Constants and Finite Unary Relations in Qualitative Constraint Reasoning

Peter Jonsson[a]

[a]*Department of Computer and Information Science, Linköping University, SE-581 83 Linköping, Sweden.*

Abstract

Extending qualitative CSPs with the ability of restricting selected variables to finite sets of possible values has been proposed as an interesting research direction with important applications, cf. "Qualitative constraint satisfaction problems: an extended framework with landmarks" by Li, Liu, and Wang (Artificial Intelligence 201:32-58, 2013). Previously presented complexity results for this kind of extended formalisms have typically focused on concrete examples and not on general principles. We propose three general methods. The first two methods are based on analysing the given CSP from a model-theoretical perspective, while the third method is based on directly analysing the growth of the representation of solutions. We exemplify the methods on temporal and spatial formalisms including Allen's algebra and RCC-5.

Keywords: Constraint satisfaction, qualitative reasoning, computational complexity.

1. Introduction

This introductory section is divided into two parts where we first discuss the background of this article and thereafter describe our results.

1.1. Background

Qualitative reasoning has a long history in artificial intelligence and the combination of qualitative reasoning and constraint reasoning has been a very productive field. A large number of constraint-based formalisms for qualitative reasoning have been invented, most notably within temporal and spatial reasoning, and they have been investigated from many different angles. It has been noted that a particular extension to qualitative CSPs is highly relevant: Cohn and Renz [25, p. 578] observe the following

Email address: peter.jonsson@liu.se (Peter Jonsson)

> One problem with this [constraint-based] approach is that spatial entities are treated as variables which have to be instantiated using values of an infinite domain. How to integrate this with settings where some spatial entities are known or can only be from a small domain is still unknown and is one of the main future challenges of constraint-based spatial reasoning.

and Li, Liu, and Wang [48, p. 33] write

> There is a growing consensus, however, that breakthroughs are necessary to bring spatial/temporal reasoning theory closer to practical applications. One reason might be that the current qualitative reasoning scheme uses a rather restricted constraint language: constraints in a qualitative CSP are always taken from the *same* calculus and only relate variables from the same *infinite* domain. This is highly undesirable, as constraints involving restricted variables and/or multiple aspects of information frequently appear in practical tasks such as urban planning and spatial query processing.

That is, they regard the question of how to extend constraint formalisms with constants and other unary relations[1] as being very important; the same observation has been made in a wider context by Kreutzmann and Wolter [44]. An interesting recent example where such extensions of qualitative formalisms are necessary is the article on spatial query processing by Nikolaou and Koubarakis [56].

Given a (finite or infinite) set of values D, we let $D_c = \{\{d\} \mid d \in D\}$ (i.e. the set of constant relations over D) and $D_f = \{D' \subseteq D \mid D' \text{ is finite}\}$ (i.e. the set of finite unary relations over D). Let us consider finite-domain CSPs for a moment. For every finite constraint language Γ over D, the computational complexity of $\mathrm{CSP}(\Gamma \cup D_f)$ is known due to results by Bulatov [17]. This is an important complexity result in finite-domain constraint satisfaction and it has been reproven several times using different methods [2, 18]. Very recently, the complexity of $\mathrm{CSP}(\Gamma \cup D_c)$ and $\mathrm{CSP}(\Gamma)$ has also been determined [19, 63].

The situation is radically different when considering infinite-domain CSPs where similar powerful results are not known. This can, at least partly, be attributed to the fact that infinite-domain CSPs constitute a much richer class of problems than finite-domain CSPs: for every computational problem X, there is an infinite-domain constraint language Γ_X such that X and $\mathrm{CSP}(\Gamma_X)$ are polynomial-time Turing equivalent [9]. Finite domain CSPs are, on the other hand, always members of NP. Hence, the majority of computational problems cannot be captured by finite-domain CSPs.

Nevertheless, there exist concrete examples where interesting qualitative and/or infinite-domain CSPs have been extended with finite unary relations and/or constant relations. A very early example is the article by Jonsson &

[1] Finite unary relations are sometimes referred to as *landmarks* in the AI literature. We will use the standard mathematical term throughout the article.

Bäckström [38] (see also Koubarakis [43]) where several temporal formalisms (including the point algebra and Allen's algebra) are extended by unary relations (and also other relations). A more recent example is the article by Li et al. [48] where the point algebra and Allen's algebra are once again considered. Li et al. also study several other formalisms including the cardinal relation algebra and RCC-5 and RCC-8 over two-dimensional regions and where constants are assumed to be polygonal regions. The results for the temporal formalisms by Jonsson & Bäckström are not completely comparable with the results by Li et al.: Jonsson & Bäckström's approach is based on linear programming while Li et al. use methods based on enforcing consistency and computational geometry. Consistency-enforcing methods have certain advantages such as lower time complexity and easier integration with existing constraint solving methods. At the same time, the linear programming method allows for more expressive extensions with retained tractability. Both consistency-based and LP-based methods have attracted attention lately, cf. Giannakopoulou et al. [30] and Kreutzmann and Wolter [44], respectively, and generalisations of the basic concepts have been proposed and analysed by de Leng and Heintz [26].

We see that this line of research has to a large extent been based on analysing concrete examples. The approach in this article will be different: instead of studying concrete examples, we study basic principles and aim at providing methods that are applicable to many different constraint formalisms.

1.2. Our results

We present three different methods. The first two methods are based on analysing the given CSP from a model-theoretical perspective. The third method is more of a toolbox for proving that the size of solutions (i.e., the number of bits needed for representing a solution) grows in a controlled way, and that problems consequently are in NP. We will now describe these methods in slightly more detail.

Method I. The first method is based on exploiting *ω-categoricity*. This is a model-theoretical property of constraint languages and other mathematical structures that have gained a lot of attention in the literature. Briefly speaking, a constraint language Γ is ω-categorical if Γ is the unique countable model (up to isomorphism) of the set of all first-order sentences that are true in Γ. One of the interesting aspects of ω-categorical constraint languages is that they in some respects resemble constraint languages over finite domains: this is expressed by a famous result proved by Engeler, Ryll-Nardzewski, and Svenonius (see Theorem 11). From a model-theoretical point of view, ω-categoricity is a very strong assumption. Nevertheless, many interesting CSP problems can be formulated using ω-categorical constraint languages: examples include the point algebra, RCC-5, RCC-8, and Allen's algebra. Among the ω-categorical constraint languages, the *model-complete cores* are particularly interesting. Such constraint languages allow us to define gadgets that can be used for simulating constants. This method is applicable to a wide selection of CSP(Γ) problems when Γ is ω-categorical. The drawback with the method is that it may be difficult to compute the gadgets used for simulating constants. Given that Γ is an ω-categorical

model-complete core and that the gadgets can be computed efficiently, we verify (based on results by Bodirsky [5]) that CSP(Γ) is polynomial-time equivalent to CSP($\Gamma \cup D_c$). To demonstrate the strength of this method, we apply it to an extended version of Allen's algebra. This exercise shows, for example, that relations with higher arity than two does not pose any particular problem. This is an important observation since previous work (such as Li et al. [48]) has mostly focused on binary relations.

Method II. The second method is based on *homogeneity*. This is a property of relational structures that has been studied for a long time in mathematics and logic. Some machinery is needed for the formal definition so we refrain from giving it here. However, we note that homogeneous relational structures have many interesting properties: for instance, they allow quantifier elimination and they are ω-categorical whenever the structure contains a finite number of relation symbols and the domain is countably infinite. Even though homogeneity is a very strong property of relational structures, there are many well-known examples within AI and computer science. An early example was provided by Hirsch [33] who proved that Allen's algebra (with the standard interval-based representation) is homogeneous. Given that Γ is homogeneous and satisfies certain additional restrictions, we show that CSP(Γ) is polynomial-time equivalent to CSP($\Gamma \cup D_c$). The additional restrictions are a bit technical; in brief, Γ needs to be based on a *partition scheme* in the sense of Ligozat & Renz [51]. This method has both advantages and disadvantages when compared to method I. The main advantage is that we do not have to compute the gadgets that are needed for applying method I while the main disadvantage is that we are restricted to a particular kind of constraint languages. Nevertheless, many interesting examples can be found within this restricted class of problems.

Method III. Even though many qualitative CSPs are ω-categorical and method I and/or II may be applicable, this is not always the case. One alternative approach is to analyse the growth of CSP($\Gamma \cup D_c$) solutions, measured as a function of instance size. If the growth is polynomially bounded, then it follows immediately that CSP($\Gamma \cup D_c$) is a member of NP. One disadvantage with this method is that we cannot obtain polynomial-time equivalences between CSP(Γ) and CSP($\Gamma \cup D_c$). Another disadvantage is that it may be quite difficult to obtain such bounds. In certain cases, one can inductively compute bounds by a fairly straightforward analysis of problem instances. We demonstrate this by analysing a variant of RCC-5 that is based on finite sets of integers instead of closed sets in topological spaces.

We illustrate the methods on both temporal and spatial formalisms (including point-based temporal constraints, Allen's algebra, and RCC-5). We want to stress that the representation of domains and relations is very important. In fact, we will see that RCC-5 can be represented in two different ways (which we denote RCC-5_{set} and RCC-$5_{\omega-\mathrm{cat}}$) that give rise to exactly the same computational problem (by Proposition 13). However, we will also see (in Sections 5 and 6.2) that there exists an RCC-5 constraint language $\mathcal{R}$ such that RCC-$5_{\mathrm{set}}(\mathcal{R} \cup \{\{c\}\})$ is NP-complete while RCC-$5_{\omega-\mathrm{cat}}(\mathcal{R} \cup X)$ is polynomial-time

solvable, where $\{c\}$ is a particular constant relation and X is an arbitrary finite set of constants. Thus, adding constant relations to constraint languages that are computationally equivalent (but represented differently) may lead to problems with different computational complexity.

The reader may find it strange that we mostly consider extensions with constant relations. The explanation is the close connection between problems extended with constants and with finite unary relations: if one of them is in NP, then both are in NP (see Lemma 3). Most problems under consideration become NP-hard when adding unary relations containing at least three elements: for example, if the constraint language contains the disequality relation $\neq$, then NP-hardness follows from a straightforward reduction from 3-Colourability. However, NP-completeness is not inevitable if we only add constants to the language—a concrete example of this phenomenon was given earlier in the context of RCC-5. Thus, we can extract more information by considering constants instead of finite unary relations. The same viewpoint is taken by, for instance, Li et al. [48].

This article has the following structure. We introduce the basic concepts from CSPs and logic together with some information about homomorphisms in Section 2. Temporal constraints, Allen's algebra, and RCC-5 will be introduced in Section 3. The three different methods outlined above are presented in Sections 4, 5, and 6, respectively. We conclude the article with a brief discussion in Section 7. This article is a revised and extended version of a conference paper [37].

2. Preliminaries

This section is divided into three parts where we provide basic information concerning constraint satisfaction, logic, and automorphisms of relational structures, respectively.

2.1. Constraint satisfaction problems

We begin by presenting CSPs in terms of *homomorphisms*. This view is widely used in the literature on finite-domain CSP and it will provide us with certain advantages: some of the properties that we consider later on are inherently based on homomorphisms. One should note, however, that there is no fundamental difference with the more common AI viewpoint that constraint satisfaction is about assigning values to variables in a way that satisfy certain constraints. In fact, it will be convenient to use both viewpoints in the sequel.

A *relational signature* τ consists of *relation symbols* R_i with associated *arities* $k_i \in \mathbb{N}$. A *(relational) structure* Γ *over relational signature* τ (also called τ-*structure*) is a set D_Γ (the *domain*) and relation $R_i^\Gamma \subseteq D_\Gamma^{k_i}$ for each relation symbol R_i of arity k_i. If the reference to the structure Γ is clear, we may omit the superscript in R_i^Γ. We sometimes use the shortened notation $\overline{x}$ for a vector $(x_1, \ldots, x_n)$ of any length.

Let Γ and Δ be τ-structures. A *homomorphism* from Γ to Δ is a function f from D_Γ to D_Δ such that for each n-ary relation symbol R in τ and each n-tuple $\overline{a} = (a_1, \ldots, a_n)$, if $\overline{a} \in R^\Gamma$, then $(f(a_1), \ldots, f(a_n)) \in R^\Delta$.

Let Γ be a structure with a relational signature τ. Then, the *constraint satisfaction problem (CSP)* for Γ is the following computational problem.

CSP(Γ)
Instance: A τ-structure Δ.
Question: Is there a homomorphism from Δ to Γ?

Example 1. *For $k \geq 1$, the k-*Colourability *problem is the computational problem of deciding for a given finite graph G whether the vertices can be coloured by k colours such that adjacent vertices get different colours. It is well-known that the k-colouring problem is NP-hard for $k \geq 3$ and tractable when $k \leq 2$. For $k \geq 1$, let K_k denote the complete loop-free graph on k vertices. We view undirected graphs as τ-structures where τ contains a single binary relation symbol E which denotes a symmetric and anti-reflexive relation. Then, the k-*Colourability *problem can be viewed as* CSP($\{K_k\}$).

Example 2. *Consider the problem CSP($(\mathbb{Q}; <)$) where $<$ is the binary order relation of the set $\mathbb{Q}$ of rational numbers. Let $G = (V, A)$ be a directed graph. It is easy to see that there is a homomorphism from G to $(\mathbb{Q}; <)$ if and only if G contains no directed cycle. Thus, CSP($(\mathbb{Q}; <)$) is solvable in polynomial time since cycle detection in directed graphs can be carried out in polynomial time.*

A homomorphism from a given τ-structure Δ to Γ is often called a *solution* of Δ for CSP(Γ). In the homomorphism perspective on CSPs, the structure Γ is typically called the *template* of the constraint satisfaction problem CSP(Γ). The reader should be aware that several different names are used in the literature; *constraint language* is probably the most common within AI. Clearly, we can equivalently define the instances of the CSP(Γ) problem as a tuple (V, C) where V is a set of variables and C is a set of constraints of the form $R(x_{i_1}, \ldots, x_{i_k})$ where R is a relation in Γ, k is the arity of R, and $\{x_{i_1}, \ldots, x_{i_k}\} \subseteq V$. In this case, a solution is a function from V to the domain of Γ satisfying $(f(x_{i_1}), \ldots, f(x_{i_k})) \in R$ for every $R(x_{i_1}, \ldots, x_{i_k}) \in C$.

To represent an input structure Δ of CSP(Γ), we need to fix a suitable representation of the relation symbols in the signature τ. We will see in the forthcoming sections that the choice of representation is very important. Given a particular representation of relation symbols, we let $||\Delta||$ denote the size of an input structure Δ. It is in general not clear how to represent solutions for CSP(Γ) in a compact way, i.e., by an object whose size is polynomially bounded in the input size. When domains are of fixed finite size, then this is indeed possible by simply writing down the variable assignment. When domains are of infinite size, this is not always possible. Note, however, that for the definition of the problem CSP(Γ) we do not need to explicitly represent solutions since we only have to decide the *existence* of solutions.

Let D be a domain with a particular representation and let $||d||$ denote the size of the representation of $d \in D$. We let $D_c = \{\{d\} \mid d \in D\}$ and $D_f = \{D' \subseteq D \mid D' \text{ is finite}\}$. Given a representation of the elements in D, we always represent the members of D_f as sets of elements in D and we may assume that the size of D_f is linear in the sizes of its elements. Other ways of representing D_f are possible (such as solution sets of equations) but they are outside the scope of this article. If Γ is a constraint language with domain D, then $\mathrm{CSP}(\Gamma \cup D_c)$ is the problem $\mathrm{CSP}(\Gamma)$ extended with constants and $\mathrm{CSP}(\Gamma \cup D_f)$ is the problem $\mathrm{CSP}(\Gamma)$ extended with finite unary relations. The next lemma strengthens Proposition 1(iii) in Li et al. [48] by extending it to arbitrary constraint languages.

