Solving Infinite-Domain CSPs Using the Patchwork Property

Konrad K. Dabrowski,[1] Peter Jonsson,[2] Sebastian Ordyniak,[3] George Osipov[2]
[1]Durham University
[2]Linköping University
[3]University of Leeds
konrad.dabrowski@durham.ac.uk, {peter.jonsson,george.osipov}@liu.se, sordyniak@gmail.com

Abstract

The constraint satisfaction problem (CSP) has important applications in computer science and AI. In particular, infinite-domain CSPs have been intensively used in subareas of AI such as spatio-temporal reasoning. Since constraint satisfaction is a computationally hard problem, much work has been devoted to identifying restricted problems that are efficiently solvable. One way of doing this is to restrict the interactions of variables and constraints, and a highly successful approach is to bound the treewidth of the underlying primal graph. Bodirsky & Dalmau [*J. Comput. System. Sci.* 79(1), 2013] and Huang et al. [*Artif. Intell.* 195, 2013] proved that CSP(Γ) can be solved in $n^{f(w)}$ time (where n is the size of the instance, w is the treewidth of the primal graph and f is a computable function) for certain classes of constraint languages Γ. We improve this bound to $f(w) \cdot n^{O(1)}$, where the function f only depends on the language Γ, for CSPs whose basic relations have the patchwork property. Hence, such problems are fixed-parameter tractable and our algorithm is asymptotically faster than the previous ones. Additionally, our approach is not restricted to binary constraints, so it is applicable to a strictly larger class of problems than that of Huang et al. However, there exist natural problems that are covered by Bodirsky & Dalmau's algorithm but not by ours.

Introduction

The constraint satisfaction problem over a constraint language Γ (CSP(Γ)) is the problem of finding a variable assignment which satisfies a set of constraints, where each constraint is constructed from a relation in Γ. This problem can be used to model many problems encountered in computer science and AI, see e.g. Rossi et al. (2006) or Dechter (2003). The CSP is computationally hard in the general case; if the variable domains are finite, then the problem is NP-complete, and otherwise it may be of arbitrarily high complexity (Bodirsky and Grohe 2008). Hence, identifying tractable problems is of great practical interest.

Tractable fragments have historically been identified using two different methods: either (1) restrict the relations that are allowed in the constraint language or (2) restrict how variables and constraints interact in problem instances. We focus on the second kind of restrictions in this paper; these are referred to as *structural restrictions*. One way of studying structural restrictions is via the *primal graph*: this graph has the variables as its vertices with two of them joined by an edge if they occur together in the scope of a constraint. The graph parameter treewidth (Bertelé and Brioschi 1972; Robertson and Seymour 1984) is very useful in this context since many NP-hard graph problems are tractable on instances with bounded treewidth. The treewidth of the primal graph has been extensively used in the study of finite-domain CSPs. It is known that the problem is *fixed-parameter tractable* (fpt), i.e. it can be solved in $f(w+d) \cdot n^{O(1)}$ time, where n is the size of the instance, w is the treewidth of the primal graph, d is the domain size, and f is some computable function. This was proven by Gottlob et al. (2002); also see Samer & Szeider (2010) for a more general treatment.

Let us now consider infinite-domain CSPs. For certain classes of constraint languages, Bodirsky & Dalmau (2013, Corollary 1) proved that CSP(Γ) can be solved in $n^{O(w)}$ time (where the exact expression in the $O(w)$ term may depend on the constraint language) while Huang et al. (2013, Theorem 6) proved the bound $O(w^3 n \cdot \mathrm{e}^{w^2 \log n}) = n^{O(w^2)}$. These results do not prove fixed-parameter tractability but the weaker property of membership in the complexity class XP. Algorithms with a running time bounded by $n^{f(w)}$ are obviously polynomial-time when w is fixed. However, since w appears in the exponent, such algorithms become impractical (even for small w) when large instances are considered. It is significantly better if a problem is fpt and can be solved in time $f(w) \cdot n^{O(1)}$ since the order of the polynomial in n does not depend at all on w.

Our main result is an fpt algorithm for CSPs where the underlying basic relations have the *patchwork property* (Lutz and Miličić 2007). Several important CSPs (such as Allen's algebra and RCC8) are known to have this property. The patchwork property ensures that the union of two satisfiable CSPs, whose constraints agree on their common variables, is also satisfiable. With the discussion above in mind, it is clear that our algorithm has better computational properties than the two previous ones. We will now briefly compare the applicability of the algorithms; more information on this can be found in the discussion section. Bodirsky & Dalmau's algorithm (BD) works for constraint languages that are ω-

categorical (in fact, it works even for languages where only the core is ω-categorical) while Huang, Li & Renz's algorithm (HLR) works for languages with binary relations that have the *atomic network amalgamation property* (aNAP). Our algorithm has a wider applicability than HLR since aNAP implies the patchwork property and our algorithm is not restricted to binary relations. The relation to BD is more complex since there are problems that are covered by BD but not by our algorithm (an example is the *branching time algebra*—the details are discussed later on). The exact dividing line is unfortunately unclear.

The rest of this paper is divided into two distinct parts. In the first part, we present our main algorithm (which is based on dynamic programming) and prove that it achieves the required time bound. In the second part, we analyse the applicability of our algorithm. Even though the patchwork property is well known within the CSP community, there are not that many formalisms that have been proven to have this property. By using certain model-theoretical concepts, we obtain an alternative way of identifying constraint languages with the patchwork property. Based on this, we demonstrate how to apply our results on constraint languages that are definable in $(\mathbb{Q}; <)$ (with applications in, for instance, temporal reasoning and scheduling) and phylogeny languages (which are useful in bioinformatics).

Preliminaries

In this section we introduce the necessary prerequisites.

Logic

A *(relational) signature* τ is a set of symbols, each with an associated natural number called its *arity*. A *(relational)* τ*-structure* $\mathbf{A}$ consists of a set D (the domain) together with relations $R^{\mathbf{A}} \subseteq D^k$ for each k-ary symbol $R \in \tau$. A structure is *countable* if its domain is a countable set.

Let $\mathbf{A}$ be a τ-structure over a domain D. We say that $\mathbf{A}$ is k*-ary* if every relation in $\mathbf{A}$ has arity k. The relations in a k-ary $\mathbf{A}$ are *jointly exhaustive* (JE) if $\bigcup_{R \in \mathbf{A}} R = D^k$. They are *pairwise disjoint* (PD) if $R \cap R' = \varnothing$ for all distinct $R, R' \in \mathbf{A}$.

First-order formulas ϕ over $\mathbf{A}$ (or, for short, $\mathbf{A}$-formulas) are defined using the logical symbols of universal and existential quantification, disjunction, conjunction, negation, equality, bracketing, variable symbols, the relation symbols from τ, and the symbol $\bot$ for the truth-value false. First-order formulas over $\mathbf{A}$ can be used for defining relations: for a formula $\phi(x_1, \ldots, x_k)$ with free variables $x_1, \ldots, x_k$, the corresponding relation R is the set of all k-tuples $(t_1, \ldots, t_k) \in D^k$ such that $\phi(t_1, \ldots, t_k)$ is true in $\mathbf{A}$. In this case we say that R is *first-order definable* in $\mathbf{A}$. Our definitions are always parameter-free, i.e. we do not allow the use of domain elements within them. We may assume without loss of generality that all formulas defining relations are in disjunctive normal form (DNF). A formula is in DNF if it is a disjunction of one or more conjunctions of one or more *atomic formulas* of the type $R(\bar{x})$ or $\neg R(\bar{x})$, where $R \in \mathbf{A} \cup \{=\}$ and $\bar{x}$ is a sequence of variables. The conjunctions of atomic formulas are referred to as *clauses*.

