Journal of Computer and System Sciences 79 (2013) 79–100

Contents lists available at SciVerse ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Datalog and constraint satisfaction with infinite templates ☆

Manuel Bodirsky [a,*,1], Víctor Dalmau [b,2]

[a] CNRS/LIX, École Polytechnique, France
[b] Universitat Pompeu Fabra, Spain

ARTICLE INFO

Article history:
Received 19 December 2008
Received in revised form 26 April 2012
Accepted 25 May 2012
Available online 7 June 2012

Keywords:
Logic in computer science
Computational complexity
Constraint satisfaction
Datalog
Countably categorical structures

ABSTRACT

On finite structures, there is a well-known connection between the expressive power of Datalog, finite variable logics, the existential pebble game, and bounded hypertree duality. We study this connection for infinite structures. This has applications for constraint satisfaction with infinite templates. If the template Γ is ω-categorical, we present various equivalent characterizations of those Γ such that the constraint satisfaction problem (CSP) for Γ can be solved by a Datalog program. We also show that CSP(Γ) can be solved in polynomial time for arbitrary ω-categorical structures Γ if the input is restricted to instances of bounded treewidth. Finally, we characterize those ω-categorical templates whose CSP has Datalog width 1, and those whose CSP has strict Datalog width k.

1. Introduction

In a constraint satisfaction problem we are given a set of variables and a set of constraints on these variables, and want to find an assignment of values from some domain D to the variables such that all the constraints are satisfied. The computational complexity of a constraint satisfaction problem depends on the type of constraints that can be used in the instances of the problem. For *finite* domains D, the complexity of the constraint satisfaction problem attracted considerable attention in recent years; we refer to a recent collection of survey papers for a more complete account [20].

Constraint satisfaction problems where the domain D is infinite have been studied in Artificial Intelligence and the theory of binary relation algebras [24,37], with applications for instance in temporal and spatial reasoning. Well-known examples of such binary relation algebras are the *point algebra*, the *containment algebra*, *Allen's interval algebra*, and the *left linear point algebra*; see [21,24,30,37] and the references therein.

Constraint satisfaction problems can be modeled as homomorphism problems [26]. For detailed formal definitions of relational structures and homomorphisms, see Section 2, and for the connection to network satisfaction problems for relation algebras, see Section 8. Let Γ be a (finite or infinite) structure with a finite relational signature τ. Then the *constraint satisfaction problem* (*CSP*) for Γ is the following computational problem.

☆ An extended abstract of this paper appeared in the proceedings of the 23rd International Symposium on Theoretical Aspects of Computer Science (STACS'06) (Bodirsky and Dalmau, 2006 [8]).

* Corresponding author.
E-mail addresses: bodirsky@lix.polytechnique.fr (M. Bodirsky), victor.dalmau@upf.edu (V. Dalmau).

[1] The first author has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement No. 257039).
[2] Supported by the MICINN through grant TIN2010-20967-C04-02.

0022-0000/$ – see front matter
http://dx.doi.org/10.1016/j.jcss.2012.05.012

CSP(Γ)
INSTANCE: A finite τ-structure A.
QUESTION: Is there a homomorphism from A to Γ?

The structure Γ is called the *template* of the constraint satisfaction problem CSP(Γ). For example, if the template is the dense linear order of the rational numbers $(\mathbb{Q}, <)$, then it is easy to see that CSP(Γ) is the well-known problem of digraph-acyclicity.

Many constraint satisfaction problems in Artificial Intelligence can be formulated with ω-*categorical* templates. The concept of ω-categoricity is of central importance in model theory and will be introduced in Section 4.1; in the context of the network satisfaction problems for relation algebras, the relevance of ω-categoricity has already been recognized in [30]. An important class of examples for ω-categorical structures are the so-called Fraïssé-limits of amalgamation classes with finite relational signature [31]. It is well known that all the CSPs for the binary relation algebras (and their fragments) mentioned above, and many other problems in temporal and spatial reasoning can be formulated with ω-categorical structures.

For ω-categorical templates we can apply the so-called *algebraic approach to constraint satisfaction* [14,15,33] to analyze the computational complexity of the corresponding CSPs. This approach was originally developed for constraint satisfaction with finite templates, but several fundamental facts of the universal-algebraic approach also hold for ω-categorical templates [5,10]. The universal-algebraic approach has been used to obtain complete complexity classifications for large classes of ω-categorical templates [9,11].

Datalog. Datalog is an important algorithmic tool to study the complexity of CSPs. It can be viewed as the language of logic programs without function symbols, see e.g. [25,35]. For constraint satisfaction with finite domains, Datalog was first investigated systematically by Feder and Vardi [26]. Also for CSPs with infinite domains, Datalog programs play an important role (even though this is usually not made explicit in the literature), because one of the most studied algorithms in infinite-domain constraint satisfaction, the *path consistency algorithm*, and many of its variants can be formulated by Datalog programs (see Section 8).