Lemma 3. *Let Γ denote a constraint language. CSP$(\Gamma \cup D_c)$ is in NP if and only if CSP$(\Gamma \cup D_f)$ is in NP.*

Proof. Assume that $\mathrm{CSP}(\Gamma \cup D_f)$ is in NP. Then, $\mathrm{CSP}(\Gamma \cup D_c)$ is in NP, too, since $D_c \subseteq D_f$. Assume instead that $\mathrm{CSP}(\Gamma \cup D_c)$ is in NP. Let us consider an arbitrary instance $I = (V, C)$ of $\mathrm{CSP}(\Gamma \cup D_f)$. Assume I has a solution $s : V \to D$. We replace each constraint $U(x) \in C$ with $U \in D_f$ by the constraint $\{s(v)\}(v)$. The resulting instance I' is an instance of $\mathrm{CSP}(\Gamma \cup D_c)$, it is satisfiable, and $||I'|| \leq ||I||$. The problem $\mathrm{CSP}(\Gamma \cup D_c)$ is in NP so the satisfiability of I' can be verified in polynomial time by a certificate X. A polynomial-time verifiable certificate for I is thus the tuple (I', X). □

Lemma 3 allows us to, for example, concentrate on $\mathrm{CSP}(\Gamma \cup D_c)$ instead of $\mathrm{CSP}(\Gamma \cup D_f)$ when proving membership in NP.

2.2. Logic

First-order formulas φ over the signature τ (or, in short, τ-formulas) are as usual inductively defined using the logical symbols of universal and existential quantification, disjunction, conjunction, negation, equality, bracketing, variable symbols and the symbols from τ. The semantics of a first-order formula over some τ-structure is defined in the ordinary Tarskian style. A τ-formula without free variables is called a τ-sentence. We write $\Gamma \models \varphi$ if and only if the τ-structure Γ is a model for the τ-sentence φ, that is, satisfies φ. This notation is lifted to sets of sentences (viewed as conjunctions) in the usual way.

2.2.1. Logical definitions

One can use first-order formulas over the signature τ to define relations over a given τ-structure Γ: for a formula $\varphi(x_1, \ldots, x_k)$ where $x_1, \ldots, x_k$ are the free variables of φ, the corresponding relation R is the set of all k-tuples $(t_1, \ldots, t_k) \in D_\Gamma^k$ such that $\varphi(t_1, \ldots, t_k)$ is true in Γ. In this case, we say that R is *first-order definable* in Γ. We extend definability to structures in the natural way: a structure Θ is first-order definable in Γ if every relation in Θ is first-order definable in Γ. Note that our definitions are always parameter-free, i.e., we do not allow the use of domain elements in them. Quantifier-free formulas will be of a certain interest in the following. We say that the τ-structure Γ

admits *quantifier elimination* if every relation with a first-order definition in Γ has a quantifier-free definition in Γ. We also say that a set of formulas T admits quantifier elimination if each $\phi \in T$ has a logically equivalent quantifier-free formula.

We will often consider quantifier-free formulas in *conjunctive normal form* (CNF). Such a formula is a conjunction of *clauses* and a clause is a disjunction of *literals*, i.e., negated or unnegated atomic formulas. A first-order τ-formula $\phi(x_1, \ldots, x_n)$ is called *positive* if it does not contain the negation symbol $\neg$. We note that every quantifier-free positive formula can be rewritten into a logically equivalent positive formula in conjunctive normal form.

A first-order τ-formula $\phi(x_1, \ldots, x_n)$ is called *existential* if it is of the form

$$\exists x_{n+1}, \ldots, x_m.\psi$$

where ψ is a quantifier-free first-order formula. A subset of positive and existential formulas is of particular interest to us: a first-order τ-formula $\phi(x_1, \ldots, x_n)$ is called *primitive positive* if it is of the form

$$\exists x_{n+1}, \ldots, x_m.\psi_1 \wedge \cdots \wedge \psi_l$$

where $\psi_1, \ldots, \psi_l$ are *atomic τ-formulas*, i.e., formulas of the form

1. $R(y_1, \ldots, y_k)$ with $R \in \tau$ and $y_i \in \{x_1, \ldots, x_m\}$ or
2. $y = y'$ for $y, y' \in \{x_1, \ldots, x_m\}$.

Primitive positive formulas will be called pp-formulas for short. In a pp-formula, equality relations can always be removed by identifying variables. This is not true for general formulas since we may, for instance, have formulas such as $\neg(x = y)$. If the relation R has a primitive positive definition in Γ, then we say that R is *pp-definable* in Γ, and we define $\langle\Gamma\rangle$ to be the set of relations that are pp-definable in Γ. Jeavons [36] has proved the following result.

Theorem 4. *If Γ is a structure and the relation R is pp-definable in Γ, then there is a polynomial-time reduction from CSP($\Gamma \cup \{R\}$) to CSP(Γ).*

This explains why pp-definability is important when studying the complexity of CSP problems. To exemplify pp-definitions and Theorem 4, consider the structure $\Gamma = \{\mathbb{N}; \{1, 2, 3, 4\}, \neq\}$. We see that the binary relation $K_4 = \{(x, y) \in \{1, 2, 3, 4\}^2 \mid x \neq y\}$ (from Example 1) is pp-definable in Γ since

$$K_4(x, y) \Leftrightarrow \{1, 2, 3, 4\}(x) \wedge \{1, 2, 3, 4\}(y) \wedge x \neq y.$$

and it follows that CSP(Γ) is NP-hard. It is worth mentioning that many of the operations that are encountered in relation algebra can be viewed as pp-definitions. Let R and S denote binary relations. Then, the converse $R^\smile$ has the pp-definition $R^\smile(x, y) \Leftrightarrow R(y, x)$, the intersection $R \cap S$ has the pp-definition $(R \cap S)(x, y) \Leftrightarrow R(x, y) \wedge S(x, y)$, and the composition $R \circ S$ has the pp-definition $(R \circ S)(x, y) \Leftrightarrow \exists z.R(x, z) \wedge S(z, y)$.

We finally discuss certain families of binary relations. *Partition schemes* were introduced by Ligozat & Renz [51] and they have been highly influential in CSP

research. Let D be a non-empty domain. Given a finite family $\mathcal{B} = \{B_1, \ldots, B_k\}$ of binary relations over D, we say that $\mathcal{B}$ is *jointly exhaustive* (JE) if $\bigcup \mathcal{B} = D^2$ and that $\mathcal{B}$ is *pairwise disjoint* (PD) if $B_i \cap B_j = \varnothing$ whenever $1 \leq i \neq j \leq k$. If $\mathcal{B}$ is simultaneously JE and PD (which we denote JEPD), then $\mathcal{B}$ forms a partition of the set D^2.

Definition 5. *Let D be a non-empty domain and let $\mathcal{B} = \{B_1, \ldots, B_k\}$ be a finite set of binary relations over D. We say that $\mathcal{B}$ is a* partition scheme *if the following holds:*

1. *$\mathcal{B}$ is JEPD,*
2. *the equality relation $\mathtt{EQ}_D = \{(x, x) \in D^2\}$ is in $\mathcal{B}$, and*
3. *for every $B_i \in \mathcal{B}$, the converse relation $B_i^\smile$ is in $\mathcal{B}$.*

It is important to note that if $\mathcal{B}$ is a partition scheme over a domain D, then for arbitrary $d, d' \in D$ there exists exactly one $B \in \mathcal{B}$ such that $(d, d') \in B$. Given a finite set of binary relations $\mathcal{B} = \{R_1, \ldots, R_k\}$, we follow notational conventions from [24, 40] and define $\mathcal{B}^{\vee=}$ as 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 $\mathrm{CSP}(\Gamma)$ where $\Gamma \subseteq \mathcal{B}^{\vee=}$ are the natural objects that are studied in connection with partition schemes. If $\mathcal{B}$ is a partition scheme, then every relation in $\mathcal{B}^{\vee=}$ is quantifier-free positive definable in $\mathcal{B}$ but not necessarily pp-definable in $\mathcal{B}$ (since disjunctions are not allowed in pp-definitions).

2.2.2. Membership in NP

When a constraint language Γ is logically defined in some constraint language Θ, then Γ sometimes inherits useful properties of Θ. We have already encountered Theorem 4 which is one example of this. We will next present two results that connect logical definability with membership in NP. These results will be helpful in the forthcoming Sections 5 and 6.

Lemma 6. *Let $\mathcal{B}$ denote a partition scheme over domain D, and let Γ denote a finite constraint language that is quantifier-free definable in $\mathcal{B}$. If CSP($\mathcal{B} \cup D_c$) is in NP, then CSP($\Gamma \cup D_c$) is in NP.*

Proof. We assume that $\mathcal{B} = \{B_1, \ldots, B_k\}$, $\Gamma = \{R_1, \ldots, R_m\}$, and that every relation R_i, $1 \leq i \leq m$, is defined by a quantifier-free formula ϕ_i. Without loss of generality, we assume that ϕ_i is written in conjunctive normal form. We first demonstrate that every clause in ϕ_i can be rewritten as a disjunction of relations in $\mathcal{B}$. If ϕ_i does not contain any negative literals, then ϕ_i itself has this property. Otherwise, consider a negation such as $\neg(B_i(x, y))$. Since $\mathcal{B}$ is a partition scheme, we have the following equivalence:

$$\neg(B_i(x, y)) \Leftrightarrow \bigvee_{B \in \mathcal{B} \setminus \{B_i\}} B(x, y).$$

We can thus assume, without loss of generality, that ϕ_i does not contain any negated literals.

Let $I = (V, C)$ denote an arbitrary instance of $\mathrm{CSP}(\Gamma \cup D_c)$ with the solution $s : V \to D$. Pick an arbitrary constraint $R_i(x_1, \ldots, x_n) \in C$ with $R_i \in \Gamma$. We consider the defining formula $\phi_i = \sigma_1 \wedge \cdots \wedge \sigma_p$. From each clause σ_i, $1 \leq i \leq p$, choose one literal that is satisfied by solution s. Add these literals to the set C'. Repeat this process for all constraints in C. Finally, add all constraints $U(x) \in C$ with $U \in D_c$ to C'. Note the following.

1. (V, C') is an instance of $\mathrm{CSP}(\mathcal{B} \cup D_c)$,
2. the size of (V, C') is polynomially bounded in the size of (V, C),
3. (V, C') is satisfiable (as is witnessed by the function s), and
4. there exists a polynomial-time verifiable certificate for the satisfiability of (V, C') (since $\mathrm{CSP}(\mathcal{B} \cup D_c)$ is in NP).

Hence, there is a polynomial-time verifiable certificate for the satisfiability of (V, C): concatenate the instance (V, C') with a polynomial-time verifiable certificate for the satisfiability of (V, C'). This implies that $\mathrm{CSP}(\Gamma \cup D_c)$ is in NP. □

Lemma 7. *Let Θ denote a constraint language over domain D, and let Γ denote a finite constraint language that is quantifier-free positive definable in Θ. If $\mathrm{CSP}(\Theta \cup D_c)$ is in NP, then $\mathrm{CSP}(\Gamma \cup D_c)$ is in NP.*

Proof. The proof is virtually the same as the proof of Lemma 6. We merely observe the following: when a quantifier-free positive formula is converted into conjunctive normal form, the resulting formula is still positive, i.e., it contains no negation symbols. Thus, we will not need to rewrite negated literals as disjunctions and it is not required that Θ is a partition scheme. □

Results like Lemma 6 and 7 may give the impression that there is a strong connection between the complexity of $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Gamma \cup D_c)$. Unfortunately, this is not true in general. To see this, we begin by arbitrarily choosing a complexity class $\mathcal{C}$ that admits complete problems under polynomial-time reductions (such as PSPACE or EXPTIME). Let $\Gamma = \{R_1, R_2, \ldots\}$ be a constraint language over domain $\mathbb{N}$ such that $\mathrm{CSP}(\Gamma)$ is $\mathcal{C}$-complete under polynomial-time reductions. Results by Bodirsky and Grohe [9, Theorem 1] provide a systematic way of obtaining such Γ. However, one should note that it is often possible (and sometimes even simpler) to obtain such a Γ via ad hoc constructions.

Let 0_k denote the all-zero vector with k elements. Given a relation $R \in \Gamma$ with arity k, we define

$$R' = \{(a_1 + 1, \ldots, a_k + 1) \mid (a_1, \ldots, a_k) \in R\} \cup \{0_k\}.$$

and let $\Gamma' = \{R' \mid R \in \Gamma\} \cup \{S'\}$ where

$$S' = \{(1, n) \in \mathbb{N}^2 \mid n \geq 1\} \cup \{0_2\}.$$

The problem $\mathrm{CSP}(\Gamma')$ is trivially in P since every instance (V, C) is satisfied by assigning the value 0 to each variable. We will next show that the problem $\mathrm{CSP}(\Gamma' \cup \mathbb{N}_c)$ is, on the other hand, $\mathcal{C}$-hard. We can thus find examples of Γ' such that $\mathrm{CSP}(\Gamma')$ and $\mathrm{CSP}(\Gamma' \cup \mathbb{N}_c)$ are arbitrarily far separated.

To verify that $\mathrm{CSP}(\Gamma' \cup \mathbb{N}_c)$ is $\mathcal{C}$-hard, we present a polynomial-time reduction from $\mathrm{CSP}(\Gamma)$ to $\mathrm{CSP}(\Gamma' \cup \{\{1\}\})$. Let (V, C) denote an arbitrary instance of $\mathrm{CSP}(\Gamma)$. Introduce one fresh variable y and define (V', C') such that $V' = V \cup \{y\}$ and

$$C' = \{R'_i(\bar{x}) \mid R_i(\bar{x}) \in C\} \cup \{S'(y, x) \mid x \in V\} \cup \{\{1\}(y)\}.$$

If (V, C) has a solution $f : V \to \mathbb{N}$, then it is obvious that the function $f' : V' \to \mathbb{N}$ defined by $f'(y) = 1$ and $f'(x) = f(x) + 1$, $x \in V$, is a solution to (V', C').

If (V', C') has a solution $f' : V' \to \mathbb{N}$, then we note that no variable can be assigned the value 0. Thus, it is easy to verify that the function $f : V \to \mathbb{N}$ defined by $f(x) = f'(x) - 1$ is a solution to (V, C).

2.3. Automorphisms

Keeping the homomorphism definition of CSPs in mind may be helpful in the rest of this section. Let Γ and Δ denote two relational τ-structures over domain D. We say that a function $f : D \to D$ *preserves* $R \in \Gamma$ if for every $(a_1, \ldots, a_k) \in R$, the tuple $(f(a_1), \ldots, f(a_k))$ is in R, too. A bijective homomorphism f from Γ to Δ is called an *isomorphism* if the inverse of f is a homomorphism from Δ to Γ. If Γ and Δ are isomorphic, then it is clear that $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Delta)$ are the same computational problem. An injective homomorphism that additionally preserves the complement of each relation is called an *embedding*; the complement of a k-ary relation R in Γ is the relation $D^k \setminus R$. Homomorphisms from Γ to Γ are called *endomorphisms* of Γ. An *automorphism* of Γ is a bijective endomorphism whose inverse is also an endomorphism. In other words, automorphisms are bijective embeddings of Γ into Γ or isomorphisms from Γ to Γ. The set containing all endomorphisms of Γ is denoted $\mathrm{End}(\Gamma)$ while the set of all automorphisms is denoted $\mathrm{Aut}(\Gamma)$.

Example 8. *Let* $R_+ = \{(x, y, z) \in \mathbb{Z}^3 \mid x + y = z\}$. *For arbitrary* $a \in \mathbb{Z}$, *let* $e_a : \mathbb{Z} \to \mathbb{Z}$ *be defined as* $e_a(n) = a \cdot n$. *Let* $e : \mathbb{Z} \to \mathbb{Z}$ *be an arbitrary endomorphism of* $(\mathbb{Z}; R_+)$; *e is a homomorphism so* $(e(x), e(y), e(z)) \in R_+$ *whenever* $(x, y, z) \in R_+$ *and, more generally,* $e(\sum_{i=1}^{k} x_i) = \sum_{i=1}^{k} e(x_i)$ *when* $x_1, \ldots, x_k \in \mathbb{Z}$. *Arbitrarily choose* $n \in \mathbb{Z}$ *and note that*

$$e(n) = e(\underbrace{1 + \cdots + 1}_{n \text{ times}}) = n \cdot e(1).$$

It follows that $\mathrm{End}((\mathbb{Z}; R_+)) = \{e_a \mid a \in \mathbb{Z}\}$. *Note that* e_a *has an inverse if and only if* $a \in \{-1, 1\}$. *Thus,* $\mathrm{Aut}((\mathbb{Z}; R_+)) = \{e_a \mid a \in \{-1, 1\}\}$.