The most common way of using JEPD relations in AI-relevant CSPs is via the constraint language $\mathbf{A}^{\vee=}$, where $\mathbf{A}$ is a k-ary structure. The set $\mathbf{A}^{\vee=}$ contains the unions of all subsets of $\mathbf{A}$. Equivalently, $R \in \mathbf{A}^{\vee=}$ if R can be written as a disjunction $R_1(\bar{x}) \vee \cdots \vee R_p(\bar{x})$ where $R_1, \ldots, R_p \in \mathbf{A}$ and $\bar{x} = (x_1, \ldots, x_k)$. Since we want to study more expressive sets of relations, we let $\langle \mathbf{A} \rangle_{\mathrm{b}}$ denote the set of relations that are definable by quantifier-free formulas that *only* contain the relations in $\mathbf{A}$, i.e. using the equality relation $=$ is not allowed unless it is a member of $\mathbf{A}$. Note that $\mathbf{A}^{\vee=} \subsetneq \langle \mathbf{A} \rangle_{\mathrm{b}}$, so $\langle \mathbf{A} \rangle_{\mathrm{b}}$ strictly generalises $\mathbf{A}^{\vee=}$. If the set of relations in $\mathbf{A}$ is finite and JEPD, then we may assume that all formulas are negation-free: any negated relation can be replaced by the disjunction of all other relations.

Constraint Satisfaction

Let $\mathbf{A}$ be a τ-structure with domain D. The *constraint satisfaction problem* over $\mathbf{A}$ (CSP($\mathbf{A}$)) is defined as follows:

INSTANCE: A set V of variables and a set C of *constraints* of the form $R(v_1, \ldots, v_r)$, where $R \in \mathbf{A}$ is a relation of arity r, and $v_1, \ldots, v_r \in V$

QUESTION: Is there an assignment $f : V \to D$ such that $(f(v_1), \ldots, f(v_r)) \in R$ for every $R(v_1, \ldots, v_r) \in C$?

The structure $\mathbf{A}$ is referred to as the *constraint language*. Let $\mathbf{A}$ be a finite k-ary constraint language with JEPD relations. Consider a finite $\mathbf{\Gamma} \subseteq \langle \mathbf{A} \rangle_{\mathrm{b}}$ and let $\mathcal{I} = (V, C)$ be an instance of CSP($\mathbf{\Gamma}$). Recall that every constraint in C can be defined by a negation-free DNF $\mathbf{A}$-formula, where every clause is a conjunction of atomic constraints of the form $R(\bar{v})$ with $R \in \mathbf{A}$. Note that any assignment satisfying a constraint in C must satisfy all atomic constraints in at least one clause of the defining DNF $\mathbf{A}$-formula. A *certificate* for $\mathcal{I}$ is a satisfiable instance $\mathcal{C} = (V, C')$ of CSP($\mathbf{A}$) that *implies* every constraint in C, i.e. for every constraint in C, there is a clause in the corresponding DNF $\mathbf{A}$-formula such that all atomic constraints in this clause are in C'.

Proposition 1. *An instance of CSP*($\mathbf{\Gamma}$) *admits a certificate if and only if it is satisfiable.*

Now assume CSP($\mathbf{A}$) is decidable. Then, one can verify whether an instance (V, C') of CSP($\mathbf{A}$) is a certificate for an instance (V, C) of CSP($\mathbf{\Gamma}$): first, check that (V, C') is satisfiable, and then, for all $R(v_1, \ldots, v_r) \in C$, verify that (V, C') implies $R(v_1, \ldots, v_r)$ by considering every clause in the definition of R and checking if it is included in C'. Note that the length of the DNF formula defining R depends only on the arity of R and $|\mathbf{A}|$, which are both bounded by constants since $\mathbf{\Gamma}$ and $\mathbf{A}$ are finite languages. Thus, if CSP($\mathbf{A}$) is solvable in polynomial time, then the certificate test can also be carried out in polynomial time.

An instance of CSP($\mathbf{A}$) is *complete* if it contains a constraint over every k-tuple of (not necessarily distinct) variables. A certificate is complete if it is a complete instance of CSP($\mathbf{A}$). Since the relations in $\mathbf{A}$ are JE, any certificate can be extended to a complete one. Thus, we can assume that all certificates are complete.

For any instance $\mathcal{I} = (V, C)$ of CSP and any set of variables $U \subseteq V$, define $C[U] \subseteq C$ to include all constraints whose scope is in U. We say that $\mathcal{I}[U] = (U, C[U])$ is the

subinstance of $\mathcal{I}$ induced by U. We will also say that $\mathcal{I}[U]$ is obtained by *projecting* $\mathcal{I}$ *onto* U. Properties of certificates (including completeness) are preserved under projections. We formalise this observation below.

Proposition 2. *If $\mathcal{C}$ is a certificate for $\mathcal{I} = (V, C)$, then $\mathcal{C}[U]$ is a certificate for $\mathcal{I}[U]$ for all $U \subseteq V$. If $\mathcal{C}$ is complete, then $\mathcal{C}[U]$ is also complete.*

Parameterized Complexity

In parameterized algorithmics (Flum and Grohe 2006; Niedermeier 2006; Downey and Fellows 2013) the runtime of an algorithm is studied with respect to a parameter $p \in \mathbb{N}$ and input size n. The idea is to find a parameter that describes the structure of the instance such that the combinatorial explosion can be confined to this parameter. In this respect, the most favourable complexity class is FPT (*fixed-parameter tractable*), which contains all problems that can be decided by an algorithm running in $f(p) \cdot n^{O(1)}$ time, where f is a computable function. Problems that can be solved in this time are said to be *fixed-parameter tractable* (fpt). A more general class XP contains all problems decidable in $n^{f(p)}$ time, i.e. the problems solvable in polynomial time when the parameter p is bounded. Clearly, FPT $\subseteq$ XP. Moreover, the inclusion is strict (see e.g. (Flum and Grohe 2006)).

We will concentrate on one well-known parameter in this paper: the *treewidth* of the *primal graph*. Thus, if we state that some problem is fpt, then we always mean with respect to this parameter. The primal graph of an instance of CSP is the undirected graph whose vertices coincide with the variables of the instance, and where two vertices are joined by an edge if they occur in the scope of the same constraint. Treewidth is based on *tree decompositions*: a tree decomposition (T, X) of an undirected graph $G = (V, E)$ consists of a rooted tree T and a mapping X from the nodes of T to the subsets of V. The subsets $X(t)$ are called *bags*. T_t stands for the subtree rooted at t, while V_t is the set of all variables occurring in the bags of T_t, i.e. $V_t = \bigcup_{s \in T_t} X(s)$. A tree decomposition fulfils the following properties: for every $u, v \in V$ that are adjacent in G, there is a node $t \in V(T)$ such that $u, v \in X(t)$, and for every $v \in V$, the set of bags of T containing v forms a non-empty sub-tree of T.

The width of a tree decomposition T is defined as $\max\{|X(t)| : t \in T\} - 1$. The treewidth of a graph G is the minimum width of a tree decomposition of G. It is NP-complete to determine if a graph has treewidth at most k (Arnborg, Corneil, and Proskurowski 1987), but when k is fixed, the graphs with treewidth k can be recognized and corresponding tree decompositions can be constructed in linear time (Bodlaender 1996).