Fix a set of relation symbols σ. A Datalog program consists of a finite set of *rules*, traditionally written in the form

$$\phi_0 :- \phi_1, \ldots, \phi_r$$

where $\phi_0, \phi_1, \ldots, \phi_r$ are *atomic σ-formulas*, that is, formulas of the form $R(x_1, \ldots, x_n)$ for $R \in \sigma$ and variables $x_1, \ldots, x_n$. In such a rule ϕ_0 is called the *head* and $\phi_1, \ldots, \phi_r$ the *body* of the rule. The relation symbols that never appear in rule heads are called the *input relation symbols*, or *EDBs* (this term comes from database theory, and stands for *extensional database*). The other relation symbols that appear in the Datalog program are called *IDBs* (short for *intentional database*).

Before we give formal definitions of the semantics of a Datalog program in Section 2, we show an instructive example.

$$tc(x, y) :- edge(x, y)$$

$$tc(x, y) :- tc(x, u), tc(u, y)$$

$$false :- tc(x, x)$$

Here, the binary relation *edge* is the only input relation symbol, *tc* is a binary IDB, and *false* is a 0-ary IDB. Informally, the Datalog program computes with the help of the relation *tc* the transitive closure of the edges in the input relation, and derives *false* if and only if the input (which can be seen as a digraph defined on the variables) contains a directed cycle. Hence, the program above derives *false* on a given directed graph if and only if the directed graph does *not* homomorphically map to $(\mathbb{Q}; <)$. In general, we say that a CSP is *solved* by a Datalog program if the distinguished 0-ary predicate *false* is derived on an instance of the CSP if and only if the instance has no solution. This will be made precise in Section 2.

An important measure for the complexity of a Datalog program is the maximal number k of variables per rule (see e.g. [27]). On structures of size n, such a Datalog program can be evaluated in time $O(n^{k+1})$. (Hence, a fixed Datalog program can be evaluated in time polynomial in n.) In this work, we are interested in capturing a finer distinction, and study the expressive power of Datalog depending both on the maximal number k of variables per rule and on the maximal number l of variables *in the head* of the rules. Such Datalog programs are said to have *width* (l, k). The Datalog program shown above, for instance, has width $(2, 3)$. The double parameterization is less common, but more general, and has already been considered in the literature on constraint satisfaction and Datalog [26].

For finite templates Γ, it has been shown that there is a tight connection between the expressive power of Datalog, the so-called existential pebble game, finite variable logics, and bounded hypertree duality; these concepts will be introduced in Section 2 and the mentioned connection will be formally stated in Section 3. The connection shows that the following are equivalent:

- there is a Datalog program of width (l, k) that solves CSP(Γ);
- for all instances A of CSP(Γ), if Duplicator has a winning strategy for the existential (l, k)-pebble game on A and Γ, then A homomorphically maps to Γ;

- for all instances A of CSP(Γ), if all sentences in the infinitary l-bounded existential positive k-variable logic $L^{l,k}_{\infty\omega}$ that hold in A also hold in Γ, then A homomorphically maps to Γ;
- there is a set $\mathcal{N}$ of finite structures of treewidth at most (l,k) such that every finite τ-structure A is homomorphic to Γ if and only if no structure in $\mathcal{N}$ is homomorphic to A.

The four characterizations provide links to different research areas – database theory, constraint satisfaction complexity, finite model theory, combinatorics – and the possibility to change between the various perspectives on width-bounded Datalog has had a profound impact on research in those areas.

We mention that recently an (effective) universal-algebraic condition has been found that characterizes which finite structures Γ have a CSP that can be solved with Datalog [4,13]. A corresponding characterization of those ω-categorical templates that can be solved by Datalog remains open.

Results. We study the connection between the expressive power of Datalog, finite variable logics, the existential pebble game, and bounded hypertree duality for *infinite structures* Γ. We show that the result for finite structures Γ mentioned above fails for general infinite structures (Section 3), but holds true if Γ is ω-categorical (Section 4). An important tool to characterize the expressive power of Datalog for constraint satisfaction is the notion of *canonical Datalog programs*. This concept was introduced by Feder and Vardi for finite templates; we present a generalization to ω-categorical templates. We prove that a CSP with an ω-categorical template can be solved by an (l,k)-Datalog program if and only if the canonical (l,k)-Datalog program for Γ solves the problem (Section 4.2).

An important consequence of our result is that for ω-categorical Γ, the problem CSP(Γ) can be solved in polynomial time if the input is restricted to a class of instances of bounded treewidth (in fact, it suffices that the *cores* of the instances have bounded treewidth).

We also investigate which CSPs can be solved with a Datalog program (and are thus polynomial-time tractable) when no restriction is imposed on the input instances (Section 6). In particular, we prove a characterization of CSPs with ω-categorical templates Γ that can be solved by a Datalog program of width $(1,k)$. In fact, every problem that is closed under disjoint unions and can be solved by a Datalog program of width $(1,k)$ for some k can be formulated as a constraint satisfaction problem with an ω-categorical template. More generally, one can find ω-categorical templates for problems that are closed under disjoint unions and can be described in the logic called *monotone monadic SNP* introduced by Feder and Vardi [26] (Section 5); to show this, we apply a model-theoretic result of Cherlin, Shelah and Shi [18].

