPP}^{-1} \subseteq R\}$$
$$\mathcal{N}_5 = \{R \mid \mathsf{EQ} \subseteq R, \mathsf{NTPP} \subseteq R, \mathsf{TPP} \not\subseteq R\}$$
$$\mathcal{N}_6 = \{R \mid \mathsf{EQ} \subseteq R, \mathsf{NTPP}^{-1} \subseteq R, \mathsf{TPP}^{-1} \not\subseteq R\}$$

Observe that the constraints that make an instance of CSP($\mathcal{R}^{\vee=}$) hard are those containing both (N)TPP and (N)TPP^{-1} ($\mathcal{N}_1$ through $\mathcal{N}_4$) and those containing EQ together with other relations ($\mathcal{N}_5$ and $\mathcal{N}_6$). Fixing a partition on the variables and restricting the instance accordingly eliminates the latter kind of constraints, while fixing an ordering according to $\mathsf{NTPP} \cup \mathsf{TPP}$ eliminates the former kind. Note that NTPP and TPP are both acyclic orders (in fact, even strict partial orders) and that the relation $\mathsf{NTPP} \cup \mathsf{TPP}$ is crucial for establishing the gadget in Lemma 6. We are now ready to present the improved algorithm for RCC-8.

Theorem 16. *CSP($\mathcal{R}^{\vee=}$) can be solved in $2^{O(n\log n)}$ time.*

Proof. Let $I = (V, C)$ be an instance of the problem. We enumerate surjective functions $\pi : V \to \{1, \ldots, r\}$, where $r \in \mathbb{Z}$, $1 \leq r \leq n$ (also known as *ordered partitions* on V) and restrict each constraint $c \in C$ over $(u, v) \in V^2$ as follows:

(π_1) If $\pi(u) = \pi(v)$, then remove $\mathsf{NTPP}^{\pm 1}$ and $\mathsf{TPP}^{\pm 1}$ from c.

(π_2) If $\pi(u) < \pi(v)$, then remove EQ, TPP^{-1} and NTPP^{-1} from c.
(π_3) If $\pi(u) > \pi(v)$, then remove EQ, TPP and NTPP from c.

Observe that constraints of the restricted instance I_π do not contain relations from $\mathcal{N}_1 \cup \cdots \cup \mathcal{N}_6$. Thus, I_π is in $\widehat{\mathcal{H}}_8$ and we can solve it in polynomial time. By definition, the algorithm is sound, i.e. if it returns yes, then I is satisfiable. To prove completeness, assume that $I' = (V, C')$ is a certificate to I. Define two relations according to C': an equivalence relation

$$\{(u, v) \in V^2 \mid \mathsf{EQ}(u, v) \in C' \text{ or } \mathsf{EQ}(v, u) \in C'\} \tag{1}$$

and an acyclic order

$$\begin{aligned}\{(u, v) \in V^2 \mid & \mathsf{NTPP}(u, v) \in C' \text{ or } \mathsf{NTPP}^{-1}(v, u) \in C' \text{ or} \\ & \mathsf{TPP}(u, v) \in C' \text{ or } \mathsf{TPP}^{-1}(v, u) \in C'\}\end{aligned} \tag{2}$$

Relation (1) partitions V into r equivalence classes for some $r \in \mathbb{Z}$, $1 \le r \le n$. Relation (2) respects (1), so it induces an acyclic ordering on the equivalent classes. Arbitrarily extend it to a linear order and index equivalence classes $V_1, \ldots, V_r$ accordingly. Define $\pi' : V \to \{1, \ldots, r\}$ as $\pi'(v) = i$ if and only if $v \in V_i$ for all $v \in V$. Since the algorithm enumerates ordered partitions, at some point it considers $I_{\pi'}$ and returns yes.

To determine the time complexity, we observe that there are at most n^n ordered partitions of an n-element set. Generating all unordered partitions of a set takes $O(1)$ amortized time per partition [41] and generating all permutations takes $O(1)$ time per permutation [40]. Checking a restricted instance takes polynomial time. Thus, the running time of the algorithm is bounded by $n^n \cdot poly(||I||) \in 2^{O(n \log n)}$. $\square$

This algorithm may be interesting to compare with the algorithm for RCC-8 suggested by Renz and Nebel [38]. Their approach is based on Algorithm 1 but SolveBaseCase is applied to instances of maximal tractable subclasses instead of instances of CSP($\mathcal{R}$). The algorithm presented in Theorem 16 also utilises a maximal tractable subclass but the underlying search strategy is very different. Renz and Nebel's algorithm improves the branching factor when compared to Algorithm 1 and the effect is clearly noticeable in their experimental evaluation. However, the algorithm by Renz and Nebel uses heuristics and it is thus difficult to calculate its running time and compare it to the concrete upper bound obtained in Theorem 16.