Qualitative Spatial and Temporal Reasoning

We will consider several well-known formalisms for qualitative spatial and temporal reasoning. All of them can be defined as $\mathbf{B}^{\vee=}$ via a binary constraint language $\mathbf{B}$ with JEPD relations. It is important to note that the exact choice of relations for representing a reasoning problem as a CSP may be crucial. This is most easily illustrated with the RCC5 formalism that is introduced in item 4 below. RCC5 can be represented with structures $\mathbf{A}$ and $\mathbf{B}$ such that CSP($\mathbf{A}$) is the same computational problem as CSP($\mathbf{B}$) while $\mathbf{A}$ and $\mathbf{B}$ are very different from a model-theoretical point of view. This is discussed in some detail for RCC5 in (Bodirsky and Jonsson 2017, Sec. 2.5.2). It is also discussed that there are $\mathbf{A}$ and $\mathbf{B}$ that look like suitable representations of RCC5 (for instance, by having the "right" composition tables) but have different CSPs. For RCC5 and RCC8, we will thus exclusively use the representations suggested by Bodirsky & Wölfl (2011), whose CSP coincides with the standard interpretation of RCC relations.

Basic relation		Example	Endpoints
I precedes J	p	iii jjj	$I^+ < J^-$
J preceded by I	pi		
I meets J	m	iiii jjjj	$I^+ = J^-$
J met-by I	mi		
I overlaps J	o	iiii jjjj	$I^- < J^- < I^+$, $I^+ < J^+$
J overl.-by I	oi		
I during J	d	iii jjjjjjj	$I^- > J^-$, $I^+ < J^+$
J includes I	di		
I starts J	s	iii jjjjjjj	$I^- = J^-$, $I^+ < J^+$
J started by I	si		
I finishes J	f	iii jjjjjjj	$I^+ = J^+$, $I^- > J^-$
J finished by I	fi		
I equals J	e	iiii jjjj	$I^- = J^-$, $I^+ = J^+$

Table 1: Thirteen basic relations in Allen's Interval Algebra. The endpoint relations $I^- < I^+$ and $J^- < J^+$ that are valid for all relations have been omitted.

The choice of representation for the formalisms in 1–3 is, fortunately, much easier: the natural representations via concrete objects in $\mathbb{Q}^d$ have proven to capture the intended computational problems and at the same time having advantageous model-theoretical properties. We will consequently use these representations throughout the article.

Allen's Interval Algebra (IA) (Allen 1983) is a temporal reasoning formalism where one considers relations between intervals of the form $I = [I^-, I^+]$, where $I^-, I^+ \in \mathbb{Q}$ are the start and end points, respectively. The language $\mathbf{B}_{\mathrm{IA}}$ consists of thirteen basic relations illustrated in Table 1.

The d-dimensional *Block Algebra* (BA$_d$) (Balbiani, Condotta, and del Cerro 1998) is a generalization of IA to d-dimensional boxes with sides parallel to the coordinate axes. The relations in $\mathbf{B}_{\mathrm{BA}}$ are d-tuples of IA relations, each one applied in the corresponding dimension.

The *Cardinal Direction Calculus* (CDC) (Ligozat 1998) is a formalism for spatial reasoning with points on the plane as the basic objects. The relations in $\mathbf{B}_{\mathrm{CDC}}$ correspond to eight cardinal directions (North, East, South, West, and four intermediate ones) plus the equality relation. They can be viewed as pairs (R_1, R_2) for all choices of $R_1, R_2 \in \{<, =, >\}$, where each relation applies to the corresponding coordinate.

The *Region Connection Calculus* (RCC8) (Randell, Cui, and Cohn 1992) is a spatial reasoning formalism where the objects are non-empty regular closed subsets of a topological space. There are eight relations in $\mathbf{B}_{\mathrm{RCC8}}$: EQ (equal),

PO (partial overlap), DC (disconnected), EC (externally connected), NTPP (non-tangential proper part), its converse NTPP^{-1}, TPP (tangential proper part) and its converse TPP^{-1}. RCC5 is a simplified formalism without tangency, i.e. NTPP and TPP cannot be distinguished, and neither can EC and DC.

The Main Algorithm

We utilize the following CSP property in our algorithm:

Definition 3 (Lutz and Miličić (2007))**.** *A JEPD constraint language* $\mathbf{B}$ *has the* patchwork property (PP) *if, for every pair of complete satisfiable instances* $\mathcal{I}_1 = (V_1, C_1)$ *and* $\mathcal{I}_2 = (V_2, C_2)$ *of CSP*$(\mathbf{B})$ *such that* $\mathcal{I}_1[V_1 \cap V_2] = \mathcal{I}_2[V_1 \cap V_2]$*, the instance* $(V_1 \cup V_2, C_1 \cup C_2)$ *is also satisfiable.*

We want to underline the importance of the completeness condition in the previous definition: for example, consider the JEPD constraint language $\{<, =, >\}$ with domain $\mathbb{Q}$ and the two satisfiable incomplete instances $(\{a, x, b\}, \{a < x, x < b\})$ and $(\{a, y, b\}, \{a > y, y > b\})$. The intersection of these instances contains no constraints, so it is trivially satisfiable. However, their union is not satisfiable since the constraints imply that $a < b$ and $a > b$ hold simultaneously.

Many formalisms for qualitative spatial and temporal reasoning are known to have the patchwork property. For example, the basic relations of Allen's Interval Algebra, the Block Algebra, and the Cardinal Direction Calculus have the patchwork property (Lutz and Miličić 2007; Huang 2012), assuming that the representations that were discussed earlier are used. The picture is more complex for RCC8 and RCC5. Lutz and Miličić (2007) show that RCC8 restricted to the real plane has the PP, and Huang (2012) points out that this result can be lifted to the multi-dimensional case via Bodirsky & Wölfl's (2011) representation. Baader & Rydval (2020) also point this out in a more general setting; we will come back to their results later. The following is a direct consequence of the patchwork property:

Lemma 4. *Let* $\mathbf{B}$ *be a finite set of JEPD relations with the patchwork property and assume that* $\mathbf{\Gamma} \subseteq \langle \mathbf{B} \rangle_{\mathrm{b}}$*. For any two satisfiable instances* $\mathcal{I}_1 = (V_1, C_1)$ *and* $\mathcal{I}_2 = (V_2, C_2)$ *of CSP*$(\mathbf{\Gamma})$ *admitting certificates* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *such that* $\mathcal{C}_1[V_1 \cap V_2] = \mathcal{C}_2[V_1 \cap V_2]$*, the instance* $(V_1 \cup V_2, C_1 \cup C_2)$ *is also satisfiable.*

To simplify the presentation of the algorithm, we will use a particular kind of tree decomposition. A tree decomposition is *nice* if it fulfils the following properties: $X(r) = \varnothing$ and $X(\ell) = \varnothing$ for the root r and all leaf nodes ℓ in T, and every non-leaf node in T is one of the following three types: an *introduce node* t with exactly one child t' such that $X(t) = X(t') \cup \{v\}$ for some $v \in V$; a *forget node* t with exactly one child t' such that $X(t) = X(t') \setminus \{w\}$ for some $w \in V$, or a *join node* t with exactly two children t_1 and t_2 such that $X(t) = X(t_1) = X(t_2)$.

Given a tree decomposition T of a graph $G = (V, E)$, one can construct a nice tree decomposition of the same width and $O(n)$ nodes in linear time (Bodlaender and Kloks 1996).

Theorem 5. *Let* $\mathbf{A}$ *be a finite* k*-ary constraint language with JEPD relations and the patchwork property. Assume CSP*$(\mathbf{A})$ *is decidable. For any finite constraint language* $\mathbf{\Gamma} \subseteq \langle \mathbf{A} \rangle_{\mathrm{b}}$*, CSP*$(\mathbf{\Gamma})$ *is fpt.*

Proof. Let $\mathcal{I} = (V, C)$ be an instance of CSP$(\mathbf{\Gamma})$ and assume (T, X) is a nice tree decomposition of its the primal graph. The algorithm works as follows: for every node $t \in T$, we compute the set $R(t)$ consisting of all certificates for $\mathcal{I}[V_t]$ projected onto $X(t)$. Clearly, $\mathcal{I}$ is satisfiable if and only if $R(r) \neq \varnothing$, where r is the root of T. We compute $R(t)$ using dynamic programming from the leaves upwards, i.e. a node is processed only if all its children have already been processed.