A useful observation is that if (V, C) is an instance of CSP(Γ) with a solution $s : V \to D$, then $s' : V \to D$ defined by $s'(x) = \alpha(s(x))$ is a solution to (V, C) for every α in Aut(Γ) or End(Γ). If a function $s : V \to D$ is *not* a solution to (V, C), then $s'(x) = \alpha(s(x))$ is not a solution for any $\alpha \in \mathrm{Aut}(\Gamma)$ while s' may or may not be a solution if $\alpha \in \mathrm{End}(\Gamma) \setminus \mathrm{Aut}(\Gamma)$.

In the following, let G be a set of permutations of a set X. We say that G is a *permutation group* if the identity permutation is in G and for arbitrary $g, f \in G$, the functions $x \mapsto g(f(x))$ and $x \mapsto g^{-1}(x)$ are also in G. In other words, G is closed under function composition and inversion. If Γ is a τ-structure, then Aut(Γ) is a permutation group on the set D_Γ. For $n \geq 1$, the *orbit* of an n-tuple $(t_1, \ldots, t_n) \in X^n$ under G is the set $\{(\alpha(t_1), \ldots, \alpha(t_n)) \mid \alpha \in G\}$. Clearly, the orbits of n-tuples under G partition the set X^n, that is, every $(t_1, \ldots, t_n) \in X^n$ lies in precisely one orbit under G.

Example 9. *Consider once again the structure $(\mathbb{Z}; R_+)$ from Example 8. It is obvious that $\{e_1, e_{-1}\}$ forms a group under function composition. If $a \in \mathbb{Z}$, then the orbit of (a) equals $\{a, -a\}$ so $(\mathbb{Z}; R_+)$ admits an infinite number of different orbits under its automorphism group.*

We will now introduce the central concept of *ω-categoricity*. It has been observed that ω-categoricity plays an important role in the context of qualitative reasoning. We will not go into the details here but the interested reader may refer to, for instance, Huang [35], Westphal et al. [62], Bodirsky & Dalmau [8], and Bodirsky & Jonsson [11].

A first-order *theory* is a set of first-order sentences. When the first-order sentences are over the signature τ, we say that T is a *τ-theory*. The *(full) theory* of a τ-structure Δ (denoted Th(Δ)) is the set of τ-sentences ϕ such that $\Delta \models \phi$. A *model* of a τ-theory T is a τ-structure Δ such that Δ satisfies all sentences in T. Theories that have a model are called *satisfiable*. Since models are structures, the notion of isomorphism for structures immediately carries over to models. A satisfiable first-order theory T is *ω-categorical* if all countable models of T are isomorphic, and a structure is ω-categorical if its first-order theory is ω-categorical. All ω-categorical structures that appear in this article will be countably infinite so we make the convention that ω-categorical structures are countably infinite. Note that the first-order theory of a finite structure does not have infinite models so finite structures are ω-categorical. One of the first infinite structures that was found to be ω-categorical (by Cantor [22]) is the linear order of the rational numbers $(\mathbb{Q}; <)$. There are many characterisations of ω-categoricity and the most important one is in terms of the automorphism group.

Definition 10. *A permutation group G over a countably infinite set X is* oligomorphic *if G has only finitely many orbits of n-tuples for each $n \geq 1$.*

An accessible proof of the following theorem can be found in Hodges' book [34].

Theorem 11 (Engeler, Ryll-Nardzewski, Svenonius)**.** *Let Γ be a countably infinite structure with countable signature. Then, Γ is ω-categorical if and only if* Aut(Γ) *is oligomorphic.*

Example 9 immediately implies that $(\mathbb{Z}; R_+)$ is not an ω-categorical structure. Consider the structure $(\mathbb{Z}; <)$. One can verify that $\mathrm{Aut}((\mathbb{Z}; <)) = \{x \mapsto x + a \mid a \in \mathbb{Z}\}$. Hence, $(\mathbb{Z}; <)$ is not ω-categorical (despite the fact that $(\mathbb{Q}; <)$ is indeed ω-categorical): the orbits of $(0, 0)$, $(0, 1)$, $(0, 2)$, ... are distinct.

We conclude this section by presenting a result that connects first-order definability with ω-categoricity.

Theorem 12 (Thm. 7.3.8 in Hodges [34])**.** *If Γ is an ω-categorical structure and Δ is first-order definable in Γ, then Δ is ω-categorical, too.*

3. CSP Examples

We will now give a brief introduction to three qualitiative formalisms: *temporal constraints* (including the *point algebra*), *Allen's algebra* and *RCC-5*. These formalisms will be used as illustrating examples during the course of the article.

We begin with the structure $(\mathbb{Q}; <)$, that is, the rational numbers with the usual linear ordering relation. Constraint languages that are first-order definable in $(\mathbb{Q}; <)$ are well-studied in the literature and they are sometimes called *temporal constraint languages*. We have already noticed that $(\mathbb{Q}; <)$ is ω-categorical which implies that all temporal constraint languages are ω-categorical by Theorem 12. Furthermore, the computational complexity of $\mathrm{CSP}(\Gamma)$ is known for every finite temporal constraint language Γ [13]. In particular, $\mathrm{CSP}(\Gamma)$ is either in P or it is an NP-complete problem. The *point algebra* (PA) is the constraint language $\{<, =, >\}^{\vee=}$ and CSP(PA) is probably the most well-known example of a polynomial-time solvable temporal constraint language.

We continue by introducing Allen's algebra [1]. It was introduced to reason about intervals and the qualitative relationships between intervals. The variable domain is

$$\mathbb{I} = \{\{x \in \mathbb{Q} \mid a \leq x \leq b\} \mid a, b \in \mathbb{Q} \text{ and } a < b\},$$

that is, it consists of all closed intervals $[a, b]$ of rational numbers. If $I = [a, b] \in \mathbb{I}$, then we write I^- for a and I^+ for b. The basic relations are the 13 relations defined in Figure 1. We let $\mathcal{A}$ denote the set of Allen basic relations. Clearly, $\mathcal{A}$ is a partition scheme and the 8192 relations of Allen's algebra are the contents of the set $\mathcal{A}^{\vee=}$. The structure $\mathcal{A}^{\vee=}$ is known to be ω-categorical [32]. Observe that $\mathcal{A}^{\vee=}$ is not a temporal constraint language since the domain is not $\mathbb{Q}$.

For every subset $\Gamma \subseteq \mathcal{A}^{\vee=}$, the complexity of $\mathrm{CSP}(\Gamma)$ is known [45]. Unlike temporal constraint languages, the complexity for all finite first-order definable constraint languages are not known. We will encounter the *Ord-Horn* subclass [55] $\mathcal{H} \subseteq \mathcal{A}^{\vee=}$ later on. A relation $R \in \mathcal{A}^{\vee=}$ is a member of $\mathcal{H}$ if and only if the following hold: $([I^-, I^+], [J^-, J^+]) \in R$ if and only if $\phi(I^-, I^+, J^-, J^+)$ evaluates to true, where ϕ is a positive CNF formula $\phi(x^-, x^+, y^-, y^+)$ over $(\mathbb{Q}; \leq, =, \neq)$ such that each clause contains at most one relation of the form

Basic relation		Illustration	Endpoints
I precedes J	p	III	$I^+ < J^-$
J preceded by I	$\mathsf{p}^\smile$	JJJ	
I meets J	m	IIII	$I^+ = J^-$
J met by I	$\mathsf{m}^\smile$	JJJJ	
I overlaps J	o	IIII	$I^- < J^- < I^+$,
J overlapped by I	$\mathsf{o}^\smile$	JJJJ	$I^+ < J^+$
I during J	d	II	$I^- > J^-$,
J includes I	$\mathsf{d}^\smile$	JJJJJJ	$I^+ < J^+$
I starts J	s	III	$I^- = J^-$,
J started by I	$\mathsf{s}^\smile$	JJJJJJ	$I^+ < J^+$
I finishes J	f	III	$I^+ = J^+$,
J finished by I	$\mathsf{f}^\smile$	JJJJJJ	$I^- > J^-$
I equals J	$\equiv$	IIII	$I^- = J^-$,
		JJJJ	$I^+ = J^+$

Figure 1: The definitions of the basic relations of Allen's algebra.

$$
\begin{array}{lll}
\mathtt{PP}(x,y) & \text{iff} & x \subset \mathrm{int}(y) \\
\mathtt{PP}^\smile(x,y) & \text{iff} & \mathrm{int}(x) \supset y \\
\mathtt{DR}(x,y) & \text{iff} & \mathrm{int}(x) \cap \mathrm{int}(y) = \varnothing \\
\mathtt{PO}(x,y) & \text{iff} & \mathrm{int}(x) \cap \mathrm{int}(y) \neq \varnothing, x \not\subseteq y, y \not\subseteq x \\
\mathtt{EQ}(x,y) & \text{iff} & x = y
\end{array}
$$

Figure 2: Basic relations of RCC-5_{reg} where $\mathrm{int}(\cdot)$ denotes the interior operator.

$x = y$ or $x \leq y$. The constraint language $\mathcal{H}$ is very well-studied within the literature on temporal constraints. We merely note that $\mathrm{CSP}(\mathcal{H})$ is polynomial-time solvable, $\mathrm{CSP}(\mathcal{H} \cup \{R\})$ is NP-hard whenever $R \in \mathcal{A}^{\vee=} \setminus \mathcal{H}$, and that $\mathcal{A} \subseteq \mathcal{H}$

We finally turn our attention to RCC-5. The RCC formalisms [58] are designed for reasoning about spatial regions and they are the basis for a large part of the work in *qualitative spatial reasoning* (QSR). There are several variants such as RCC-23, RCC-8, and RCC-5. We will heneceforth concentrate on the simplest formalism RCC-5. RCC-5 is based on five basic relations PP, $\mathtt{PP}^\smile$ (which is the inverse of PP), PO, DR, and EQ, which together form a partition scheme. Here, PP stands for *proper part*, PO stands for *partial overlap*, DR stands for *disconnected regions*, and EQ stands for *equality*. We will consider several different variants of RCC-5 based on different choices of variable domains.

The first variant is based on the standard representation of the spatial calculus RCC-8 (the reader is referred to Renz and Nebel [60, Sec. 3.1] for details concerning RCC-8) but where one is not able to distinguish regions from their topological closure: the disconnectedness relations DC and EC are replaced by DR and the tangential and non-tangential proper part relations TPP and NTPP are replaced by PP (and the relation $\mathtt{PP}^\smile$ is introduced analogously). Here, the

$$
\begin{array}{lll}
\texttt{PP}(x,y) & \text{iff} & x \subset y \\
\texttt{PP}^{\smile}(x,y) & \text{iff} & x \supset y \\
\texttt{DR}(x,y) & \text{iff} & x \cap y = \varnothing \\
\texttt{PO}(x,y) & \text{iff} & \exists a,b,c : a \in x, a \notin y, b \in x, b \in y, c \notin x, c \in y \\
\texttt{EQ}(x,y) & \text{iff} & x = y
\end{array}
$$

Figure 3: Basic relations of RCC-5_{set}.

domain consists of the nonempty regular closed subsets of some regular and connected topological space. A subset of a topological space is called *regular closed* if it is equal to the closure of its interior. Note that the sets are *not* required to be connected. The basic relations of RCC-5 under this choice of variable domain is given in Figure 2. We henceforth call this algebra RCC-5_{reg}.

We continue by providing another variant of RCC-5. Here, we consider arbitrary non-empty subsets of an infinite set such as $\mathbb{N}$. We define the relations $\texttt{PP}, \texttt{PP}^{\smile}, \texttt{DR}, \texttt{PO}, \texttt{EQ}$ as in Figure 3 and denote the resulting structure by RCC-5_{set}. RCC-5_{set} is not ω-categorical while it is unknown whether there is a topological space (with the properties described above) where one can define RCC-5_{reg} such that the resulting structure is ω-categorical. However, there are ways of defining RCC-5 such that the resulting structure is ω-categorical described in the literature. For instance, Bodirsky and Chen [7] presents such a structure based on Fraïssé amalgamation. We let RCC-$5_{\omega-\text{cat}}$ denote this constructions.

The following result is a consequence of Proposition 15 in Bodirsky & Jonsson [11] combined with Section 6 in Bodirsky and Chen [7].

Proposition 13. *Let (V, C) be an instance of RCC-5. The following three statements are equivalent.*

1. *(V, C) is satisfiable if the relations are interpreted over RCC-5_{reg}.*
2. *(V, C) is satisfiable if the relations are interpreted over RCC-5_{set}.*
3. *(V, C) is satisfiable if the relations are interpreted over RCC-$5_{\omega-\text{cat}}$.*

In other words, the RCC-5 representations RCC-5_{reg}, RCC-5_{set}, and RCC-$5_{\omega-\text{cat}}$ are indistinguishable from a computational perspective. The RCC-5 representation used by Li et al. [48] is restricted to regions in the plane. The exact computational properties of this representation are not known, but it is worth noting that Proposition 13 cannot be extended to RCC-5 based on non-empty open *disks* in the plane [11, Sec. 2.5.2]. Further discussions concerning different interpretations of RCC-5 and other spatial formalisms can be found in [7, 27, 49].

4. Method I: ω-categoricity and model-complete cores

Our first method is based on analysing a given constraint language Γ with respect to its automorphisms and the central notions here are ω-categoricity and model-complete cores. We present the method in Section 4.1 and, as an example, apply it to an extended version of Allen's algebra in Section 4.2.

4.1. Constants and model-complete cores

We first introduce the concept of *homomorphically equivalent CSPs*. Let Γ and Δ denote two relational τ-structures. If Γ and Δ are isomorphic, then it is clear that CSP(Γ) and CSP(Δ) correspond to the same computational problem. However, Γ and Δ may be non-isomorphic and still correspond to the same computational problem. Assume that $\Gamma = (D; R_1, R_2, \dots)$ and $\Delta = (D'; R'_1, R'_2, \dots)$. Given an instance (V, C) of CSP(Γ), let (V, C') denote the instance of CSP(Δ) where each relation R_i appearing in C has been replaced by R'_i. We say that Γ and Δ *have the same CSP* if the following holds for all instances (V, C) of CSP(Γ): (V, C) is satisfiable if and only if (V, C') is satisfiable. This is, for example, the case when there simultaneously exists a homomorphism from Γ to Δ and a homomorphism from Δ to Γ. In this case, we say that Γ and Δ are *homomorphically equivalent* and this defines an equivalence relation on structures. We note that there are structures that have the same CSP even when they are not homomorphically equivalent. Consider for example the structures $(\mathbb{Z}; <)$ and $(\mathbb{Q}; <)$. They have the same CSP and there is a homomorphism from $(\mathbb{Z}; <)$ to $(\mathbb{Q}; <)$ but there is no homomorphism from $(\mathbb{Q}; <)$ to $(\mathbb{Z}; <)$.

For ω-categorical structures Γ, the equivalence classes have interesting properties: the homomorphic equivalence class of Γ contains a distinguished member Δ which is up to isomorphism uniquely given by two properties: Δ is a *core* and Δ is *model-complete*. A relational structure Γ is a core if all endomorphisms of Γ are embeddings. Cores are important when studying the complexity of finite-domain CSPs: we refer to the textbook by Hell and Nešetřil [31] that extensively covers cores in the context of graph homomorphisms and to Bulatov et al. [20] that covers cores in general finite-domain CSPs. It is known that when the domain of a relational structure is infinite, then there are several reasonable ways of defining cores; see, for instance, Bauslaugh [4] or Bodirsky [6, Sec. 3.6.3]. The reason for choosing the definition above is simple: it is the definition used by Bodirsky [5] in his proof of the forthcoming Theorem 14.

Model completeness is a central concept in model theory. Let T be a first-order theory. We say that the formulas ϕ and ψ are *equivalent modulo* T if $T \models (\phi \Leftrightarrow \psi)$. A structure Γ is model-complete if every formula in Th(Γ) is equivalent to an existential formula modulo T. This may be viewed as a limited notion of quantifier elimination.