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 an acyclic order. We identified the four properties C_1, C_2, C_3, C_4 and provided several real-world examples of acyclic orders satisfying these properties. More importantly, we proved that these properties are sufficient for establishing both NP-hardness and lower bounds under the ETH, even for the degree-bounded problems. At this stage it is worth to yet again point out that none of our results require model-theoretic assumptions such as ω-categoricity.

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 [6,16], 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 [1,4,12], but they typically result in polynomial-time or subexponential algorithms and are therefore unlikely to be usable for CSP($\mathcal{B}^{\vee=}$) problems.

Naturally, there are types of exponential-time algorithms that are not ruled out by our lower bounds, and one of the main open questions of our article is whether it is possible to classify the CSP($\mathcal{B}^{\vee=}$) problems solvable faster than $2^{O(n^2)}$ — similarly to how we were able to obtain an improved algorithm for RCC-8. Could it be the case that all problems of this form are solvable in $2^{O(n \log n)}$ time, or even in single-exponential time? A uniform upper bound of this form would be a major improvement but would not contradict any of our lower bounds. Jonsson and Lagerkvist [25] have presented general results for obtaining algorithms based on enumeration of domain values, which are sometimes much faster than branching algorithms. For example, Allen's interval algebra is solvable in $2^{O(n^2)}$ time with the branching algorithm in Section 5, but the enumeration-based algorithm in Jonsson and Lagerkvist [25] 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 whether it generalises to broader classes of partition schemes.

Another viable approach is to use methods that have been successful in solving finite-domain CSPs. Einarson [15] demonstrates how the finite-domain version of the PPSZ algorithm [20] can be applied to infinite-domain CSPs. His results are inconclusive: the algorithm is faster than previously known algorithms for certain problems but it is, for instance, not competetive for Allen's interval algebra CSP($\mathcal{A}^{\vee=}$). Similarly, there exist several general results on the *kernelizability* of finite domain CSPs [10,24,28], i.e., algorithms for reducing the number of constraints. Could some of these approaches be generalised to infinite-domain CSPs? If yes, then we might be able to obtain improved algorithm simply by combining a kernelization procedure with the branching algorithm presented in Section 5.

These examples also 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($\mathcal{B}^{\vee=}$) which only allows binary relations. Another possibility is to use bounds based on the Chromatic Number problem: Jonsson and Lagerkvist [25, 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?

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. In addition, the first author is partially supported by the Swedish Research Council (VR) under grant 2017-04112, and the second author by VR under grant 2019-03690.