To start, we set $R(\ell) = \{(\varnothing, \varnothing)\}$ for all leaf nodes $\ell \in T$. Since the decomposition is nice, we only need to consider three cases. If t is an introduce node with a child t', we enumerate certificates $\mathcal{C}$ for $\mathcal{I}[X(t)]$ and add $\mathcal{C}$ to $R(t)$ if $\mathcal{C}[X(t')]$ is in $R(t')$. If t forgets a variable w and has a child t', then $R(t)$ is obtained by enumerating certificates in $R(t')$ and removing w together with all constraints involving it from the certificate. Finally, if t joins nodes t_1 and t_2, then set $R(t) = R(t_1) \cap R(t_2)$.

Towards showing correctness, we prove the following:

Claim 1. *$\mathcal{C}$ is a certificate for $\mathcal{I}[V_t]$ if and only if $\mathcal{C}[X(t)] \in R(t)$.*

We prove the claim by induction. In the base case, $R(\ell) = \{(\varnothing, \varnothing)\}$ is indeed the set of all certificates for $\mathcal{I}[V_\ell]$ for all leaves ℓ in T, since $V_\ell = X(\ell) = \varnothing$.

If t is an introduce node with child t', consider a certificate $\mathcal{C}$ for $\mathcal{I}[V_t]$. Note that $\mathcal{C}[V_{t'}]$ is a certificate for $\mathcal{I}[V_{t'}]$ so $\mathcal{I}[X(t')] \in R(t')$ by the inductive hypothesis. Furthermore, $\mathcal{C}[X(t)]$ is a certificate for $\mathcal{I}[X(t)]$, thus the algorithm adds it to $R(t)$. In the opposite direction, consider $\mathcal{K} \in R(t)$ and observe that, by construction, there is a certificate $\mathcal{C}'$ to $\mathcal{I}[V_{t'}]$ such that $\mathcal{K}[X(t')] = \mathcal{C}'[X(t')]$. Since $\mathcal{K}$ is a certificate for $X(t)$ and $X(t) \cap V_{t'} = X(t')$, the union of $\mathcal{K}$ and $\mathcal{C}'$ is a certificate for $\mathcal{I}[V_t]$ by the patchwork property, and $\mathcal{K}$ is precisely its projection onto $X(t)$.

If t is a forget node with a child t', consider a certificate $\mathcal{C}$ for $\mathcal{I}[V_{t'}]$ and note that, since $V_t = V_{t'}$, it is also a certificate for $\mathcal{I}[V_t]$. By the inductive hypothesis, $\mathcal{C}[X(t')] \in R(t')$, hence, the algorithm adds $\mathcal{C}[X(t)]$ to $R(t)$. In the opposite direction, consider $\mathcal{K}[X(t)] \in R(t)$ where $\mathcal{K} \in R(t')$ and note that the inductive hypothesis implies that $\mathcal{K}$ and, subsequently, $\mathcal{K}[X(t)]$ are projections of a certificate for $\mathcal{I}[V_t]$.

If t joins nodes t_1 and t_2, consider a certificate $\mathcal{C}$ for $\mathcal{I}[V_t]$. Note that it is also a certificate for $\mathcal{I}[V_{t_1}]$ and $\mathcal{I}[V_{t_2}]$, since $V_{t_1}, V_{t_2} \subseteq V_t$. By the inductive hypothesis, $\mathcal{C}[X(t_1)] = \mathcal{C}[X(t_2)] = \mathcal{C}[X(t)] \in R(t_1) \cap R(t_2)$ and the algorithm adds it to $R(t)$. In the opposite direction, consider $\mathcal{K} \in R(t) = R(t_1) \cap R(t_2)$. By the inductive hypothesis, there are certificates $\mathcal{C}_1$ for $\mathcal{I}[V_{t_1}]$ and $\mathcal{C}_2$ for $\mathcal{I}[V_{t_2}]$ such that $\mathcal{C}_1[X(t)] = \mathcal{C}_2[X(t)] = \mathcal{K}$. By the property of tree decompositions, $V_{t_1} \cap V_{t_2} \subseteq X(t)$, thus the the union of $\mathcal{C}_1$ and $\mathcal{C}_2$ is a certificate for $\mathcal{I}[V_t]$ by the patchwork property, and $\mathcal{K}$ is precisely its projection onto $X(t)$.

We continue with the time complexity of the algorithm. Let w denote the width of the decomposition (T, X) and

assume that $\tau(m)$ is the time required to enumerate certificates for an instance of CSP($\mathbf{\Gamma}$) with m variables. Note that since $\mathbf{A}$ and $\mathbf{\Gamma}$ are finite, the function τ depends only on the number of variables. Furthermore, $\tau(m)$ is an upper bound on the number of complete satisfiable instances of CSP($\mathbf{A}$) with m variables.

Claim 2. *$R(t)$ is computed in $\tau(w+1)^2 \cdot O(w^k)$ time.*

First, note that $\tau(|X(t)|)$ is an upper bound on $|R(t)|$. Further, taking a projection of a certificate onto $U \subseteq V$ requires $O(|U|^k)$ time. If t is an introduce node, the computation of $R(t)$ requires at most $\tau(w+1)|R(t')| \cdot O(|X(t')|^k) \leq \tau(w+1)^2 \cdot O(w^k)$ time. If t is a forget node, the computation requires at most $|R(t')| \cdot O(|X(t)|^k) \leq \tau(w+1) \cdot O(w^k)$ time. Finally, if t joins nodes t_1 and t_2, the computation of $R(t)$ takes at most $|R(t_1)||R(t_2)| \cdot O(|X(t)|^k) \leq \tau(w+1)^2 \cdot O(w^k)$ time, where $O(w^k)$ accounts for the comparison of two certificates.

There are $O(n)$ nodes in the tree T, so the algorithm solves CSP($\mathbf{\Gamma}$) in $\tau(w+1)^2 \cdot O(w^k) \cdot O(n)$ time. The $\tau(w+1)^2 \cdot O(w^k)$ term depends only on the parameter w, hence CSP($\mathbf{\Gamma}$) is fpt. □

We continue by taking a closer look at some CSPs for qualitative spatial and temporal reasoning.

Corollary 6. *CSP($\mathbf{B}^{\vee=}$) is solvable in $2^{O(w^2)} \cdot O(n)$ time if $\mathbf{B}$ is $\mathbf{B}_{\mathrm{RCC5}}$ or $\mathbf{B}_{\mathrm{RCC8}}$, and $2^{O(w \log w)} \cdot O(n)$ time if $\mathbf{B}$ is $\mathbf{B}_{\mathrm{IA}}$, $\mathbf{B}_{\mathrm{BA}_d}$ or $\mathbf{B}_{\mathrm{CDC}}$.*

Proof. Consider Claim 2 in Theorem 5. Since the languages under consideration are JEPD and binary, the total number of instances of CSP($\mathbf{B}$) with w variables is $|\mathbf{B}|^{w^2} = 2^{O(w^2)}$, since $|\mathbf{B}|$ is constant. Solving instances of these CSPs takes polynomial time, so $\tau(w) = 2^{O(w^2)}$. This yields the result for RCC5 and RCC8.