Consider the relation $<$ over the rationals $\mathbb{Q}$. The structure $(\mathbb{Q}; <)$ admits quantifier elimination [46] so every formula in Th($(\mathbb{Q}; <)$) is equivalent to a quantifier-free formula (and, naturally, an existential formula). It follows that $(\mathbb{Q}; <)$ is model-complete, and that every Γ that is first-order definable in $(\mathbb{Q}; <)$ is model-complete, too. The structure $(\mathbb{Q}; <)$ is also a core. Let $e : \mathbb{Q} \to \mathbb{Q}$ be an endomorphism of $(\mathbb{Q}; <)$, i.e., if $a < b$, then $e(a) < e(b)$. Clearly, e is injective and it preserves the relation $\geq$ (that is, the negation of $<$) since if $a > b$, then $e(a) > e(b)$ and if $a = b$, then $e(a) = e(b)$. However, there are relations R that are first-order definable in $(\mathbb{Q}; <)$ and $(\mathbb{Q}; R)$ is not a core. One trivial example is the equality relation $=$. The function $x \mapsto 1$ is obviously an endomorphism

of $=$ but it is not injective and thus not an embedding. We have the following important result.

Theorem 14 (Theorem 16 in Bodirsky [5])**.** *Every ω-categorical structure Δ is homomorphically equivalent to a model-complete core structure Γ which is unique up to isomorphism. Moreover, Γ is ω-categorical and the orbits of n-tuples over D_Γ are pp-definable in Γ for all $n \geq 1$.*

Since homomorphically equivalent structures have the same CSP, one can focus on ω-categorical structures that have these properties. The fact that we can pp-define the orbits of n-tuples will now become highly important.

Theorem 15. *Let Γ be a constraint language over the domain D. Assume the following:*

1. *Γ is a model-complete ω-categorical core and*
2. *the domain elements are represented in a way such that given a vector $\bar{d} = (d_1, \ldots, d_n) \in D^n$, a pp-definition in Γ of the orbit of $\bar{d}$ can be generated in polynomial time (in the size of the representation of $d_1, \ldots, d_n$).*

Then, CSP(Γ) and CSP($\Gamma \cup D_c$) are polynomial-time equivalent.

Proof. Let $\Gamma' = \Gamma \cup D_c$. The reduction from CSP(Γ) to CSP(Γ') is trivial so we concentrate on the other direction. Let $I' = (V', C')$ be an instance of CSP(Γ'). Assume without loss of generality that if $\{d_i\}(x)$ is in C', then there is no variable $y \neq x$ such that $\{d_i\}(y) \in C'$; if so, the constraint $\{d_i\}(y)$ can be removed and the variable y be replaced by x. Normalising an instance in this way can easily be done in polynomial-time. We assume (without loss of generality) that the only constraints in C' with relations from D_c are $\{d_1\}(x_1), \ldots, \{d_m\}(x_m)$. This can be achieved in polynomial time by renaming of variables.

Compute (in polynomial time) the formula $F(x_1, \ldots, x_m)$ for the orbit of $(d_1, \ldots, d_m)$. Define $I = (V, C)$ such that the constraint set C equals C' extended with $F(x_1, \ldots, x_m)$ and with the constant relations removed. Let V denote V' expanded with the existentially quantified variables in $F(x_1, \ldots, x_m)$. Note that I can be constructed in polynomial time and it is an instance of CSP(Γ).

If the instance I has no solution, then it follows immediately that I' does not have a solution—one can view I as being a relaxation of I' since the formula $F(x_1, \ldots, x_m)$ is, in particular, satisfiable when $x_1 = d_1, \ldots, x_m = d_m$. If the instance I has a solution $s : V \to D$, then we claim that there is a solution $s' : V' \to D$ to I', too. Since F describes the orbit of $(d_1, \ldots, d_m)$, there is an automorphism α of Γ such that $\alpha(s'(x_i)) = d_i$, $1 \leq i \leq m$. This implies that $\alpha(s'(x))$ restricted to the set V is a solution to I. □

For those that are familiar with the algebraic approach to finite-domain CSPs, it may be illuminating to compare the proof of Theorem 15 with the proof of Theorem 4.7 in Bulatov et al. [20].

We concretise the method by presenting an example based on temporal constraint languages, i.e. constraint languages that are first-order definable in the

structure $(\mathbb{Q};<)$. We have already observed (in Section 3) that temporal constraint languages are ω-categorical. It is additionally known (by Junker and Ziegler [41], also see Cameron [21]) that there are five possible choices of $\mathrm{Aut}(\Gamma)$. We concentrate on the (for our purposes) most interesting case when $< \in \langle\Gamma\rangle$ and $\mathrm{Aut}(\Gamma) = \mathrm{Aut}(\mathbb{Q};<)$. Arbitrarily choose such a language Γ and assume (without loss of generality due to Theorem 4) that $< \in \Gamma$. We know that Γ is model-complete (by the discussion preceding Theorem 14) so we assume that Γ is a core. For instance, Γ may be the point algebra $\{<,>,=\}^{\vee=}$. We represent all members of $\mathbb{Q}$ in the natural way, i.e., as (a/b) where a, b are integers written in binary and $b \neq 0$.

The automorphisms of $(\mathbb{Q};<)$ are the bijective functions $f : \mathbb{Q} \to \mathbb{Q}$ that are monotonically increasing. The orbit of 1-tuples equals $\mathbb{Q}$ while the orbit of a 2-tuple (a, b) with $a < b$ equals $\{(x, y) \in \mathbb{Q}^2 \mid x < y\}$. More generally, the orbit of a k-tuple $(a_1, \ldots, a_k)$ with $a_1 < a_2 < \cdots < a_k$ equals

$$\{(x_1, \ldots, x_k) \in \mathbb{Q}^k \mid x_1 < x_2 < \cdots < x_k\}$$

so the orbit-defining formulas can be generated in polynomial time. Theorem 15 is thus applicable and $\mathrm{CSP}(\Gamma \cup \mathbb{Q}_c)$ is polynomial-time equivalent to $\mathrm{CSP}(\Gamma)$. In particular, $\mathrm{CSP}(\Gamma \cup \mathbb{Q}_c)$ is in P if $\mathrm{CSP}(\Gamma)$ is in P, and $\mathrm{CSP}(\Gamma \cup \mathbb{Q}_c)$ is in NP if $\mathrm{CSP}(\Gamma)$ is in NP.

This example shows that ω-categoricity is indispensable. Theorem 15 combined with the tractability of $\mathrm{CSP}((\mathbb{Q};<,\neq))$ implies that $\mathrm{CSP}(\Gamma_{\mathbb{Q}})$ is in P when $\Gamma_{\mathbb{Q}}$ denotes $(\mathbb{Q};<,\neq)$ extended with the constant relations in $\mathbb{Q}_c$. Recall that $(\mathbb{Z};<)$ and $(\mathbb{Z};<,\neq)$ are not ω-categorical and define $\Gamma_{\mathbb{Z}}$ by expanding $(\mathbb{Z};<,\neq)$ with $\mathbb{Z}_c$. The problem $\mathrm{CSP}(\Gamma_{\mathbb{Z}})$ is NP-hard since the relation $\{0, 1, 2\}$ can be pp-defined via

$$\{0, 1, 2\}(x) \Leftrightarrow \exists y, z.\{-1\}(y) \wedge \{3\}(z) \wedge y < x \wedge x < z,$$

and the problem $\mathrm{CSP}((\mathbb{Z};\{0, 1, 2\},\neq))$ is NP-hard since there is an obvious polynomial-time reduction from 3-Colourability. A similar example of this phenomenon (but based on RCC-5 instead of $(\mathbb{Q};<)$) will be presented in Section 6.2.

By Theorem 14, we know that orbit-defining formulas always can be pp-defined in Γ under the given assumptions. Whether these can be generated or not in polynomial time is a completely different question, though. Bodirsky [5, Sec. 7] notes that if the set of possible constants is finite, then an orbit-defining formula for these constants can be computed off-line and subsequently be used without additional cost. This gives us the following result.

Corollary 16. *Let Γ be a constraint language over the domain D and let D'_c be a finite subset of D_c. If Γ is a model-complete ω-categorical core, then CSP(Γ) and CSP($\Gamma \cup D'_c$) are polynomial-time equivalent problems.*

Assuming that the set of constant relations is finite is sensible in applications such as geographical information systems: geographical maps contain a comparatively small number of features and they tend to evolve quite slowly.

4.2. An example based on Allen's algebra

We will now illustrate the results presented in Section 4.1 with the aid of a slightly more involved example. There are several reasons for doing this. First of all, we want to show how the results can be used for studying non-binary constraint languages. One may argue that most constraint formalisms studied in AI are binary and studying higher-arity formalisms is of minor importance. However, there are many interesting formalisms based on relations with higher arity. We refer the reader to the survey by Dylla et al. [28] that contains both examples of higher-arity formalisms and a thorough discussion concerning their properties. Another reason is to give the reader some familiarity with the use of results such as Theorem 15. Even though it may seem quite abstract at first sight, it is both powerful and fairly easy to use in concrete applications.

Our departure is the following dichotomy result.

Theorem 17 (Theorem 5.5.23 in Bodirsky [6])**.** *Let Γ be a finite set of relations that are first-order definable in $\mathcal{A}$ and $\mathsf{m} \in \Gamma$ (where m is the Allen relation satisfying $(I, J) \in \mathsf{m}$ if and only if $I^+ = J^-$). Then, CSP(Γ) is either polynomial-time solvable or NP-complete.*

The reader may be puzzled about the restriction to constraint languages containing the relation m. Intuitively, m allows us to define certain relations that make endpoints identical and such relations are the very basis of the dichotomy result (see the construction in the beginning of Lemma 22).

Before continuing, we want to point out that switching from $\mathcal{A}^{\vee=}$ to relations that are first-order definable in $\mathcal{A}$ gives new possibilities but also poses new problems. Consider, for example, the following ternary Allen relation:

$$R^* = \left\{ (I, J, K) \in \mathbb{I}^3 \;\middle|\; \begin{array}{l} (\mathsf{s}, \mathsf{s}^\smile, \equiv)(I, J) \wedge (\mathsf{s}, \mathsf{s}^\smile, \equiv)(I, K) \wedge \\ (\mathsf{s}, \mathsf{s}^\smile, \equiv)(J, K) \wedge (\mathsf{s}(K, I) \vee \mathsf{s}(K, J)) \end{array} \right\}.$$

This relation expresses that I, J, and K have the same starting point and the ending point of K is before at least one of the ending points of I and J. Note that the disjunction $\mathsf{s}(K, I) \vee \mathsf{s}(K, J)$ is not expressible with relations in $\mathcal{A}^{\vee=}$. Relations that are first-order definable in $\mathcal{A}$ are inherently different from the relations in $\mathcal{A}^{\vee=}$. We exemplify by the Ord-Horn class $\mathcal{H}$ that was described in Section 3. Nebel and Bürckert [55] have proved that the ORD-Horn class $\mathcal{H}$ has the following uniqueness property: if $\mathcal{A} \subseteq X \subseteq \mathcal{A}^{\vee=}$ and CSP(X) in P, then $X \subseteq \mathcal{H}$. Such a unique class of relations does not exist if we consider relations that are first-order definable in $\mathcal{A}$: there exist two incomparable classes of relations X_1, X_2 that are first-order definable in $\mathcal{A}$, the ORD-Horn class is a strict subset of both, and CSP(X_i), $1 \leq i \leq 2$, is in P. This is a straightforward consequence of results proved by Bodirsky and Kára [14]; more details can be found in Bodirsky [6, Chapter 10]. One extension of $\mathcal{H}$ can be used for studying the relation R^*: CSP($\mathcal{H} \cup \{R^*\}$) is indeed polynomial-time solvable. First, R^* can be rewritten as a relation that is definable in $(\mathbb{Q}; <)$ by splitting the intervals

into variables that range over $\mathbb{Q}$:

$$R^{**} = \left\{ (I^-, I^+, J^-, J^+, K^-, K^+) \in \mathbb{Q}^6 \;\middle|\; \begin{array}{l} I^- < I^+ \wedge J^- < J^+ \wedge \\ K^- < K^+ \wedge \\ I^- = J^- = K^- \wedge \\ (K^+ < I^+ \vee K^+ < J^+) \end{array} \right\}.$$

Every relation in $\mathcal{H}$ can be rewritten in a similar way; we let $\mathcal{H}'$ denote the resulting constraint language. Now, the constraint language $\mathcal{H}' \cup \{R^{**}\}$ is a subset of the tractable *dual-ll* class [14], and we conclude that $\mathrm{CSP}(\mathcal{H} \cup \{R^*\})$ is tractable, too. This kind of transformations of constraint languages will be an important ingredient in the proofs of the forthcoming results.

Bodirsky's proof of Theorem 17 is based on the following fundamental result.

Theorem 18 (Bodirsky and Kára [13])**.** *Let Γ be a finite set of relations that are first-order definable in $(\mathbb{Q}; <)$. Then, CSP(Γ) is either polynomial-time solvable or NP-complete.*

The problem of applying Theorem 18 directly to Allen relations stems from the different domains: the theorem is concerned with the domain $\mathbb{Q}$ while Allen relations are defined over the interval domain $\mathbb{I}$. We will now generalise the dichotomy result for the first-order extension of Allen's algebra (Theorem 17) as follows: we show that adding constants to the constraint language do not change the complexity of the corresponding constraint satisfaction problem.

Theorem 19. *Let Γ be a finite set of relations that are first-order definable in $\mathcal{A}$ and $\mathsf{m} \in \Gamma$. Then, the problems CSP(Γ) and CSP($\Gamma \cup \mathbb{I}_c$) are polynomial-time equivalent. Additionally, these problems exhibit a dichotomy: they are either polynomial-time solvable or NP-complete depending on the choice of Γ.*

Theorem 19 can be proved in several different ways. Our proof does not use any complex formal machinery and it emphasises the connections with the results in Section 4.1. The proof can be found in Section 4.2.1 and we discuss some aspects of this result in Section 4.2.2. Special cases of Theorem 19 and related results have been studied in the literature. We have already mentioned that Li et al. [48] have showed that $\mathrm{CSP}(\mathcal{A}^{\vee=} \cup \mathbb{I}_c)$ is in NP. Their approach seems difficult to generalise for handling relations with arbitrarily high arity. Another example is Jonsson and Bäckström [38] who have proved that $\mathrm{CSP}(\mathcal{H} \cup \mathbb{I}_c)$ is in P. Their approach does not easily generalise to larger classes of relations, either. In fact, it cannot be used for proving that $\mathrm{CSP}(\mathcal{H} \cup \{R^*\} \cup \mathbb{I}_c)$ is in P—a result that can easily be inferred from Theorem 19.

4.2.1. Dichotomy result

We begin by proving that Allen relations admit quantifier elimination. The exact definition of homogeneity (which is used in the proof below) is not important at this point but we will come back to it in Section 5.

Lemma 20. *Structure $\mathcal{A}$ admits quantifier elimination.*

Proof. By Statement 2.22 in Cameron [21], we know that an ω-categorical structure admits quantifier elimination if and only if it is homogeneous. Corollary 5.9 in Hirsch [33] shows that $\mathcal{A}^{\vee=}$ is homogeneous. Every homogeneous structure that contains a finite number of relations and has an infinite countable domain is ω-categorical, cf. Lemma 3.2.10 in Bodirsky [6]. We conclude that $\mathcal{A}^{\vee=}$ admits quantifier elimination. This immediately implies that $\mathcal{A}$ admits quantifier elimination, too, since every relation in $\mathcal{A}^{\vee=}$ can be rewritten as a disjunction of relations in $\mathcal{A}$ without introducing any quantifiers. □

Let $R \subseteq \mathbb{I}^k$ denote a k-ary relation that is first-order definable in $\mathcal{A}$. Assume without loss of generality (due to Lemma 20) that $(I_1, \ldots, I_k) \in R$ if and only if $\phi_R(I_1, \ldots, I_k)$ holds where ϕ_R is a quantifier-free formula. Define the formula $X(\phi_R)$ to be a formula $\phi' \wedge \phi''$ with $2k$ free variables $x_1^-, x_1^+, \ldots, x_k^-, x_k^+$ where

1. $\phi' = x_1^- < x_1^+ \wedge \cdots \wedge x_k^- < x_k^+$ and
2. ϕ'' is ϕ where each atomic formula $\mathsf{a}(I_m, I_n)$ with $\mathsf{a} \in \mathcal{A}$ is replaced by the corresponding definition in Table 1 over variables $x_m^-, x_m^+, x_n^-, x_n^+$.