References

[1] J. Alber, H. Fernau, R. Niedermeier, Parameterized complexity: exponential speed-up for planar graph problems, J. Algorithms 52 (1) (2004) 26–56.
[2] J.F. Allen, Maintaining knowledge about temporal intervals, Commun. ACM 26 (11) (1983) 832–843.
[3] F. Anger, D. Mitra, R. Rodríguez, Satisfiability in nonlinear time: algorithms and complexity, in: Proc. 12th International Florida Artificial Intelligence Research Society Conference (FLAIRS-1999), 1999, pp. 406–411.
[4] S. Arora, B. Barak, D. Steurer, Subexponential algorithms for unique games and related problems, J. ACM 62 (5) (2015) 42:1–42:25.
[5] Nikhil Bansal, Avrim Blum, Shuchi Chawla, Correlation clustering, Mach. Learn. 56 (1–3) (2004) 89–113.
[6] M. Bodirsky, V. Dalmau, Datalog and constraint satisfaction with infinite templates, J. Comput. Syst. Sci. 79 (1) (2013) 79–100.
[7] M. Bodirsky, P. Jonsson, A model-theoretic view on qualitative constraint reasoning, J. Artif. Intell. Res. 58 (2017) 339–385.
[8] M. Bodirsky, S. Wölfl, RCC8 is polynomial on networks of bounded treewidth, in: Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI-2011), 2011, pp. 756–761.
[9] A. Bulatov, A dichotomy theorem for nonuniform CSPs, in: Proc. 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS-2017), 2017, pp. 319–330.
[10] H. Chen, B.M.P. Jansen, A. Pieterse, Best-case and worst-case sparsifiability of Boolean CSPs, in: Proc. 13th International Symposium on Parameterized and Exact Computation (IPEC-2018), in: LIPIcs, vol. 115, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, pp. 15:1–15:13.
[11] K. Dabrowski, P. Jonsson, S. Ordyniak, G. Osipov, Fine-grained complexity of temporal problems, in: Proc. 17th International Conference on Principles of Knowledge Representation and Reasoning (KR-2020), 2020, pp. 284–293.
[12] R. de Haan, I.A. Kanj, S. Szeider, On the subexponential-time complexity of CSP, J. Artif. Intell. Res. 52 (2015) 203–234.
[13] I. Düntsch, Relation algebras and their application in temporal and spatial reasoning, Artif. Intell. Rev. 23 (4) (Jun 2005) 315–357.
[14] F. Dylla, J.H. Lee, T. Mossakowski, T. Schneider, A. van Delden, J. van de Ven, D. Wolter, A survey of qualitative spatial and temporal calculi: algebraic and computational properties, ACM Comput. Surv. 50 (1) (2017) 7:1–7:39.
[15] 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.
[16] T. Feder, 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) (1998) 57–104.
[17] M.C. Golumbic, R. Shamir, Complexity and algorithms for reasoning about time: a graph-theoretic approach, J. ACM 40 (5) (1993) 1108–1133.
[18] M. Grigni, D. Papadias, C.H. Papadimitriou, Topological inference, in: Proc. 14th International Joint Conference on Artificial Intelligence (IJCAI-1995), 1995, pp. 901–907.
[19] H. Güsgen, Spatial reasoning based on Allen's temporal logic, Technical report ICSI TR89-049, International Computer Science Institute, 1993.
[20] T. Hertli, I. Hurbain, S. Millius, R.A. Moser, D. Scheder, 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), 2016, pp. 421–437.
[21] R. Hirsch, Relation algebras of intervals, Artif. Intell. 83 (2) (1996) 267–295.
[22] W. Hodges, A Shorter Model Theory, Cambridge University Press, USA, 1997.
[23] R. Impagliazzo, R. Paturi, On the complexity of k-SAT, J. Comput. Syst. Sci. 62 (2) (2001) 367–375.
[24] B.M.P. Jansen, A. Pieterse, Optimal sparsification for some binary csps using low-degree polynomials, ACM Trans. Comput. Theory 11 (4) (2019) 28:1–28:26.
[25] P. Jonsson, V. Lagerkvist, An initial study of time complexity in infinite-domain constraint satisfaction, Artif. Intell. 245 (2017) 115–133.
[26] P. Jonsson, V. Lagerkvist, G. Nordh, B. Zanuttini, Strong partial clones and the time complexity of SAT problems, J. Comput. Syst. Sci. 84 (2017) 52–78.
[27] A. Krokhin, P. Jeavons, P. Jonsson, Reasoning about temporal relations: the tractable subclasses of Allen's interval algebra, J. ACM 50 (5) (2003) 591–640.
[28] V. Lagerkvist, M. Wahlström, Sparsification of SAT and CSP problems via tractable extensions, ACM Trans. Comput. Theory 12 (2) (2020) 13:1–13:29.
[29] G. Ligozat, J. Renz, What is a qualitative calculus? A general framework, in: Proc. 8th Pacific Rim International Conference on Artificial Intelligence (PRICAI-2004), 2004, pp. 53–64.
[30] D. Lokshtanov, D. Marx, S. Saurabh, Lower bounds based on the exponential time hypothesis, Bull. Eur. Assoc. Theor. Comput. Sci. 105 (2011) 41–72.
[31] R. Moratz, J. Renz, D. Wolter, Qualitative spatial reasoning about line segments, in: Proc. 14th European Conference on Artificial Intelligence (ECAI-2000), 2000, pp. 234–238.
[32] A. Mukerjee, G. Joe, A qualitative model for space, in: Proc. 8th National Conference on Artificial Intelligence (AAAI-1990), 1990, pp. 721–727.
[33] I. Pe'er, R. Shamir, Satisfiability problems on intervals and unit intervals, Theor. Comput. Sci. 175 (2) (1997) 349–372.

[34] D. Randell, Z. Cui, A. Cohn, A spatial logic based on regions and connection, in: Proc. 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-1992), 1992, pp. 165–176.
[35] J. Renz, Qualitative spatial and temporal reasoning: efficient algorithms for everyone, in: Proc. 20th International Joint Conference on Artifical Intelligence (IJCAI-2007), Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007, pp. 526–531.
[36] J. Renz, J.J. Li, Automated complexity proofs for qualitative spatial and temporal calculi, in: Proc. Principles of Knowledge Representation and Reasoning (KR-2008), 2008, pp. 715–723.
[37] J. Renz, B. Nebel, On the complexity of qualitative spatial reasoning: a maximal tractable fragment of the region connection calculus, Artif. Intell. 108 (1–2) (1999) 69–123.
[38] J. Renz, B. Nebel, Efficient methods for qualitative spatial reasoning, J. Artif. Intell. Res. 15 (1) (2001) 289–318.
[39] J. Renz, B. Nebel, Qualitative spatial reasoning using constraint calculi, in: Marco Aiello, Ian Pratt-Hartmann, Johan van Benthem (Eds.), Handbook of Spatial Logics, Springer, 2007, pp. 161–215.
[40] R. Sedgewick, Permutation generation methods, ACM Comput. Surv. 9 (2) (1977) 137–164.
[41] I. Semba, An efficient algorithm for generating all k-subsets $(1 \leq k \leq m \leq n)$ of the set $\{1, 2, \ldots, n\}$ in lexicographical order, J. Algorithms 5 (2) (1984) 281–283.
[42] D. Zhuk, A proof of the CSP dichotomy conjecture, J. ACM 67 (5) (2020) 30:1–30:78.