For the remaining cases, we need a tighter bound on $\tau(w)$. We show that the number of complete certificates for these problems is at most $2^{O(w \log w)}$. Recall that an ordered partition of a set S of size n is a surjective function $\pi : S \to \{1, \ldots, r\}$ for some $r \in \{1, \ldots, n\}$. Any two elements of S can be compared with the usual relations $\{<, =, >\}$ according to the values assigned to them by π. Observe that that there are at most $n^n = 2^{O(n \log n)}$ ordered partitions of S.

Every complete satisfiable instance of CSP($\mathbf{B}_{\mathrm{IA}}$) corresponds to a unique ordered partition of the endpoints of the intervals (see e.g. (Stockman 2016)). For an instance with w variables (i.e. $2w$ endpoints), there are at most $2^{O(w \log w)}$ such partitions. Thus, an instance of CSP($\mathbf{B}_{\mathrm{IA}}^{\vee=}$) with w variables admits at most $2^{O(w \log w)}$ complete certificates. Given an ordered partition on the endpoints of the intervals, a polynomial-time procedure can recover the corresponding complete satisfiable instance of CSP($\mathbf{B}_{\mathrm{IA}}$), if one exists: for every variable, check that its left endpoint precedes its right endpoint – if not, then there is no corresponding instance; otherwise, deduce the relation between every pair of variables according to the ordered partition of their endpoints. The last step works since $\mathbf{B}_{\mathrm{IA}}$ is JEPD. Finally, observe that generating all (unordered) partitions of a set takes $O(1)$ amortized time per partition (Ichiro 1984) and generating all permutations takes $O(1)$ time per permutation (Sedgewick 1977). Thus, $\tau(w) = 2^{O(w \log w)}$ for CSP($\mathbf{B}_{\mathrm{IA}}^{\vee=}$).

The Block Algebra BA_d can be viewed as an extension of Allen's Interval Algebra to d dimensions, and the complete certificates correspond to d ordered partitions of the endpoints. We have $((2w)^{2w})^d = 2^{O(w \log w)}$ since d is fixed so $\tau(w) = 2^{O(w \log w)}$ for CSP($\mathbf{B}_{\mathrm{BA}_d}^{\vee=}$).

Every satisfiable instance of the CSP($\mathbf{B}_{\mathrm{CDC}}$) corresponds to two ordered partitions, one for each coordinate. There are $(w^w)^2 = 2^{O(w \log w)}$ such pairs of partitions, so $\tau(w) = 2^{O(w \log w)}$ for CSP($\mathbf{B}_{\mathrm{CDC}}^{\vee=}$). □

Applicability of the Algorithm

We analyse the applicability of our fpt result (Theorem 5) in this section. The patchwork property has not been directly verified for many formalisms—the list in Corollary 6 is quite meager. However, it has been verified implicitly for wide classes of relations. We make this explicit by connecting the patchwork property with the *amalgamation property*. This allows us to use model-theoretical concepts and results to identify interesting classes of relations that have the patchwork property. In the final step, we demonstrate how these ideas can be used on concrete examples — we study constraint languages that are definable in $\mathbf{Q} = (\mathbb{Q}; <, =, >)$ and phylogeny languages.

Patchwork and Amalgamation

When analysing PP from a model-theoretic angle, it is convenient to view CSPs in terms of homomorphisms. A *homomorphism* for τ-structures $\mathbf{A}, \mathbf{B}$ is a mapping $h : \mathbf{A} \to \mathbf{B}$ that preserves each relation of $\mathbf{A}$, i.e. if $(a_1, \ldots, a_k) \in R^{\mathbf{A}}$ for some k-ary relation symbol $R \in \tau$, then $(h(a_1), \ldots, h(a_k)) \in R^{\mathbf{B}}$. Let $\mathbf{B}$ be a structure with a (not necessarily finite) signature τ. CSP($\mathbf{B}$) is then the following decision problem:

INSTANCE. A finite τ-structure $\mathbf{A}$.
QUESTION. Is there a homomorphism from $\mathbf{A}$ to $\mathbf{B}$?

This definition coincides with the definition given earlier. We will use an analogue of subinstances for τ-structures: a τ-structure $\mathbf{A}$ is a *substructure* of a τ-structure $\mathbf{B}$ if and only if (1) the domain of $\mathbf{A}$ is a subset of the domain of $\mathbf{B}$ and (2) for each $R \in \tau$, the tuple $\overline{a}$ is in $R^{\mathbf{A}}$ if and only if $\overline{a}$ is in $R^{\mathbf{B}}$. We need various types of homomorphisms in what follows. A *strong* homomorphism additionally satisfies the only if direction in the definition of a homomorphism, i.e. it also preserves the complements of relations. An *embedding* is an injective strong homomorphism. An *isomorphism* is a surjective (and thus bijective) embedding, and an *automorphism* is an isomorphism from $\mathbf{A}$ to $\mathbf{A}$.

We connect the definition of patchwork with the *amalgamation property* (AP). A class $\mathcal{K}$ of τ-structures has AP if for every $\mathbf{B}_1, \mathbf{B}_2 \in \mathcal{K}$ such that their maximal common substructure $\mathbf{A}$ contains all elements that are both in $\mathbf{B}_1$ and $\mathbf{B}_2$, there exists $\mathbf{C} \in \mathcal{K}$ (called an *amalgam*) and embeddings $f_1 : \mathbf{B}_1 \to \mathbf{C}$ and $f_2 : \mathbf{B}_2 \to \mathbf{C}$ such that $f_1(a) = f_2(a)$

for every $a \in \mathbf{A}$. Let $\mathbf{D}$ be a countable τ-structure. Age($\mathbf{D}$) denotes the class of all finite τ-structures that embed into $\mathbf{D}$. Various connections between patchwork and amalgamation concepts have been hinted upon in the literature many times (see e.g. Bodirsky and Jonsson (2017), Huang (2012), and Li et al. (2008; 2009)) but the details have not been clearly spelled out. Connections between PP and amalgamation were also studied by Baader & Rydval (2020), but their results do not apply directly to structures that are k-ary (which is required by Theorem 5). Thus, we prove:

Theorem 7. *Let $\mathbf{D}$ be a k-ary JEPD τ-structure with domain D and assume that the k-ary equality relation $Eq_k = \{(d, \dots, d) \in D^k \mid d \in D\}$ is in $\mathbf{D}$. If* Age($\mathbf{D}$) *has the amalgamation property, then $\mathbf{D}$ has the patchwork property.*

Proof. Consider the instances $I_1 = (V_1, C_1)$, $I_2 = (V_2, C_2)$ of CSP($\mathbf{D}$) in Definition 3 as τ-structures $\mathbf{I}_1$, $\mathbf{I}_2$. Note that the intersection $I_1[V_1 \cap V_2] = I_2[V_1 \cap V_2]$ viewed as a τ-structure $\mathbf{A}$ is the maximal common substructure of $\mathbf{I}_1$, $\mathbf{I}_2$ and contains all elements that appear in both of them. To apply AP, we need to show that $\mathbf{I}_1$ and $\mathbf{I}_2$ embed into $\mathbf{D}$. Recall that an embedding is an injective strong homomorphism.

The remainder of the proof applies for all $i \in \{1, 2\}$. Since I_i is satisfiable, there is a homomorphism $h_i : \mathbf{I}_i \to \mathbf{D}$. Additionally, I_i is complete and $\mathbf{D}$ has JEPD relations, so for all $R \in \tau$, $(h_i(x_1), \dots, h_i(x_k)) \in R^{\mathbf{D}}$ implies that the constraint $R(x_1, \dots, x_k)$ is in C_i and it is satisfied. Hence, h_i is a strong homomorphism. To show that it is injective, we observe that for all $x, y \in \mathbf{I}_i$, if $Eq_k(x, y, \dots, y) \in C_i$, then $x = y$. Otherwise, by completeness, there is another $R \in \tau$ such that $R(x, y, \dots, y) \in C_i$. By PD, $R \cap Eq_k = \varnothing$, so $x \neq y$. Thus, h_i is injective, and ergo, an embedding.