Note that the formula $X(\phi)$ is always quantifier-free first-order definable in $(\mathbb{Q}; <)$. We adapt $X(\cdot)$ for handling relations in the natural way: for a relation R with arity a, let

$$\hat{X}(R) = \{(x_1^-, x_1^+, \ldots, x_a^-, x_a^+) \in \mathbb{Q}^a \mid X(\phi_R)(x_1^-, x_1^+, \ldots, x_a^-, x_a^+)\}.$$

If we go back to the relations R^* and R^{**} that were introduced earlier, then we see that $R^{**} = \hat{X}(R^*)$. Finally, we extend $\hat{X}$ to constraint languages: $\hat{X}(\Gamma) = \{\hat{X}(R) \mid R \in \Gamma\}$. By the very definition of $\hat{X}(\cdot)$, it is straightforward to verify that CSP(Γ) is polynomial-time reducible to CSP($\hat{X}(\Gamma)$). The basic step is to replace each interval variable I with two point variables x^-, x^+ where x^- is interpreted as the starting point of I and x^+ the ending point of x^+. Showing that there is a polynomial-time reduction in the other direction needs some more work.

Lemma 21. *The relations* $(\mathsf{s}, \mathsf{s}^\smile, \equiv)$ *and* $(\mathsf{f}, \mathsf{f}^\smile, \equiv)$ *are pp-definable in* $\{\mathsf{m}\}$.

Proof. Note that

$$(\mathsf{s}, \mathsf{s}^\smile, \equiv)(I, J) \Leftrightarrow \exists K.\mathsf{m}(K, I) \wedge \mathsf{m}(K, J)$$

and that $(\mathsf{f}, \mathsf{f}^\smile, \equiv)$ can be pp-defined analogously. □

Lemma 22. *Let* Γ *be a finite set of relations that are first-order definable in* $\mathcal{A}$ *and* $\mathsf{m} \in \Gamma$. *Then, CSP*($\hat{X}(\Gamma)$) *is polynomial-time reducible to CSP*(Γ).

Proof. Assume without loss of generality (due to Lemma 21 and Theorem 4) that $\{(\mathsf{s}, \mathsf{s}^\smile, \equiv), (\mathsf{f}, \mathsf{f}^\smile, \equiv)\} \subseteq \Gamma$.

Arbitrarily choose an instance (V, C) of CSP($\hat{X}(\Gamma)$). We say that variables $x, y \in V$ are a *pair* if x appears at position k (where k is odd) in some constraint in C and y appears in the same constraint at position $k + 1$. Note that if x, y are a pair, then every solution must assign a strictly higher value to y than to x.

Based on (V, C), we define an instance (V', C') of CSP(Γ) as follows.

1. For each pair of variables $x, y \in V$, introduce an interval variable $I_{x,y}$ and put it into V'.
2. For variables $I_{x,y}$ and $I_{x,z}$ in V', add the constraint $(\mathsf{s}, \mathsf{s}^\smile, \equiv)(I_{x,y}, I_{x,z})$ to C'.
3. For variables $I_{y,x}$ and $I_{z,x}$ in V', add the constraint $(\mathsf{f}, \mathsf{f}^\smile, \equiv)(I_{x,y}, I_{x,z})$ to C'.
4. For variables $I_{y,x}$ and $I_{x,z}$ in V', add the constraint $\mathsf{m}(I_{y,x}, I_{x,z})$ to C'.
5. For variables $I_{x,y}$ and $I_{z,x}$ in V', add the constraint $\mathsf{m}(I_{z,x}, I_{x,y})$ to C'.
6. For each constraint $S(x_1, x_2, \ldots, x_{2m-1}, x_{2m}) \in C$, add the constraint $R(I_{x_1,x_2}, \ldots, I_{x_{2m-1},x_{2m}})$ to C' where $R \in \Gamma$ satisfies $S = \hat{X}(R)$.

We claim that (V', C') is satisfiable if and only if (V, C) is satisfiable.

Assume (V, C) is satisfiable and that it has the solution $f : V \to \mathbb{Q}$. Define the function f' such that $f'(I_{x,y}) = [f(x), f(y)]$. The variables x, y are a pair so they appear in some constraint that requires $f(x) < f(y)$. Thus, f' is a function from V' to $\mathbb{I}$. If there are interval variables $I_{x,y}$ and $I_{x,z}$ in V', then we know that the constraint $(\mathsf{s}, \mathsf{s}^\smile, \equiv)(I_{x,y}, I_{x,z})$ is in C'. The function f' satisfies this constraint since $f'(I_{x,y}) = [f(x), f(y)]$ and $f'(I_{x,z}) = [f(x), f(z)]$. It can analogously be verified that the constraints introduced in steps 3-5 are satisfied by f'. Finally, the constraints introduced in steps 6 are satisfied by f' due to the definition of $\hat{X}(\cdot)$.

Assume (V', C') is satisfiable and that it has the solution $f' : V' \to \mathbb{I}$. For each variable $x \in V$, there is at least one variable $I_{x,y}$ or $I_{y,x}$ in V' for some $y \in V$. Arbitrarily choose a function $g : V \to V'$ such that for every $x \in V$, $g(x) = I_{x,y}$ or $g(x) = I_{y,x}$ (for some $y \in V$) and $g(x) \in V'$. Define a function $f : V \to \mathbb{Q}$ as follows:

1. $f(x) = a$ if $g(x) = I_{x,y}$ and $f'(I_{x,y}) = [a, b]$ or
2. $f(x) = b$ if $g(x) = I_{y,x}$ and $f'(I_{y,x}) = [a, b]$.

Note that the exact choice of function g for defining $f(x)$ is irrelevant: whenever we have interval variables, say, $I_{x,y}$ and $I_{x,z}$, then $f'(I_{x,y}) = [a, b]$ and $f'(I_{x,z}) = [a, c]$, due to the constraints introduced in steps 2-5 in the reduction. Thus, there exists exactly one function f corresponding to a given solution f'.

We conclude the proof by proving that f is a solution to (V, C). Arbitrarily choose a constraint $S(x_1, x_2, \ldots, x_{2m-1}, x_{2m})$ in C. Assume first that all variables are distinct. We need to verify that $(f(x_1), f(x_2), \ldots, f(x_{2m-1}), f(x_{2m})) \in S$. We know that $(f'(I_{x_1,x_2}), \ldots, f'(I_{x_{2m-1},x_{2m}})) \in R$ so the very construction of f combined with the definition of $\hat{X}(\cdot)$ implies that this holds. Assume now instead that the variables are not distinct. We exemplify with a 4-ary constraint $S(x, y, x, z)$. This corresponds to a constraint $R(I_{x,y}, I_{x,z})$ in C' where we additionally have the constraint $(\mathsf{s}, \mathsf{s}^\smile, \equiv)(I_{x,y}, I_{x,z}) \in C'$ due to step 2 in the reduction. It is thus guaranteed that the intervals $f'(I_{x,y})$ and $f'(I_{x,z})$ start at the same point. Since (V', C') has a solution, we know that $(f'(I_{x,y}), f'(I_{x,z})) \in R$. This implies that $(f(x), f(y), f(x), f(z)) \in S$ due to the definition of $\hat{X}(\cdot)$. It is not hard to generalise this reasoning to constraints of higher arity. □

We are now ready to prove Theorem 19.

Proof. We prove the result by giving four polynomial-time reductions.

1. $\mathrm{CSP}(\Gamma \cup \mathbb{I}_c)$ to $\mathrm{CSP}(\hat{X}(\Gamma) \cup \mathbb{Q}_c)$. We have earlier discussed the fact that there is a straightforward reduction from $\mathrm{CSP}(\Gamma)$ to $\mathrm{CSP}(\hat{X}(\Gamma))$. Assume that we have a unary relation that constrains the interval variable I to have the value $[a, b]$. In the reduction, I is associated with two variables x^- and x^+ ranging over $\mathbb{Q}$. It is easy to see that we can replace a constraint $[a, b](I)$ with $[a, b] \in \mathbb{I}_c$ with the constraints $\{a\}(x^-)$ and $\{b\}(x^+)$ where $\{a\}, \{b\} \in \mathbb{Q}_c$.
2. $\mathrm{CSP}(\hat{X}(\Gamma) \cup \mathbb{Q}_c)$ to $\mathrm{CSP}(\hat{X}(\Gamma))$. We merely note that the relation $<$ is pp-definable in $\hat{X}(\Gamma)$ since $x < y \Leftrightarrow \exists z, w.\hat{X}(\mathsf{m})(x, z, w, y)$ so the reduction exists due to the discussion after Theorem 15.
3. $\mathrm{CSP}(\hat{X}(\Gamma))$ to $\mathrm{CSP}(\Gamma)$. This reduction exists due to Lemma 22.
4. $\mathrm{CSP}(\Gamma)$ to $\mathrm{CSP}(\Gamma \cup \mathbb{I}_c)$. Trivial.

For an arbitrary constraint language Θ that is first-order definable in $(\mathbb{Q}; <)$, we know that $\mathrm{CSP}(\Theta)$ is either in P or is an NP-complete problem by Theorem 18. Since $\hat{X}(\Gamma)$ is first-order definable in $(\mathbb{Q}; <)$, the reduction above shows that $\mathrm{CSP}(\Gamma)$, $\mathrm{CSP}(\Gamma \cup \mathbb{I}_c)$, $\mathrm{CSP}(\hat{X}(\Gamma))$, and $\mathrm{CSP}(\hat{X}(\Gamma) \cup \mathbb{Q}_c)$ are either in P or they are NP-complete problems. □

Theorem 19 only gives the complexity for $\mathrm{CSP}(\Gamma \cup \mathbb{I}_c)$ and not for $\mathrm{CSP}(\Gamma \cup \mathbb{I}_f)$. However, we can conclude (by Lemma 3) that $\mathrm{CSP}(\Gamma \cup \mathbb{I}_f)$ is in NP. Recall that the complexity of $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Gamma \cup \mathbb{I}_c)$ is always the same (up to polynomial-time reductions) when $\mathsf{m} \in \Gamma$. This does not hold for $\mathrm{CSP}(\Gamma \cup \mathbb{I}_f)$. Let $\Theta = \{\mathsf{m}, \not\equiv\}$ where $\not\equiv = \bigcup_{B \in \mathcal{A} \setminus \{\equiv\}} B$. $\mathrm{CSP}(\Theta)$ is in P since Θ is a subset of the Ord-Horn class $\mathcal{H}$. The problem $\mathrm{CSP}(\Theta \cup \mathbb{I}_f)$ is NP-complete, though. This can be proved by a reduction from 3-Colourability. Let $G = (V, E)$ be an undirected graph. For each $v \in V$, introduce a variable v' and the constraint $\{[0, 1], [1, 2], [2, 3]\}(v')$. For each edge $(v, w) \in E$, introduce the constraint $\not\equiv (v', w')$. It is easy to see that the resulting instance is satisfiable if and only if G is 3-colourable, and this implies that $\mathrm{CSP}(\Gamma \cup \mathbb{I}_f)$ is NP-complete.

4.2.2. Discussion

It is now suitable to discuss some aspects of the dichotomy result presented in the previous section. One obvious question is what happens if we consider languages that do not contain the relation m. Unfortunately, the proof breaks down. Let $\Gamma = \{R_1, \ldots, R_n\}$ be first-order definable in $\mathcal{A}$ such that $\mathsf{m} \in \Gamma$ and R_i, $1 \leq i \leq n$, is defined by the quantifier-free formula ϕ_i. The reductions in the proof of Theorem 19 shows that the problems $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Theta)$ have the same complexity up to polynomial-time reductions where the relations in Θ are defined via $X(\phi_1), \ldots, X(\phi_n)$. We demonstrate that this is not true in the general case where we do not have access to the relation m.

Just as in the previous section, we let $\not\equiv$ denote the 'not equal' relation, i.e., the relation $\not\equiv = \bigcup_{B \in \mathcal{A} \setminus \{\equiv\}} B$. Let $\phi_R(I, J, X, Y)$ denote the formula

$$\mathsf{m}(I, J) \to \not\equiv(X, Y)$$

and $\phi_S(X, A, B, C)$ the formula

$$\equiv(X, A) \vee \equiv(X, B) \vee \equiv(X, C).$$

Define the relations $R, S \subseteq \mathbb{I}^4$ as follows:

$$R = \{(I, J, X, Y) \in \mathbb{I}^4 \mid \phi_R(I, J, X, Y)\} \text{ and}$$

$$S = \{(X, A, B, C) \in \mathbb{I}^4 \mid \phi_S(X, A, B, C)\}.$$

Let $\Gamma' = \{R, S\}$. Clearly, $\mathsf{m} \notin \Gamma'$ and the relations in Γ' are first-order definable in $\mathcal{A}$ by definition. Observe that the problem CSP(Γ') is in P: if (V, C) is an instance of CSP(Γ'), then the constant function $f(v) = [0, 1]$ is a solution to (V, C). Now take $\Theta' = \{R', S'\}$ where

$$R' = \{(x_1, \ldots, x_8) \in \mathbb{Q}^8 \mid X(\phi_R)(x_1, \ldots, x_8)\} \text{ and}$$

$$S' = \{(x_1, \ldots, x_8) \in \mathbb{Q}^8 \mid X(\phi_S)(x_1, \ldots, x_8)\}.$$

We show that the problem CSP($\{R', S'\}$) is NP-hard by a reduction from 3-Colourability. Let (V, E) denote an arbitrary undirected graph. Introduce "colour variables" $c_1^-, c_1^+, c_2^-, c_2^+, c_3^-, c_3^+$, auxiliary variables a_1, a_2, a_3, and impose the constraints

(1) $R'(a_1, a_2, a_2, a_3, c_1^-, c_1^+, c_2^-, c_2^+)$,

(2) $R'(a_1, a_2, a_2, a_3, c_1^-, c_1^+, c_3^-, c_3^+)$, and

(3) $R'(a_1, a_2, a_2, a_3, c_2^-, c_2^+, c_3^-, c_3^+)$.

We first note that these three constraints are simultaneously satisfiable: for instance, every function $f : \{a_1, a_2, a_3\} \cup \{c_i^-, c_i^+ \mid 1 \leq i \leq 3\} \to \mathbb{Q}$ such that

$$f(a_1) < f(a_2) < f(a_3) <$$

$$f(c_1^-) < f(c_1^+) < f(c_2^-) < f(c_2^+) < f(c_3^-) < f(c_3^+)$$

is a solution. By inspecting the definition of R', it is evident that every solution f must satisfy $f(a_1) < f(a_2) < f(a_3)$. Given this, it follows that for arbitrary $1 \leq i \neq j \leq 3$, either $f(c_i^-) \neq f(c_j^-)$ or $f(c_i^+) \neq f(c_j^+)$ (or both). We conclude that $[f(c_1^-), f(c_1^+)]$, $[f(c_2^-), f(c_2^+)]$, and $[f(c_3^-), f(c_3^+)]$ are distinct intervals.

For each vertex $v \in V$, introduce variables v^-, v^+ and the constraint

$$S'(v^-, v^+, c_1^-, c_1^+, c_2^-, c_2^+, c_3^-, c_3^+).$$

This constraint implies that $[f(v^-), f(v^+)] \in \{[f(c_i^-), f(c_i^+)] \mid 1 \leq i \leq 3\}$. Thus, $[f(v^-), f(v^+)]$ may have one of three possible values due to constraints (1)–(3). This value correspond to the colour given to vertex v.

Finally, for each edge $(v, w) \in E$, introduce the constraint

$$R'(a_1, a_2, a_2, a_3, v^-, v^+, w^-, w^+).$$

This constraint ensures that $[f(v^-), f(v^+)] \neq [f(w^-), f(w^+)]$, i.e., adjacent vertices are assigned different colours. It is not hard to verify that the resulting instance is satisfiable if and only if (V, E) is 3-colourable. This shows that the complexities for CSP(Γ) and CSP(Θ) do not always match when $\mathsf{m} \notin \Gamma$.