We know that $\mathbf{I}_1, \mathbf{I}_2 \in$ Age($\mathbf{D}$) so the amalgam of $\mathbf{I}_1$ and $\mathbf{I}_2$ is in Age($\mathbf{D}$) by AP. Note that the structure $\mathbf{C}$ defined by $(V_1 \cup V_2, C_1 \cup C_2)$ embeds into the amalgam. Hence, it is homomorphic to $\mathbf{D}$ and $(V_1 \cup V_2, C_1 \cup C_2)$ is satisfiable. □

Homogeneity

Theorem 7 allows us to relate PP to some properties and results that have been important in the study of CSPs. To this end, we will use *homogeneity*. A homogeneous structure $\mathbf{A}$ is a countable structure such that for every isomorphism $f : \mathbf{B} \to \mathbf{C}$ between finite substructures $\mathbf{B}, \mathbf{C}$ of $\mathbf{A}$, there is an automorphism f' of $\mathbf{A}$ extending f. The following result is part of the classical *Fraïssé's Theorem* (1953).

Theorem 8. Age($\mathbf{A}$) *has AP when $\mathbf{A}$ is a countable homogeneous structure with a countable signature.*

Fraïssé's Theorem is explained in most textbooks on model theory, e.g. Hodges (1997). Combining Theorems 7 and 8 gives us the next result.

Corollary 9. *Let $\mathbf{A}$ be a k-ary structure with a countable domain, a finite signature, and that contains the k-ary equality relation Eq_k. Then, $\mathbf{A}$ has PP if $\mathbf{A}$ is homogeneous.*

A large number of homogeneous structures are known from the literature (see, for example, the survey by Macpherson (2011) and Hirsch (1997)) and they play an important role in CSP research. In fact, after the Feder-Vardi conjecture on finite-domain CSPs was settled (independently) by Bulatov (2017) and Zhuk (2020), much of the complexity-oriented work has concentrated on homogeneous infinite-domain CSPs. We note that all examples in Corollary 6 can be formulated by homogeneous structures; for instance, Hirsch (1996) has proven this for Allen's algebra and Bodirsky and Wölfl (2011) for RCC8.

Examples

The machinery presented above allows us to show fpt results for large families of CSPs. Our first example is the set of CSPs $\mathcal{T}$ whose constraint languages consist of relations that are in $\langle \mathbf{Q} \rangle_{\mathrm{b}}$. Well-known CSPs in $\mathcal{T}$ are the point algebra (Vilain and Kautz 1986), the ORD-Horn class (Nebel and Bürckert 1995) and certain scheduling problems (Möhring, Skutella, and Stork 2004) together with basic problems in complexity theory such as BETWEENNESS and CYCLIC ORDERING (Garey and Johnson 1979). Clearly, $\mathcal{T}$ contains many different CSPs based on non-binary relations and, in fact, the CSPs with binary relations are a subset of the point algebra and thus polynomial-time solvable (Vilain and Kautz 1986). The CSPs in $\mathcal{T}$ have been intensively studied in the literature: for instance, Bodirsky and Kára (2010) proved that any CSP in $\mathcal{T}$ is either polynomial-time solvable or NP-complete.

Arbitrarily choose CSP(Γ) in $\mathcal{T}$. It is folklore that the structure $\mathbf{Q}$ is homogeneous (see, for instance, Example 2.1.2 in Macpherson (2011) for a proof sketch). The structure $\mathbf{Q}$ is obviously JEPD and it contains the binary equality relation, so it has PP by Corollary 9. Since CSP($\mathbf{Q}$) is decidable, it follows from Theorem 5 that CSP(Γ) is fpt.

Proposition 10. *Every problem in $\mathcal{T}$ is fpt.*

Hirsch (1997) points out and discusses interesting homogeneous structures whose CSP can be solved with the same approach as for $\mathcal{T}$. Moreover, Hirsch (1996) proposed studying the computational complexity of CSPs for *relation algebras*, with obvious applications in AI. Inspired by this research programme, Bodirsky and Knäuer (2021) recently identified sufficient conditions for homogeneity of relation algebras. Their results provide further examples of CSPs that are covered by Theorem 5.

We continue with a more elaborate example that demonstrates usefulness of Theorem 5 beyond CSPs. *Phylogeny problems* are used for phylogenetic reconstruction in bioinformatics, but also in areas such as database theory, computational genealogy, and computational linguistics. A recent overview can be found in Warnow (2017). The problem is intuitively the following: given a partial description of a tree, is there a tree that is compatible with the given information? Many problems of this kind are NP-hard: concrete examples include the *subtree avoidance problem* (Ng, Steel, and Wormald 2000), the *forbidden triple problem* (Bryant 1997), and the *quartet consistency problem* (Steel 1992). Fpt algorithms are thus an interesting option for solving phylogeny problems. Our basic idea is to rephrase phylogeny problems as CSPs and then apply Theorem 5. We formalise this below, mostly following Bodirsky et al. (2017).

Let T be a *tree*, i.e. an undirected, acyclic, connected graph, and let r be the *root* of T. We only consider binary

trees, i.e. all vertices except for the root have either degree 3 or 1, and the root has either degree 2 or 0. The vertices of T are denoted by $V(T)$ and the leaves $L(T) \subseteq V(T)$ are the vertices of degree 1. For arbitrary $u, v \in V(T)$, we say that u *lies below* v if the path from u to the root r passes through v. We say that u *lies strictly below* v if u lies below v and $u \neq v$. The *youngest common ancestor* (yca) of $S \subseteq V(T)$ is the vertex u that lies above all vertices in S and has maximal distance from r; this vertex is uniquely determined by S. The *leaf structure* of T is the $\{C\}$-structure $(L(T); C)$ where $(x, y, z) \in C$ if and only if $yca(\{y, z\})$ lies strictly below $yca(\{x, y, z\})$. Following the literature on phylogeny problems, we write $x|yz$ instead of $C(x, y, z)$.

An *atomic phylogeny formula* ϕ is a conjunction of formulas of the form $x|yz$ and $x = y$. We say that ϕ with variables V is *satisfiable* if there exists a rooted binary tree T and a mapping $s : V \to L(T)$ such that ϕ is satisfied by T under s. The atomic phylogeny problem APHYL is the computational problem with atomic phylogeny formulas as instances and the question is whether the formula is satisfiable or not. APHYL is connected to CSPs as follows.

Theorem 11 (Prop. 2 in Bodirsky et al. (2017))**.** *There exists a homogeneous $\{|, =\}$-structure $\mathbf{A}$ with a countable domain and the following property: an instance I of* APHYL *is satisfiable if and only if I (viewed as an instance of CSP($\{|, =\}$)) homomorphically maps to $\mathbf{A}$.*

The relation $x|yz$ will be a basic relation in the CSP we are aiming for. Since we need a JEPD set of relations as the basis for Theorem 5, the following observation (see, for instance Bodirsky et al. (2017, Sec. 2.1)) is useful.

Observation. Let x, y, z be arbitrary leaves in an arbitrarily chosen rooted binary tree. If $x|yz$, then it may be the case that $y = z$. However, $x|yz$ implies that $x \neq y$ and $x \neq z$. Hence, we either have $x|yz$, $y|xz$, $z|xy$, or $x = y = z$.