Another interesting question concerns the complexity of the *metaproblem*: given a constraint language Γ such that $\mathsf{m} \in \Gamma$ and Γ is first-order definable in $\mathcal{A}$, what is the complexity of CSP(Γ)? Preferably, we want problems of this kind to be decidable. Let us first consider the structure $(\mathbb{Q}; <)$. Let Θ be a finite set of relations that are first-order definable in $(\mathbb{Q}; <)$. Bodirsky and Kára [13] have showed that there are nine different cases where CSP(Θ) is polynomial-time solvable, and they have algebraically characterised these classes. A similar algebraic classification is possible for CSP(Γ) and, consequently, CSP($\Gamma \cup \mathbb{I}_c$) by Theorem 19—the complexity of such a CSP is determined by a corresponding set of relations that are first-order definable in $(\mathbb{Q}; <)$. In practice, the following result may be more useful.

Theorem 23 (Bodirsky and Kára [13], see also Bodirsky et al. [15])**.** *Let $\Gamma = \{R_1, \ldots, R_n\}$ be a finite set of relations that are first-order definable in $(\mathbb{Q}; <)$, and let $\phi_1, \ldots, \phi_n$ denote their quantifier-free first-order definitions. There is an algorithm that given $\phi_1, \ldots, \phi_n$ decides whether CSP(Γ) is polynomial-time solvable or NP-complete.*

As far as we know, there are no specialised algorithms for performing quantifier elimination in Allen's algebra described in the literature. There are, however, general algorithms that can be used for this purpose.

Proposition 24 (Proposition 3.1.22 in Marker [54])**.** *Suppose that T is a decidable theory with quantifier elimination. Then, there is an algorithm which when given a formula ϕ as input will output a quantifier-free formula ψ such that $T \models \forall x_1, \ldots, x_n.\phi(\bar{x}) \leftrightarrow \psi(\bar{x})$.*

The theory of Allen relations is first-order axiomatisable: this follows from the fact that the theory of dense linear order has this property, (cf. Section 1.2 in Marker [54]) combined with the definitions of the basic relations in $\mathcal{A}$. By choosing T to be the theory of Allen relations, it follows from Proposition 24 that there exists a suitable quantifier elimination algorithm.

Theorem 25. *Let $\Gamma = \{R_1, \ldots, R_n\}$ be a finite set of relations that are first-order definable in $\mathcal{A}$ and contains the relation m, and let $\phi_1, \ldots, \phi_n$ denote their first-order definitions. There is an algorithm that given $\phi_1, \ldots, \phi_n$ decides whether CSP(Γ) is polynomial-time solvable or NP-complete.*

Proof. The first-order formulas $\phi_1, \ldots, \phi_n$ can be algorithmically converted into logically equivalent quantifier-free formulas $\phi'_1, \ldots, \phi'_n$, as described above. Given $\phi'_1, \ldots, \phi'_n$, we can easily compute the quantifier-free formulas $X(\phi'_1), \ldots, X(\phi'_n)$ by exploiting the definitions in Table 1. We note the following.

- $\mathrm{CSP}(\Gamma)$ is polynomial-time equivalent with $\mathrm{CSP}(\hat{X}(\Gamma))$ by Lemma 22, and
- the computational complexity of $\mathrm{CSP}(\hat{X}(\Gamma))$ is decidable by Theorem 23 since we have access to the formulas $X(\phi'_1), \ldots, X(\phi'_n)$ that define the relations in $\hat{X}(\Gamma)$.

Combining these two facts concludes the proof. □

The same result holds for $\mathrm{CSP}(\Gamma \cup \mathbb{I}_c)$ by Theorem 19. Even though the metaproblem is indeed decidable, it is quite evident that the method outlined above is time-consuming. We want to stress that there may exist more efficient methods: one encouraging example is conservative constraints over finite domains. They admit a polynomial-time algorithm for the meta-problem whereas the straightforward algorithm is super-exponential [23].

5. Method II: Homogeneous structures

Our second method is based on analysing a given constraint language Γ with respect to its automorphisms, just as method I. However, the central notion in method II is *homogeneity* instead of ω-categoricity. In our context, homogeneous structures are typically ω-categorical so method I is, in principle, applicable. The main advantage of working with homogeneous structures is that we do not need a method for computing orbit-defining formulas. This advantage comes at a price: the method is restricted to constraint languages based on partition schemes.

Homogeneous[2] structures have been intensively studied in mathematics and logics (for instance, in connection with combinatorics, model theory, and group theory) and they are also highly relevant in the study of CSPs. Homogeneous structures have useful properties such as admitting quantifier elimination and they are ω-categorical whenever the structure contains a finite number of relations and the domain is countably infinite. Theoretically interesting examples include $(\mathbb{Q}; <)$, $(\mathbb{N}; =)$, and the *random* (or Rado) graph. There are also many examples that are (more obviously) relevant in AI and computer science. One of the earliest studies of homogeneous structures in AI was performed by Hirsch [33]. He proved, among other things, that Allen's algebra under the interval representation is homogeneous. Another early example is the homogeneous representation of RCC-5 proposed by Bodirsky and Chen [7]; this is the ω-categorical representation that was briefly discussed in Section 3. Other examples include RCC-8 [16], phylogeny constraints [12], and temporal constraints for partially-ordered time [42].

[2]The term *ultra-homogeneous* is sometimes used in the literature.

We need some machinery before providing the formal definition of homogeneity. Let D be the domain of a relational τ-structure Γ and arbitrarily choose $S \subseteq D$. Then the *substructure induced by S in* Γ is the τ-structure Δ with domain S such that $R^\Delta = R^\Gamma \cap S^n$ for each n-ary $R \in \tau$; we also write $\Gamma[S]$ for Δ. The structure Γ is called *homogeneous* if every isomorphism $f : D_1 \to D_2$ between finite induced substructures of Γ can be extended to an automorphism of Γ, that is, there exists an automorphism α such that $f(x) = \alpha(x)$ when $x \in D_1$. The survey by Macpherson [53] is a good introduction to homogeneous structures. We give a very simple example to illustrate the idea behind homogeneity.

Example 26. *We revisit Example 1. Assume $k \geq 1$ to be fixed, let $D = \{1, \ldots, k\}$, and let K_k denote the binary relation $\{(x, y) \in D^2 \mid x \neq y\}$. One may easily verify that $\mathrm{Aut}(\{K_k\})$ equals the full symmetric group on the set D, i.e., the group of all permutations of D.*

Arbitrarily choose $D_1, D_2 \subseteq D$ and an isomorphism $f : D_1 \to D_2$ between $K_k[D_1]$ and $K_k[D_2]$. Note that $|D \setminus D_1| = |D \setminus D_2|$ and let $g : D \setminus D_1 \to D \setminus D_2$ be an arbitrary bijection. If we define $\alpha : D \to D$ such that $\alpha(d) = f(d)$ when $d \in D_1$ and $\alpha(d) = g(d)$ otherwise, then α is an extension of f and α is a permutation on D which implies that $\alpha \in \mathrm{Aut}(\{K_k\})$. Since D_1, D_2, and f were arbitrarily chosen, we conclude that the structure $(D; K_k)$ is homogeneous.

Many additional examples can be found in, for instance, the article by Hirsch [33]. Homogeneity is a more "fragile" concept than ω-categoricity. For instance, Γ being homogeneous and Δ being first-order definable in Γ does not necessarily imply that Δ is homogeneous.

In the following, we will concentrate on partition schemes as defined in Section 2.2.

Theorem 27. *Let $\mathcal{B} = \{B_1, \ldots, B_k\}$ be a partition scheme over the domain D and let $\mathcal{B} \subseteq \Gamma \subseteq \mathcal{B}^{\vee=}$. Assume the following:*

1. *Γ is homogeneous, and*
2. *the domain elements are represented in a way such that given two elements $a, b \in D$, it is possible to find (by using an algorithm A) the unique B_i, $1 \leq i \leq k$, such that $(a, b) \in B_i$ in polynomial time (measured in the size of the representations of a and b).*

Then, CSP(Γ) and CSP($\Gamma \cup D_c$) are polynomial-time equivalent.

Proof. Let $\Gamma' = \Gamma \cup D_c$. The reduction from CSP(Γ) to CSP(Γ') is trivial so we concentrate on the other direction. Let $I' = (V', C')$ be an instance of CSP(Γ'). We assume without loss of generality (just as in the proof of Theorem 15) that the only constraints in C' with relations from D_c are $\{d_1\}(x_1), \ldots, \{d_m\}(x_m)$.

Construct an instance $I = (V, C)$ of CSP(Γ) as follows: let

- $V = V'$,
- $\widehat{C} = \{B(x_i, x_j) \mid 1 \leq i \neq j \leq m \text{ and } B = A(d_i, d_j)\}$, and

- $C = (C' \cup \widehat{C}) \setminus \{\{d_1\}(x_1), \ldots, \{d_m\}(x_m)\}$.

The instance $I = (V, C)$ can obviously be generated in polynomial time.

If the instance I' has a solution, then it follows immediately that I has a solution—the constraints in $\widehat{C}$ are satisfiable by the assignment $x_1 = d_1, \ldots, x_m = d_m$.

If the instance I has a solution $s : V \to D$, then we claim that there is a solution $s' : V \to D$ to I', too. Let $S = \{s(x_1), \ldots, s(x_m)\}$ and $T = \{d_1, \ldots, d_m\}$. The set T contains m elements by our initial assumptions and the set S contains m elements due to the constraints in $\widehat{C}$; all variables in $\{x_1, \ldots, x_m\}$ are assigned distinct values since none of the constraints in $\widehat{C}$ allows equality (due to the fact that $\mathcal{B}$ is a partition scheme and $d_1, \ldots, d_m$ are distinct values). Thus, $f : S \to T$ is a well-defined bijective function if we let $f(s(x_i)) = d_i$, $1 \leq i \leq m$. We continue by proving the following claim.

Claim: f is an homomorphism from $B[S]$ to $B[T]$ when $B \in \mathcal{B}$. Arbitrarily choose a tuple $(a, b) \in B[S]$. By the choice of S, we know that $a = s(x_i)$ and $b = s(x_j)$ for some distinct $1 \leq i, j \leq m$. We see that

$$(f(a), f(b)) = (f(s(x_i)), f(s(x_j))) = (d_i, d_j).$$

We know that $d_i, d_j \in T$ so it remains to show that $(d_i, d_j) \in B$. If $A(d_i, d_j) = B$, then we are done. If $A(d_i, d_j) = B' \neq B$, then $B'(x_i, x_j) \in \widehat{C} \subseteq C$ so $(s(x_i), s(x_j)) \in B'$. This contradicts that $a = s(x_i)$, $b = s(x_j)$, and $(a, b) \in B$ since $B \cap B' = \varnothing$.

We show that f is an isomorphism between $\Gamma[S]$ and $\Gamma[T]$. We have already verified that f is a bijective function. Hence, we need to show that f is a homomorphism from $\Gamma[S]$ to $\Gamma[T]$, and that the inverse function f^{-1} is a homomorphism from $\Gamma[T]$ to $\Gamma[S]$.

We begin by showing that f is a homomorphism from $\Gamma[S]$ to $\Gamma[T]$. Arbitrarily choose a relation $R \in \Gamma$ where $R = B_1 \cup \cdots \cup B_p$ and $B_i \in \mathcal{B}$, $1 \leq i \leq p$. Arbitrarily choose $(a, b) \in R[S]$. The tuple (a, b) is a member of some relation B_i in $\{B_1, \ldots, B_p\}$. By the Claim, $(f(a), f(b)) \in B_i[T]$ so $(f(a), f(b)) \in R[T]$ since $B_i \subseteq R$. It follows that f is a homomorphism from $R[S]$ to $R[T]$ since (a, b) was arbitrarily chosen in $R[S]$. This, in turn, implies that f is a homomorphism from $\Gamma[S]$ to $\Gamma[T]$ since R was arbitrarily chosen in $\mathcal{B}^{\vee=}$.

Next, we show that f^{-1} is a homomorphism from $\Gamma[T]$ to $\Gamma[S]$. Arbitrarily choose a relation $R \in \Gamma$ where $R = B_1 \cup \cdots \cup B_p$ and $B_i \in \mathcal{B}$, $1 \leq i \leq p$. Arbitrarily choose $(d_i, d_j) \in R[T]$. The tuple (d_i, d_j) is a member of some relation B_m in $\{B_1, \ldots, B_p\}$. Now, consider the tuple $(f^{-1}(d_i), f^{-1}(d_j))$. By the definition of f, we see that $(f^{-1}(d_i), f^{-1}(d_j)) = (s(x_i), s(x_j))$. If $(s(x_i), s(x_j)) \in B_m$, then we are done. Assume to the contrary that $(s(x_i), s(x_j)) \in B_n$ where $n \neq m$. By the Claim, $(f(s(x_i)), f(s(x_j))) \in B_n$ and $(f(s(x_i)), f(s(x_j))) = (d_i, d_j)$. This leads to a contradiction since (d_i, d_j) cannot simultaneously be a member of B_n and B_m due to the fact that $\mathcal{B}$ is a partition scheme.

Since Γ is a homogeneous structure, the function f can be extended to an automorphism α of Γ. It follows that the function $s' : V \to D$ defined such that $s'(x) = \alpha(s(x))$ is a solution to I'; merely note that $s'(x_i) = d_i$, $1 \leq i \leq m$. $\square$

The major advantage of working with homogeneous structures instead of using Method I is that we do not need any efficient algorithm for computing orbit formulas. Presumably, the computations required by condition (2) in Theorem 27 are easier to carry out than computing orbit formulas given that domain elements are represented in some suitable way. In fact, if the constants represent objects in the real world, we may very well know their relations *a priori* and we do not need an algorithm for computing them.

We exemplify the result by taking a look at Allen's algebra with domain $\mathbb{I}$. Hirsch [33] has shown that $\mathcal{A}^{\vee=}$ is a homogeneous structure and the second precondition of Theorem 27 is clearly satisfied with the given representation. We conclude that $\mathrm{CSP}(\mathcal{A}^{\vee=} \cup \mathbb{I}_c)$ is an NP-complete problem since $\mathrm{CSP}(\mathcal{A}^{\vee=})$ is NP-complete. By Lemma 3, it follows directly that $\mathrm{CSP}(\mathcal{A}^{\vee=} \cup \mathbb{I}_f)$ is an NP-complete problem, too. One may also note that $\mathrm{CSP}(\mathcal{H} \cup \mathbb{I}_c)$ is in P when $\mathcal{H}$ is the ORD-Horn subclass [55] since $\mathcal{H}$ contains all 13 basic relations.

RCC-8 may serve as another example. We first recall the result for RCC-8 presented by Li et al. [48]. They use a representation of RCC-8 where the objects are regions in the plane, and they show that RCC-8 extended with polygonal constants give rise to an NP-complete problem. Unfortunately, their representation of RCC-8 is not known to be homogeneous so method II is not applicable in this case. There are other representations that are homogeneous, though: one example is presented by Bodirsky and Wölfl [16]. In this case, we do not have an explicit method for analysing the relationships between domain elements. Thus, either one has to construct such a method or one may restrict oneself to objects with known relationships. In particular, one may restrict oneself to finite sets of constant relations. If D'_c is a finite set of constant relations, then the relations between the corresponding domain elements can be computed in advance. Since D'_c is finite, this information can be stored in a finite table which can be efficiently accessed. This gives us a result that is analogous to Corollary 16.

Corollary 28. *Let $\mathcal{B} = \{B_1, \ldots, B_k\}$ be a partition scheme over the domain D and let $\mathcal{B} \subseteq \Gamma \subseteq \mathcal{B}^{\vee=}$. Let D'_c denote a finite subset of D_c. If Γ is homogeneous, then CSP(Γ) and CSP($\Gamma \cup D'_c$) are polynomial-time equivalent.*

One may additionally note that Corollary 28 is applicable to RCC-$5_{\omega-\mathrm{cat}}$ since this representation is known to be homogeneous [7].

Theorem 27 can be utilised in many other ways. One example is the following result which "lifts" Theorem 27 to first-order definable relations. Note, however, that this result is only useful for proving membership in NP.