Assume that the structure $\mathbf{A}$ in Theorem 11 has domain A and contains the relations $|'$ and $='$. Let $\mathbf{P}$ denote the structure $(A; R_1, R_2, R_3, R_4)$ where $R_1(x, y, z) \Leftrightarrow x|'yz$, $R_2(x, y, z) \Leftrightarrow y|'xz$, $R_3(x, y, z) \Leftrightarrow z|'xy$, and $R_4(x, y, z) \Leftrightarrow (x =' y =' z)$.

Proposition 12. *Let $\Gamma \subseteq \langle \mathbf{P} \rangle_{\mathrm{b}}$. Then CSP($\Gamma$) is fpt.*

Proof. We know that $\mathbf{P} \subseteq \langle (A; |', =') \rangle_{\mathrm{b}}$ and it is straightforward to verify that $\mathbf{P}$ is homogeneous since $(A; |', =')$ is homogeneous—all relations in $\mathbf{P}$ can be obtained by permuting the arguments of relations in $(A; |', =')$. The structure $\mathbf{P}$ is JEPD by the Observation and it contains the ternary equality relation. Thus, $\mathbf{P}$ has PP by Corollary 9. Finally, CSP($\mathbf{P}$) is solvable in polynomial time (Aho et al. 1981) and the proposition follows from Theorem 5. □

This proves that the three examples of NP-hard phylogeny problems that were discussed earlier are fpt: the essence of Theorem 11 and Proposition 12 is that there exist relations in $\langle \mathbf{P} \rangle_{\mathrm{b}}$ that capture the underlying formulas. The exact problem formulations can be found in Bodirsky et al. (2017, Sec. 2.2). The disequality relation neq is used for defining relations in some of these examples—note that $\mathrm{neq}(x, y) \Leftrightarrow \neg R_4(x, x, y) \Leftrightarrow R_1(x, x, y) \vee R_2(x, x, y) \vee R_3(x, x, y)$ so it is a member of $\langle \mathbf{P} \rangle_{\mathrm{b}}$.

Discussion and Future Research

Huang et al. (2013) proved that CSP($\mathbf{A}$) is in XP whenever $\mathbf{A}$ is a binary constraint language with aNAP. This property is PP restricted to binary relations with the completeness condition replaced by the algebraic closure condition. aNAP is less restrictive than PP, so it might be preferred in practical implementations for some constraint languages. However, in the worst case, using aNAP yields no advantage over using PP, and it is only defined for binary languages. We can thus conclude that our algorithm has a larger scope of applicability than the algorithm by Huang et al.

Bodirsky & Dalmau (2013) show that CSP($\mathbf{A}$) is in XP whenever $\mathbf{A}$ is a countable structure that is ω-categorical. While our results improve this to fixed-parameter tractability in many cases, there are interesting examples of ω-categorical structures $\mathbf{A}$ that do not have PP. An eminent ω-categorical example is the *branching time algebra* (BTA) which has been used, for example, in planning (Dean and Boddy 1988) and as the basis for temporal logics (Emerson and Halpern 1986). One may formulate BTA as a CSP($\mathbf{B}_{\mathrm{BTA}}$) where $\mathbf{B}_{\mathrm{BTA}}$ is JEPD and ω-categorical (Adeleke and Neumann 1998), However, one cannot formulate this problem as a CSP with PP; this follows from adapting an argument by Hirsch (1997, Sec. 4.1). Both CSP($\mathbf{B}_{\mathrm{BTA}}$) and CSP($\mathbf{B}_{\mathrm{BTA}}^{\vee=}$) are solvable in polynomial time (Hirsch 1997, Sec. 4.2), but there are finite $\mathbf{\Gamma} \subseteq \langle \mathbf{B}_{\mathrm{BTA}} \rangle_{\mathrm{b}}$ such that CSP($\mathbf{\Gamma}$) is NP-hard (Broxvall and Jonsson 2003). It is thus natural to ask whether CSP($\mathbf{\Gamma}$) is fpt when $\mathbf{\Gamma}$ contains higher-arity relations defined over $\mathbf{B}_{\mathrm{BTA}}$.

There are many relevant CSP($\mathbf{\Gamma}$) where $\mathbf{\Gamma}$ is not ω-categorical. Well-known examples include the unit interval algebra (i.e. Allen's algebra restricted to intervals of equal length (Pe'er and Shamir 1997)) and temporal problems that can express metric time such as the simple temporal problem, various disjunctive temporal problems and extended variants of Allen's algebra (Dechter, Meiri, and Pearl 1991; Krokhin, Jeavons, and Jonsson 2004; Oddi and Cesta 2000). It is known that some of these problems are in XP (Dabrowski et al. 2021) but a more complete picture is lacking. Studying this is an obvious future research direction.

We know from Corollary 6 that Allen's Algebra can be solved in $2^{O(w \log w)} \cdot n^{O(1)}$ time. Recall that a primal graph of a CSP($\mathbf{B}^{\vee=}$) instance with n variables has treewidth at most n. A $2^{o(w \log w)} \cdot n^{O(1)}$ time algorithm would thus lead to the fastest known algorithm for Allen's Algebra by beating the $2^{O(n \log n)}$ time algorithm pointed out by Stockman (2016). Improving dependence on w is an interesting research direction, but it may be a difficult problem given this observation.

One way of attacking computationally hard CSPs is to analyse parameters other than primal treewidth. Examples that come to mind are the treewidth of the dual graph or the incidence graph and variants of hypertree width, since they have been successfully used for efficiently solving CSPs as well as other combinatorial problems. These parameters are not so interesting for finite constraint languages since they are within a constant factor of the primal treewidth.

Acknowledgements

The second and the fourth authors were supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. In addition, the second author was partially supported by the Swedish Research Council (VR) under grant 2017-04112. Sebastian Ordyniak acknowledges support from the Engineering and Physical Sciences Research Council (EPSRC, project EP/V00252X/1).

References

Adeleke, S. A.; and Neumann, P. M. 1998. *Relations Related to Betweenness: Their Structure and Automorphisms*. American Mathematical Society.

Aho, A. V.; Sagiv, Y.; Szymanski, T. G.; and Ullman, J. D. 1981. Inferring a Tree from Lowest Common Ancestors with an Application to the Optimization of Relational Expressions. *SIAM Journal on Computing* 10(3): 405–421.

Allen, J. F. 1983. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM* 26(11): 832–843.

Arnborg, S.; Corneil, D.; and Proskurowski, A. 1987. Complexity of Finding Embeddings in a k-Tree. *SIAM Journal on Matrix Analysis and Applications* 8(2): 277–284.

Baader, F.; and Rydval, J. 2020. Description Logics with Concrete Domains and General Concept Inclusions Revisited. In *Proc. 10th International Joint Conference on Automated Reasoning (IJCAR-2020)*, 413–431.

Balbiani, P.; Condotta, J.; and del Cerro, L. F. 1998. A Model for Reasoning about Bidimensional Temporal Relations. In *Proc. 6th International Conference on Principles of Knowledge Representation and Reasoning (KR-1998)*, 124–130.

Bertelé, U.; and Brioschi, F. 1972. *Nonserial Dynamic Programming*. Academic Press.

Bodirsky, M.; and Dalmau, V. 2013. Datalog and Constraint Satisfaction with Infinite Templates. *Journal of Computer and System Sciences* 79(1): 79–100.

Bodirsky, M.; and Grohe, M. 2008. Non-Dichotomies in Constraint Satisfaction Complexity. In *Proc. 35th International Colloquium on Automata, Languages and Programming (ICALP-2008)*, 184–196.

Bodirsky, M.; and Jonsson, P. 2017. A Model-Theoretic View on Qualitative Constraint Reasoning. *Journal of Artificial Intelligence Research* 58: 339–385.

Bodirsky, M.; Jonsson, P.; and Pham, V. T. 2017. The Complexity of Phylogeny Constraint Satisfaction Problems. *ACM Transactions on Computational Logic* 18(3): 23:1–23:42.

Bodirsky, M.; and Kára, J. 2010. The Complexity of Temporal Constraint Satisfaction Problems. *Journal of the ACM* 57(2): 9:1–9:41.

Bodirsky, M.; and Knäuer, S. 2021. Network satisfaction for symmetric relation algebras with a flexible atom. In *Proc. 35th AAAI Conference on Artificial Intelligence (AAAI-2021)*, to appear.

Bodirsky, M.; and Wölfl, S. 2011. RCC8 is Polynomial on Networks of Bounded Treewidth. In *Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI-2011)*, 756–761.

Bodlaender, H. L. 1996. A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. *SIAM Journal on Computing* 25(6): 1305–1317.

Bodlaender, H. L.; and Kloks, T. 1996. Efficient and Constructive Algorithms for the Pathwidth and Treewidth of Graphs. *Journal of Algorithms* 21(2): 358–402.

Broxvall, M.; and Jonsson, P. 2003. Point Algebras for Temporal Reasoning: Algorithms and Complexity. *Artificial Intelligence* 149(2): 179–220.

Bryant, D. 1997. *Building Trees, Hunting for Trees, and Comparing Trees.* Ph.D. thesis, University of Canterbury.

Bulatov, A. 2017. A Dichotomy Theorem for Nonuniform CSPs. In *Proc. 58th Annual Symposium on Foundations of Computer Science (FOCS-2017)*.

Dabrowski, K.; Jonsson, P.; Ordyniak, S.; and Osipov, G. 2021. Disjunctive Temporal Problems under Structural Restrictions. In *Proc. 35th AAAI Conference on Artificial Intelligence (AAAI-2021)*, to appear.

Dean, T. L.; and Boddy, M. S. 1988. Reasoning About Partially Ordered Events. *Artificial Intelligence* 36(3): 375–399.

Dechter, R. 2003. *Constraint Processing*. Elsevier Morgan Kaufmann.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal Constraint Networks. *Artificial Intelligence* 49(1-3): 61–95.

Downey, R. G.; and Fellows, M. R. 2013. *Fundamentals of Parameterized Complexity*. Texts in Computer Science. Springer.

Emerson, E. A.; and Halpern, J. Y. 1986. "Sometimes" and "Not Never" Revisited: on Branching versus Linear Time Temporal Logic. *Journal of the ACM* 33(1): 151–178.

Flum, J.; and Grohe, M. 2006. *Parameterized Complexity Theory*. Springer. ISBN 3540299521.

Fraïssé, R. 1953. Sur Certaines Relations qui Généralisent l'Ordre des Nombres Rationnels. *Comptes Rendus de l'Académie des Sciences* 237: 540–542.

Garey, M.; and Johnson, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman and Company.

Gottlob, G.; Scarcello, F.; and Sideri, M. 2002. Fixed-Parameter Complexity in AI and Nonmonotonic Reasoning. *Artificial Intelligence* 138(1-2): 55–86.

Hirsch, R. 1996. Relation Algebras of Intervals. *Artificial Intelligence* 83(2): 267–295.

Hirsch, R. 1997. Expressive Power and Complexity in Algebraic Logic. *Journal of Logic and Computation* 7(3): 309–351.

Hodges, W. 1997. *A Shorter Model Theory*. New York, NY, USA: Cambridge University Press.

Huang, J. 2012. Compactness and Its Implications for Qualitative Spatial and Temporal Reasoning. In *Proc. 13th International Conference on Principles of Knowledge Representation and Reasoning (KR-2012)*.

Huang, J.; Li, J. J.; and Renz, J. 2013. Decomposition and Tractability in Qualitative Spatial and Temporal Reasoning. *Artificial Intelligence* 195: 140–164.

Ichiro, S. 1984. An Efficient Algorithm for Generating all Partitions of the Set $\{1, 2, \ldots, n\}$. *Journal of Information Processing* 7(1): 41–42.

Krokhin, A. A.; Jeavons, P.; and Jonsson, P. 2004. Constraint Satisfaction Problems on Intervals and Length. *SIAM Journal of Discrete Mathematics* 17(3): 453–477.

Li, J. J.; Huang, J.; and Renz, J. 2009. A Divide-and-Conquer Approach for Solving Interval Algebra Networks. In *Proc. 21st International Joint Conference on Artificial Intelligence (IJCAI-2009)*, 572–577.

Li, J. J.; Kowalski, T.; Renz, J.; and Li, S. 2008. Combining Binary Constraint Networks in Qualitative Reasoning. In *Proc. 18th European Conference on Artificial Intelligence (ECAI-2008)*, volume 178, 515–519.

Ligozat, G. 1998. Reasoning about Cardinal Directions. *Journal of Visual Languages and Computing* 9(1): 23–44.

Lutz, C.; and Miličić, M. 2007. A Tableau Algorithm for Description Logics with Concrete Domains and General TBoxes. *Journal of Automated Reasoning* 38(1-3): 227–259.

Macpherson, D. 2011. A Survey of Homogeneous Structures. *Discrete Mathematics* 311(15): 1599–1634.

Möhring, R. H.; Skutella, M.; and Stork, F. 2004. Scheduling with AND/OR Precedence Constraints. *SIAM Journal on Computing* 33(2): 393–415.

Nebel, B.; and Bürckert, H. 1995. Reasoning about Temporal Relations: A Maximal Tractable Subclass of Allen's Interval Algebra. *Journal of the ACM* 42(1): 43–66.

Ng, M. P.; Steel, M. A.; and Wormald, N. C. 2000. The Difficulty of Constructing a Leaf-labelled Tree Including or Avoiding Given Subtrees. *Discrete Applied Mathematics* 98(3): 227–235.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press.

Oddi, A.; and Cesta, A. 2000. Incremental Forward Checking for the Disjunctive Temporal Problem. In *Proc. 14th European Conference on Artificial Intelligence (ECAI-2000)*, 108–112.

Pe'er, I.; and Shamir, R. 1997. Satisfiability Problems on Intervals and Unit Intervals. *Theoretical Computer Science* 175(2): 349–372.

Randell, D.; Cui, Z.; and Cohn, A. 1992. A Spatial Logic based on Regions and Connection. In *Proc. 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-1992)*, 165–176.

Robertson, N.; and Seymour, P. D. 1984. Graph minors. III. Planar Tree-Width. *Journal of Combinatorial Theory, Series B* 36(1): 49–64.

Rossi, F.; van Beek, P.; and Walsh, T., eds. 2006. *Handbook of Constraint Programming*. Elsevier.

Samer, M.; and Szeider, S. 2010. Constraint Satisfaction with Bounded Treewidth Revisited. *Journal of Computer and System Sciences* 76(2): 103–114.

Sedgewick, R. 1977. Permutation Generation Methods. *ACM Computing Surveys* 9(2): 137–164.

Steel, M. 1992. The Complexity of Reconstructing Trees from Qualitative Characters and Subtrees. *Journal of Classification* 9: 91–116.

Stockman, P. 2016. *Upper Bounds on the Time Complexity of Temporal CSPs*. Master's thesis, Linköping University, Department of Computer and Information Science.

Vilain, M. B.; and Kautz, H. A. 1986. Constraint Propagation Algorithms for Temporal Reasoning. In *Proc. 5th National Conference on Artificial Intelligence (AAAI-1986)*, 377–382.

Warnow, T. 2017. *Computational Phylogenetics: An Introduction to Designing Methods for Phylogeny Estimation*. Cambridge University Press.

Zhuk, D. 2020. A Proof of the CSP Dichotomy Conjecture. *Journal of the ACM* 67(5): 30:1–30:78.