Corollary 29. *Let $\mathcal{B} = \{B_1, \ldots, B_k\}$ be a partition scheme and let $\mathcal{B} \subseteq \Gamma \subseteq \mathcal{B}^{\vee=}$. Assume the following:*

1. *Γ is homogeneous,*

2. $\mathcal{B}$ *satisfies precondition (2) of Theorem 27, and*
3. *CSP*($\mathcal{B}$) *is in NP.*

For every finite set $\Theta = \{R_1, \ldots, R_k\}$ *of relations that are first-order definable in* Γ, *CSP*($\Theta \cup D_c$) *and CSP*($\Theta \cup D_f$) *are in NP.*

Proof. Arbitrarily choose a $\Theta = \{R_1, \ldots, R_k\}$ that satisfies the preconditions. Note that CSP(Γ) is in NP since CSP($\mathcal{B}$) is in NP and $\Gamma \subseteq \mathcal{B}^{\vee=}$; this can easily be proved along the same lines as Lemma 6. By Theorem 27, CSP($\Gamma \cup D_c$) is in NP since CSP(Γ) is in NP. This implies that CSP($\mathcal{B} \cup D_c$) is in NP since $\mathcal{B} \subseteq \Gamma$. By arguing as in the proof of Lemma 20, we may assume that each R_i, $1 \leq i \leq k$, has a quantifier-free definition in $\mathcal{B}$. It follows from Lemma 6 that CSP($\Theta \cup D_c$) is in NP, too. By Lemma 3, it additionally holds that CSP($\Theta \cup D_f$) is in NP. □

6. Method III: Small solutions

The methods in Sections 4 and 5 provide polynomial-time equivalences between CSP(Γ) and CSP($\Gamma \cup D_c$) under certain conditions. In this section, we will instead analyse the constraint language $\Gamma \cup D_c$ directly. The main result will be weaker than in the previous two sections since we will only be able to prove membership in NP. On the other hand, the approach is applicable also without ω-categoricity.

6.1. The small solution property

Let Γ be an arbitrary constraint language with domain D, and assume that the relations in Γ and the elements in D are represented is some fixed way. We say that Γ has the *small solution property* if there exists a polynomial p (that only depends on the choice of Γ) such that for every satisfiable instance $I = (V, C)$ of CSP(Γ), there exists a solution $s : V \to D$ such that $||s(v)|| \leq p(||I||)$ for every $v \in V$.

Lemma 30. *Let* Γ *denote a constraint language over the domain* D. *Assume that*

1. Γ *has the small solution property and*
2. *there exists an algorithm* A *and a polynomial* q *such that for arbitrary* k*-ary* $R \in \Gamma$ *and* $d_1, \ldots, d_k \in D$, *algorithm* A *can verify whether* $(d_1, \ldots, d_k) \in R$ *in time* $O(q(||R|| + \sum_{i=1}^{k} ||d_i||))$.

Then CSP(Γ) *is in NP.*

Proof. Let (V, C) denote an arbitrary instance of CSP(Γ). To show that $I = (V, C)$ is satisfiable, non-determinstically guess a solution $s : V \to D$ such that $||s(v)|| \leq p(||I||)$ for every $v \in V$ (where p denotes a fixed polynomial). Such a solution exists since Γ has the small solution property, and the size of s is consequently polynomially bounded in $||I||$. The solution s can thus be verified in polynomial time with the aid of algorithm A. □

Many well-known structures possess the small solution property. One example is the temporal constraint problem that we introduced in Sec. 3: if (V, C) is an instance of this problem with a solution $s : V \to \mathbb{Q}$, then there exists a solution $t : V \to \{1, \ldots, |V|\}$, too, and this solution can be represented by approximately $|V| \cdot \log_2 |V|$ bits. The existence of t can be shown along the following lines: let $S = \{s(v) \mid v \in V\} = \{a_1, \ldots, a_p\}$ with $a_1 < a_2 < \cdots < a_p$, and define $t(v) = i$ if and only if $s(v) = a_i$. It is easy to verify that t is indeed a solution to (V, C). This idea can, for instance, be used for verifying that Allen's algebra has the small solution property.

Another example is relations R defined by linear expressions, that is, R is defined by

$$(x_1, \ldots, x_k) \in R \Leftrightarrow \sum_{i=1}^{k} c_i \cdot x_i \leq c_0$$

or

$$(x_1, \ldots, x_k) \in R \Leftrightarrow \sum_{i=1}^{k} c_i \cdot x_i = c_0$$

where the coeffecients are in $\mathbb{Z}$ and the variables range over, for instance, $\mathbb{Q}$ or $\mathbb{Z}$. Given a constraint language Γ containing such relations, the small solution property for $\mathbb{Q}$ follows from the fact that linear programming can be solved (and a concrete solution written down) in polynomial time while the property for $\mathbb{Z}$ has been proven by Papadimitriou [57]. This example is interesting in several respects. First of all, the constants in $\mathbb{Q}_c$ are, of course, linear. Furthermore, we know (from Example 8) that not even the language $\Gamma = \{\{(x, y, z) \in \mathbb{Z}^3 \mid x+y = z\}\}$ is ω-categorical; the same can be proved for the domain $\mathbb{Q}$. Thus, the methods in Section 4 and 5 are not applicable in this case.

An important observation is that it is *not* sufficient to verify that Γ itself has the small solution property—one needs to verify that $\Gamma \cup D_c$ has the small solution property if one wants to use Lemma 30 in connection with constants. We exemplify by using the relation $R = \{(x, y) \in \mathbb{N}^2 \mid x = 2^{y-1}\}$. The constraint language $\{R\}$ has the small solution property since every instance has the solution that assigns 1 to every variable. However, $\mathrm{CSP}(\{R, \{2\}\})$ does not have small solutions if we assume integers to be written in binary. Consider the instance (V, C) where $V = \{x_0, \ldots, x_n\}$ and

$$C = \{\{2\}(x_0), R(x_1, x_0), R(x_2, x_1), \ldots, R(x_n, x_{n-1})\}.$$

It is easy to verify that (V, C) is solvable and every solution $s : V \to \mathbb{N}$ must satisfy $s(x_n) = \mathrm{Tower}(n)$ where Tower is the rapidly increasing function

$$\mathrm{Tower}(n) = \underbrace{2^{2^{\cdot^{\cdot^{2}}}}}_{n \text{ times}}.$$

The small solution property is particularly useful in connection with relations that are constructed via logical definitions.

Corollary 31. *Let Γ be a set of relations with domain D such that precondition (2) of Lemma 30 is satisfied and $\Gamma \cup D_c$ has the small solution property.*

1. *If Γ is a partition scheme and Θ is a finite set of relations that are quantifier-free definable in Γ, then CSP($\Theta \cup D_c$) and CSP($\Theta \cup D_f$) are in NP.*
2. *If Θ is a finite set of relations that are quantifier-free positive definable in Γ, then CSP($\Theta \cup D_c$) and CSP($\Theta \cup D_f$) are in NP.*

Proof. We begin by proving case 1. Hence, assume that Γ is a partition scheme. We know that $\Gamma \cup D_c$ has the small solution property so CSP($\Gamma \cup D_c$) is in NP by Lemma 30. Lemma 6 implies that CSP($\Theta \cup D_c$) is in NP and, consequently, CSP($\Theta \cup D_f$) is in NP by Lemma 3.

We continue by proving case 2. We know that Θ has a quantifier-free positive definition in Γ. This and the fact that CSP($\Gamma \cup D_c$) is in NP allow us to apply Lemma 7 and conclude that CSP($\Theta \cup D_c$) is in NP and that CSP($\Theta \cup D_f$) is in NP by Lemma 3. □

We exemplify Corollary 31 with the point algebra PA over $\mathbb{Z}$; recall from Sec. 2.3 that $(\mathbb{Z}; <)$ is not ω-categorical so methods I and II are not applicable. Let Γ denote the basic relations $\{<, >, =\}$ over the domain $\mathbb{Z}$. By using the same idea as we used for proving that temporal constraints have the small solution property, it follows that if an instance (V, C) of CSP(Γ) has a solution, then it has a solution $s : V \to \{1, \ldots, |V|\}$ and Γ has the small solution property. Let us now consider the constraint language $\Gamma \cup \mathbb{Z}_c$. We assume as usual that the constants in $\mathbb{Z}_c$ are represented by c written in binary. Let (V, C) denote an arbitrary instance of CSP($\Gamma \cup \mathbb{Z}_c$) and define $S = \{d \mid \{d\}(x) \in C\}$. Building on the same proof idea once again, one can easily verify that if (V, C) has a solution, then it has a solution $s : V \to \{(\min S) - |V|, \ldots, (\max S) + |V|\}$. Now, at most $r = \lceil \log_2(|\min S|) + 1 \rceil$ and $r' = \lceil \log_2(|\max S|) + 1 \rceil$ bits are needed to represent the numbers $\min S$ and $\max S$, respectively. This implies that $\Gamma \cup \mathbb{Z}_c$ has the small solution property since r and r' are smaller than $||(V, C)||$. We see that Γ is a partition scheme and $\Gamma \cup \mathbb{Z}_c$ satisfies precondition (2) of Lemma 30. We conclude that both CSP($\Theta \cup \mathbb{Z}_c$) and CSP($\Theta \cup \mathbb{Z}_f$) are members of NP whenever Θ is quantifier-free first-order definable in Γ.

6.2. An example based on RCC-5$_{\text{set}}$

We will now illustrate the small solution property with RCC-5$_{\text{set}}$. Henceforth, let $\mathcal{R}$ denote the RCC-5 basic relations interpreted as RCC-5$_{\text{set}}$ relations (see Figure 3 in Section 3). It is known that RCC-5$_{\text{set}}$ is not ω-categorical so the methods in Sections 4 and 5 are not applicable. As a warm-up, we give a simple way of proving this under the assumption that P $\neq$ NP. Let $\Gamma = \mathcal{R} \cup \{\neq\}$ where $\neq$ equals $\bigcup_{B \in \mathcal{R} \setminus \{\mathtt{EQ}\}} B$. It is known that CSP($\Gamma$) is in P [39, 59]. We extend Γ with one constant: $\Gamma' = \Gamma \cup \{\{0, 1, 2\}\}$. Consider the constraints $\{\mathtt{PP}(y, z), \{0, 1, 2\}(z)\}$. It is clear that if s is a solution, then

$$s(y) \in \{\{0\}, \{1\}, \{2\}, \{0, 1\}, \{0, 2\}, \{1, 2\}\}$$

so there are 6 distinct possible choices for the variable y. This implies that there is a straightforward polynomial-time reduction from 6-Colourability to CSP(Γ') (since the relation $\neq$ is in Γ') and, consequently, that CSP(Γ') is NP-complete. If Theorem 15 or Theorem 27 were applicable, then CSP(Γ') would be polynomial-time solvable.

Proving that a particular constraint language has the small solution property may be a non-trivial task; one may, for instance, think of the previously mentioned results concerning linear and integer programming. When studying qualitative constraint languages, one often encounters relations that only relates "smaller" objects with "larger" objects: obvious examples include the less-than relation $<$ or the subset relation $\subset$. Such relations are occasionally useful for inductively proving the small solution property. This idea is illustrated in the next lemma.

Lemma 32. *Let $D = 2^{\mathbb{N}} \setminus \{\varnothing\}$. The constraint language $\mathcal{R} \cup D_c$ has the small solution property.*

Proof. Let $I = (V, C)$ be a satisfiable instance of CSP($\mathcal{R} \cup D_c$) with solution $s : V \to 2^{\mathbb{N}} \setminus \{\varnothing\}$. Construct a new instance $I' = (V', C')$ as follows.

Step 1. Remove every $\mathtt{EQ}(x, y)$ constraint: this can be done by collapsing the variables x and y (we leave the obvious details of this step to the reader).

Step 2. Replace every $\mathtt{PP}^{\smile}(x, y)$ constraint with $\mathtt{PP}(y, x)$.

Step 3. Remove every $\mathtt{PO}(x, y)$ constraint by replacing it with

$$\begin{array}{lll} \mathtt{DR}(z_1, z_2) & \mathtt{DR}(z_2, z_3) & \mathtt{DR}(z_3, z_1) \\ \mathtt{PP}(z_1, x) & \mathtt{DR}(z_1, y) & \\ \mathtt{PP}(z_2, x) & \mathtt{PP}(z_2, y) & \\ \mathtt{PP}(z_3, y) & \mathtt{DR}(z_3, x) & \end{array}$$

where z_1, z_2, z_3 are fresh variables.

Note the following.

1. I' is a satisfiable instance of CSP($\mathcal{R} \cup D_c$),
2. the only non-unary relations that appear in I are $\mathtt{DR}$ and $\mathtt{PP}$, and
3. the size of V' is upper bounded by some polynomial q (that does not depend on (V, C)).

Fact 3 can be established as follows: fresh variables are only introduced in Step 3 where $\mathtt{PO}$ constraints are removed, and there are at most $O(|V|^2)$ such constraints since $\mathtt{PO}$ is a binary relation.

We say that two variables u, v in I' are $\mathtt{PP}$-*connected* if there exists a sequence of variables $w_1, \ldots, w_p$ such that

1. $w_1 = u$,
2. $w_p = v$, and
3. $\mathtt{PP}(w_i, w_{i+1}) \in C'$ for all $1 \leq i < p$.

Note that if u and v are PP-connected, then in any solution s' of I' we have that $(s'(u), s'(v)) \in \mathtt{PP}$.

Given a constant relation $U = \{\{a_1, \ldots, a_k\}\} \in D_c$, we let $sz(U) = k$. Now, let

$$T = \max\{sz(U) \mid U(x) \in C' \text{ and } U \in D_c\}.$$

If u is PP-connected with some variable v and $U(v) \in C'$, then we know that $|s'(u)| < T$ for any solution s' to I'.

We prove that at most $|V'| \cdot T$ different elements are needed for representing a solution by induction over the number of variables in V'. This implies the result by reasoning as follows: we can without loss of generality assume that the set of possible values is $2^{\{1,\ldots,|V'|\cdot T\}} \setminus \{\varnothing\}$. To represent such a value, i.e., a set, we need at most $|V'| \cdot T$ bits if we view each value as a bit vector where the i^{th} component equals 1 if and only if i is a member of the set. Hence, $\mathrm{CSP}(\mathcal{R} \cup D_c)$ has the small solution property since $|V'| \leq q(|V|) \leq q(||I||)$ and $T \leq ||I|| \leq q(||I||)$.

Basis step. If $|V'| = 1$ and $V' = \{v\}$, then either a singleton set is sufficient as a value for v (if v is not constrained by a unary relation) or a set with cardinality T is sufficient (otherwise).

Induction hypothesis. Assume the claim holds when $|V'| = p$.

Induction step. We show the claim when $|V'| = p + 1$. If there are variables $v, v' \in V'$ such that v is PP-connected to v' and v' is PP-connected to v, then (V', C') has no solution and this leads to a contradiction. Thus, we can choose a variable $v \in V'$ such that v is maximal with respect to PP-connectedness, i.e. v is not PP-connected to any other variable. Let I'' be the instance I' restricted to variable set $V \setminus \{v\}$. By the induction hypothesis, we need at most pT values for the instance I''. If there exists $U(v) \in C$, then we need at most T values for v which gives us at most $pT + T = (p + 1)T$ values in total. If there is no $U(v) \in C$, then we need at most one additional value for v so we need at most $pT + 1 \leq (p + 1)T$ values in total. To see this, v may (in the worst case) be PP-connected to every other variable and v must (by the induction hypothesis) contain at least pT different values. However, it must also be a strict superset of the other variables and this is accomplished by adding one fresh element. □

Theorem 33. *Let Γ be a finite set of relations that are quantifier-free definable in $\mathcal{R}$. Then, CSP($\Gamma \cup D_c$) and CSP($\Gamma \cup D_f$) are in NP.*

Proof. Combine Lemma 32 with Corollary 31. □

6.3. Discussion

The idea behind Lemma 32 can readily be extended to other classes of relations that are related to RCC-5 such as (certain variants of) *set relations* (cf. Bodirsky and Hils [10] and the references in their article), and it can also be generalised in other directions. An interesting observation is that the NP membership results for RCC-5 and RCC-8 in the plane by Li et al. [48] is implicitly

based on the small solution property. There, the representational size of the regions are analysed and bounded by exploiting a particular parameter that is related to embeddings of planar graphs in the plane. Another interesting observation is that Li [47] uses concepts that are similar to PP-connectedness when constructing different realisations of the RCC-8 formalism. This may indicate that the approach taken in the proof of Lemma 32 could quite easily be adapted to other spatial formalisms.

We conclude this section by a few observations concerning the small solution property. First of all, it is important to realise that the converse of Lemma 30 does not necessarily hold. Clearly, the function $\log(\mathrm{Tower}(n))$ grows faster than any polynomial in n. Now consider the constraint language $\Gamma = \{U_1, U_2, \dots\}$ where $U_i = \{x \in \mathbb{N} \mid x = \mathrm{Tower}(i)\}$. Checking if an instance of $\mathrm{CSP}(\Gamma)$ is satisfiable or not can trivially be solved in polynomial time if $U_1, U_2, \dots$ are represented in a reasonable way—for instance, if U_i is represented by the number i written in binary. Thus, $\mathrm{CSP}(\Gamma)$ is in NP, too. It is obvious, though, that Γ does not have the small solution property if we represent the natural numbers in binary.

Finally, we want to emphasise once again that the choice of exact interpretation and representation of relations and domain elements is extremely important. Recall the ω-categorical and homogeneous representation of RCC-5 that we denoted RCC-$5_{\omega-\mathrm{cat}}$. In this case, adding constants preserves computational complexity (up to polynomial-time reductions) by Theorem 27 (given that relations and domain elements are represented in a suitable way). Recall that adding a *finite* number of constant relations always preserve the complexity by Corollary 28. We know from an earlier example that adding even a single constant may not preserve the complexity of RCC-5_{set}. At the same time, we know from Proposition 13 that the CSPs for RCC-5_{set} and RCC-$5_{\omega-\mathrm{cat}}$ are the very same computational problem.

7. Conclusions

We have presented three different methods for analysing the complexity of qualitative CSPs extended with finite unary relations. Assume we have a constraint language Γ over domain D and we want to analyse the complexity of $\mathrm{CSP}(\Gamma \cup D_c)$ or $\mathrm{CSP}(\Gamma \cup D_f)$. Which method should we use? If Γ is ω-categorical, then methods I (model-complete cores) and II (homogeneity) should be considered first. If there is a way of efficiently computing orbit-defining formulas, then method I is typically the easiest method to use and it gives polynomial-time equivalence of $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Gamma \cup D_c)$. However, if $\mathcal{B} \subseteq \Gamma$ and Γ is quantifier-free definable in $\mathcal{B}$ for some partition scheme $\mathcal{B}$, then one should always check whether Γ is homogeneous or not. If so, one can apply method II and circumvent the need for computing orbit-defining formulas. Note, though, that one does not always get polynomial-time equivalence of $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Gamma \cup D_c)$ in this case. If Γ is not ω-categorical (or there are other problems in applying method I and/or II), then one has to resort to method III (small solutions).

We exemplify this approach by taking a closer look at the *cardinal relation algebra* (CRA) [29, 50]. Here, the domain elements are points in the plane and we have nine basic relations (N, NE, E, SE, S, SW, W, NW and the equality relation EQ) that describe cardinal directions. For instance, (x, x') SW (y, y') holds if and only if $x < x'$ and $y < y'$. Let $\mathcal{B}$ denote this set of relations over the set $\mathbb{Q}^2$. Li et al. [48] have shown that $\mathcal{B}^{\vee=}$ extended with constants and/or finite unary relations give rise to an NP-complete constraint satisfaction problem. We will now give a more fine-grained analysis in the case when we add constants to constraint languages. Hirsch [32, Corollary 8] has shown that CRA is ω-categorical so method I is in principle applicable. However, given a constraint language $\Gamma \subseteq \mathcal{B}^{\vee=}$, we do not know right away whether it is a model-complete core or not, and we do not know how to compute orbit-defining formulas. This can probably be worked out quite easily since CRA is closely related to the point algebra PA. A simpler way, though, is to note that Hirsch [32, Theorem 1] has proved that CRA is homogeneous. If we combine this with the fact that CRA is a partition scheme, we can easily apply method II and conclude the following: if $\mathcal{B} \subseteq \Gamma \subseteq \mathcal{B}^{\vee=}$, then $\mathrm{CSP}(\Gamma)$ and $\mathrm{CSP}(\Gamma \cup (\mathbb{Q}^2)_c)$ are polynomial-time equivalent problems.

Methods I and II are based on exploiting model-theoretical properties of the underlying constraint languages. While methods based on model theory and universal algebra have been very common when studying CSPs from the viewpoint of theoretical computer science [3, 13, 20], such methods have been less popular within the AI community (with some notable exceptions such as Hirsch [32] and Huang [35]). Thus, we take the opportunity to discuss these methods in slightly more detail.

Method I. The main obstacle for applying method I is the need for computing orbit-defining formulas efficiently. In fact, it is not even known if this problem is decidable or not in the general case. Studying this problem is a very important future research direction. In cases where we do not know how to efficiently generate orbit-defining formulas, there are (at least) two possible workarounds. We have already encountered the first workaround in Corollary 16: the restriction to finite sets of constant relations. Another workaround is to sacrifice polynomial-time equivalence and allow more time for computing the orbit-defining formula. If the problem at hand is NP-hard, then a (preferably mildly) exponential algorithm can be acceptable. In both cases, algorithmic methods for generating orbit-defining formulas would be helpful. We note, on the positive side, that related definability problems have recently been successfully addressed, cf. Bodirsky et al. [15]. Their methods are interesting since they combine methods taken from universal algebra, Ramsey theory, and topological dynamics.

Method II. Given a structure Γ, it may be difficult to verify that it is indeed homogeneous. Here, one should note that if Γ contains a finite number of relations, the domain of Γ is countably infinite, and Γ is homogeneous, then Γ is ω-categorical, cf. Macpherson [53]. This explains why one should always check ω-categoricity first, and this can quite often be accomplished by using Theorem 11. If Γ is ω-categorical, then Γ is homogeneous *if and only if* every

formula in Th(Γ) is equivalent to a quantifier-free formula (see, for instance, Macpherson [53]). This gives an alternative way of proving homogeneity than using the automorphism-based definition directly. This also clarifies the connections between method I and method II: recall that Γ is model-complete if and only if every formula in Th(Γ) is equivalent to an existential formula.

Another approach for using homogeneity is to construct suitable homogeneous structures "from scratch". The main tool for this is *Fraïssé amalgamation*. The details are outside the scope of this article: Macpherson [53] outlines the approach and concrete constructions for RCC-5 and RCC-8 can be found in Bodirsky & Chen [7] and Bodirsky & Wölfl [16], respectively. One should note that amalgamation is quite common in the literature on CSPs and related problems; however, it is often referred to as the *patchwork property* [35, 52, 61].

Acknowledgements

We thank Meysam Aghighi, Manuel Bodirsky, Fredrik Heintz, Johan Thapper, and the anonymous reviewers for providing valuable input to this article. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

[1] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[2] Libor Barto. The dichotomy for conservative constraint satisfaction problems revisited. In *Proc. 26th Annual IEEE Symposium on Logic in Computer Science (LICS-2011)*, pages 301–310, 2011.

[3] Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *J. ACM*, 61(1):3:1–3:19, 2014.

[4] Bruce L. Bauslaugh. Core-like properties of infinite graphs and structures. *Discrete Mathematics*, 138(1-3):101–111, 1995.

[5] Manuel Bodirsky. Cores of countably categorical structures. *Logical Methods in Computer Science*, 3(1):1–16, 2007.

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

[7] Manuel Bodirsky and Hubie Chen. Qualitative temporal and spatial reasoning revisited. *J. Log. Comput.*, 19(6):1359–1383, 2009.

[8] Manuel Bodirsky and Víctor Dalmau. Datalog and constraint satisfaction with infinite templates. *J. Comput. Syst. Sci.*, 79(1):79–100, 2013.

[9] Manuel Bodirsky and Martin Grohe. Non-dichotomies in constraint satisfaction complexity. In *Proc. 35th International Colloquium on Automata, Languages and Programming (ICALP-2008)*, pages 184–196, 2008.

[10] Manuel Bodirsky and Martin Hils. Tractable set constraints. *J. Artif. Intell. Res. (JAIR)*, 45:731–759, 2012.

[11] Manuel Bodirsky and Peter Jonsson. A model-theoretic view on qualitative constraint reasoning. *J. Artif. Intell. Res. (JAIR)*, 58:339–385, 2017.

[12] Manuel Bodirsky, Peter Jonsson, and Trung Van Pham. The reducts of the homogeneous binary branching C-relation. *Journal of Symbolic Logic*, 81(4):1255–1297, 2016.

[13] Manuel Bodirsky and Jan Kára. The complexity of temporal constraint satisfaction problems. *J. ACM*, 57(2), 2010.

[14] Manuel Bodirsky and Jan Kára. A fast algorithm and datalog inexpressibility for temporal reasoning. *ACM Trans. Comput. Log.*, 11(3):15:1–15:21, 2010.

[15] Manuel Bodirsky, Michael Pinsker, and Todor Tsankov. Decidability of definability. *J. Symb. Log.*, 78(4):1036–1054, 2013.

[16] Manuel Bodirsky and Stefan Wölfl. RCC8 is polynomial on networks of bounded treewidth. In *Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI-2011)*, pages 756–761, 2011.

[17] Andrei Bulatov. Tractable conservative constraint satisfaction problems. In *Proc. 18th annual IEEE symposium on logic in computer science (LICS-2003)*, pages 321–330, 2003.

[18] Andrei Bulatov. Conservative constraint satisfaction re-revisited. *Journal of Computer and System Sciences*, 82(2):347–356, 2016.

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

[20] Andrei Bulatov, Peter Jeavons, and Andrei Krokhin. Classifying the computational complexity of constraints using finite algebras. *SIAM J. Comput.*, 34(3):720–742, 2005.

[21] Peter J. Cameron. *Oligomorphic permutation groups*. Cambridge University Press, Cambridge, 1990.

[22] Georg Cantor. Über unendliche, lineare Punktmannigfaltigkeiten. *Mathematische Annalen*, 23:453–488, 1884.

[23] Clément Carbonnel. The dichotomy for conservative constraint satisfaction is polynomially decidable. In *Proc. 22nd International Conference on Principles and Practice of Constraint Programming (CP-2016)*, pages 130–146, 2016.

[24] David Cohen, Peter Jeavons, Peter Jonsson, and Manolis Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47(5):826–853, 2000.

[25] Anthony G. Cohn and Jochen Renz. Qualitative spatial representation and reasoning. In *Handbook of Knowledge Representation*, pages 551–596. Elsevier, 2008.

[26] Daniel de Leng and Fredrik Heintz. Qualitative spatio-temporal stream reasoning with unobservable intertemporal spatial relations using landmarks. In *Proc. 30th AAAI Conference on Artificial Intelligence (AAAI-2016)*, pages 957–963, 2016.

[27] Ivo Düntsch, Hui Wang, and Stephen McCloskey. A relation-algebraic approach to the region connection calculus. *Theor. Comput. Sci.*, 255(1-2):63–83, 2001.

[28] Frank Dylla, Jae Hee Lee, Till Mossakowski, Thomas Schneider, André van Delden, Jasper van de Ven, and Diedrich Wolter. A survey of qualitative spatial and temporal calculi: Algebraic and computational properties. *ACM Comput. Surv.*, 50(1):7:1–7:39, 2017.

[29] Andrew U. Frank. Qualitative spatial reasoning with cardinal directions. In *Proc. 7th Austrian Conference on Artificial Intelligence (ÖGAI-91)*, pages 157–167, 1991.

[30] Stella Giannakopoulou, Charalampos Nikolaou, and Manolis Koubarakis. A reasoner for the RCC-5 and RCC-8 calculi extended with constants. In *Proc. 28th AAAI Conference on Artificial Intelligence (AAAI-2014)*, pages 2659–2665, 2014.

[31] Pavol Hell and Jaroslav Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, Oxford, 2004.

[32] Robin Hirsch. Relation algebras of intervals. *Artif. Intell.*, 83(2):267–295, 1996.

[33] Robin Hirsch. Expressive power and complexity in algebraic logic. *Journal of Logic and Computation*, 7(3):309–351, 1997.

[34] Wilfrid Hodges. *Model theory*. Cambridge University Press, 1993.

[35] Jinbo Huang. Compactness and its implications for qualitative spatial and temporal reasoning. In *Proc. 13th International Conference on Knowledge Representation and Reasoning (KR-2012)*, 2012.

[36] Peter Jeavons. On the algebraic structure of combinatorial problems. *Theoret. Comput. Sci.*, 200(1–2):185–204, 1998.

[37] Peter Jonsson. Finite unary relations and qualitative constraint satisfaction. In *Proc. 22nd European Conference on Artificial Intelligence (ECAI-2016)*, pages 37–45, 2016.

[38] Peter Jonsson and Christer Bäckström. A unifying approach to temporal constraint reasoning. *Artif. Intell.*, 102(1):143–155, 1998.

[39] Peter Jonsson and Thomas Drakengren. A complete classification of tractability in RCC-5. *J. Artif. Intell. Res.*, 6:211–221, 1997.

[40] Peter Jonsson and Victor Lagerkvist. Upper and lower bounds on the time complexity of infinite-domain CSPs. In *Proc. 21st International Conference on Principles and Practice of Constraint Programming (CP-2015)*, pages 183–199, 2015.

[41] Markus Junker and Martin Ziegler. The 116 reducts of $(\mathbb{Q}, <, a)$. *J. Symb. Log.*, 73(3):861–884, 2008.

[42] Michael Kompatscher and Trung Van Pham. A complexity dichotomy for poset constraint satisfaction. In *Proc. 34th Symposium on Theoretical Aspects of Computer Science, (STACS-2017)*, pages 47:1–47:12, 2017.

[43] Manolis Koubarakis. Tractable disjunctions of linear constraints: Basic results and applications to temporal reasoning. *Theoret. Comput. Sci.*, 266(1–2):311–339, 2001.

[44] Arne Kreutzmann and Diedrich Wolter. Qualitative spatial and temporal reasoning with AND/OR linear programming. In *Proc. 21st European Conference on Artificial Intelligence (ECAI-2014)*, pages 495–500, 2014.

[45] Andrei Krokhin, Peter Jeavons, and Peter Jonsson. Reasoning about temporal relations: The tractable subclasses of Allen's interval algebra. *J. ACM*, 50(5):591–640, 2003.

[46] C. Langford. Some theorems on deducibility. *Annals of Mathematics*, 28:16–40, 1927.

[47] Sanjiang Li. On topological consistency and realization. *Constraints*, 11(1):31–51, 2006.

[48] Sanjiang Li, Weiming Liu, and Sheng-sheng Wang. Qualitative constraint satisfaction problems: An extended framework with landmarks. *Artif. Intell.*, 201:32–58, 2013.

[49] Sanjiang Li and Mingsheng Ying. Extensionality of the RCC8 composition table. *Fundam. Inform.*, 55(3-4):363–385, 2003.

[50] Gérard Ligozat. Reasoning about cardinal directions. *J. Vis. Lang. Comput.*, 9(1):23–44, 1998.

[51] Gérard Ligozat and Jochen 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.

[52] Carsten Lutz and Maja Milicic. A tableau algorithm for description logics with concrete domains and general Tboxes. *J. Autom. Reasoning*, 38(1-3):227–259, 2007.

[53] Dugald Macpherson. A survey of homogeneous structures. *Discrete Mathematics*, 311(15):1599–1634, 2011.

[54] David Marker. *Model Theory: An Introduction*. Springer, 2002.

[55] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *J. ACM*, 42(1):43–66, 1995.

[56] Charalampos Nikolaou and Manolis Koubarakis. Querying incomplete information in RDF with SPARQL. *Artif. Intell.*, 237:138–171, 2016.

[57] Christos Papadimitriou. On the complexity of integer programming. *J. ACM*, 28(4):765–768, 1981.

[58] David A. Randell, Zhan Cui, and Anthony G. Cohn. A spatial logic based on regions and connection. In *Proc. 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-1992)*, pages 165–176, 1992.

[59] Jochen Renz and Bernhard 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.

[60] Jochen Renz and Bernhard 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.

[61] Michael Sioutis and Manolis Koubarakis. Consistency of chordal RCC-8 networks. In *Proc. 24th International Conference on Tools with Artificial Intelligence (ICTAI-2012)*, pages 436–443, 2012.

[62] Matthias Westphal, Julien Hué, and Stefan Wölfl. On the scope of qualitative constraint calculi. In *Proc. 37th Annual German Conference on AI (KI-2014)*, pages 207–218, 2014.

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