A special case of width $(1,k)$-Datalog programs are problems that can be decided by establishing *arc-consistency* (sometimes also called *hyperarc consistency*), which is a well-known and intensively studied technique in artificial intelligence. We show that if a constraint satisfaction problem with an ω-categorical template can be decided by establishing arc-consistency, then it can also be formulated as a constraint satisfaction problem with a finite template (Section 6).

Finally, we characterize *strict width* l, a notion that was again introduced for finite templates and for $l \geqslant 2$ in [26]. Intuitively, CSP(Γ) has strict width l, for $l \geqslant 2$, if there is a Datalog program of width (l,k), for some $k \geqslant l+1$, that computes on a given instance A of CSP(Γ) *'all the l-ary facts that are implied by A'*, that is, it makes all semantically entailed l-ary constraints syntactically present. Obviously, this needs a careful formal definition, which we present in Section 7. Jeavons et al. [32] say that in this case *establishing strong l-consistency ensures global consistency*. For finite templates, strict width l can be characterized by an algebraic closure condition [26,32]. In Section 7 we generalize this result to ω-categorical templates Γ with a finite signature, and show that CSP(Γ) has strict width l if and only if for every finite subset A of the domain of Γ there is an $(l+1)$-ary polymorphism of Γ that is a *near-unanimity operation on A*, i.e., it satisfies the identity $f(x,\dots,x,y,x,\dots,x) = x$ for all $x, y \in A$. We would like to remark that this is the only known tractability condition formulated in terms of polymorphism identities that holds unconditionally for the class of all ω-categorical templates (besides the trivial condition of having a constant polymorphism).

2. Definitions and basic facts

A *relational signature* τ is a (here always at most countable) set of *relation symbols* R_i (also called *predicates*), each associated with an *arity* $k_i \in \mathbb{N}$. A (*relational*) *structure* Γ *over relational signature* τ (also called *τ-structure*) is a set D_Γ (the *domain*) together with a relation $R_i \subseteq D_\Gamma^{k_i}$ for each relation symbol of arity k_i. If necessary, we write R^Γ to indicate that we are talking about the relation R belonging to the structure Γ. For simplicity, we otherwise denote both a relation symbol and its corresponding relation with the same symbol. For a τ-structure Γ and $R \in \tau$ it will also be convenient to say that $R(u_1,\dots,u_k)$ *holds in* Γ iff $(u_1,\dots,u_k) \in R$. We sometimes use the shortened notation $\bar{x}$ for a vector $x_1,\dots,x_n$ of any length. If we add relations to a given τ-structure Γ, then the resulting structure Γ' with a larger signature $\tau' \supset \tau$ is called a *τ'-expansion* of Γ, and Γ is called a *τ-reduct* of Γ'.

The *union* of two τ-structures Γ and Γ' with disjoint domains is a τ-structure Δ that is defined on the union of the domains of Γ and Γ'. We have $R^\Delta := R^\Gamma \cup R^{\Gamma'}$ for every $R \in \tau$. When the domain of Γ and Γ' is not disjoint, a *disjoint union* Δ of Γ and Γ' is the union of Γ with a copy of Γ' whose domain is distinct from the domain of Γ. Since we usually consider structures up to isomorphism, we also call the structure Δ *the* disjoint union of Γ and Γ'.

A τ-structure is called *connected* iff it is not the disjoint union of two τ-structures with a non-empty domain. The *Gaifman graph* (sometimes also called the *shadow*) of a relational structure A is a graph on the vertex set $v_1,\dots,v_n$ where

two distinct vertices v_k and v_l are adjacent if there is a relation in A that is imposed on both v_k and v_l, i.e., there is a relation R such that A satisfies $R(v_{i_1}, \dots, v_{i_j})$ and $k, l \in \{i_1, \dots, i_j\}$. It is clear that a structure is connected if and only if its Gaifman graph is connected.

2.1. Homomorphisms

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 $\bar{a} = (a_1, \dots, a_n)$, if $\bar{a} \in R^\Gamma$, then $(f(a_1), \dots, f(a_n)) \in R^\Delta$. In this case we say that the map f *preserves* the relation R. Two structures Γ and Δ are called *homomorphically equivalent* if there is a homomorphism from Γ to Δ and a homomorphism from Δ to Γ.

A *strong homomorphism* f satisfies the stronger condition that for each n-ary relation symbol in τ and each n-tuple $\bar{a}$ we have that $\bar{a} \in R^\Gamma$ if and only if $(f(a_1), \dots, f(a_n)) \in R^\Delta$. An *embedding* of a structure Γ into a structure Δ is an injective strong homomorphism. An *isomorphism* is a surjective embedding. Isomorphisms from Γ to Γ are called *automorphisms*.

If Γ and Δ are structures of the same signature, with $D_\Gamma \subseteq D_\Delta$, and the inclusion map is an embedding, then we say that Δ is an *extension* of Γ, and that Γ is a *restriction* of Δ.

A partial mapping h from a relational structure A to a relational structure B is called a *partial homomorphism* (from A to B) if h is a homomorphism from a restriction A' of A to B and A' is the domain of h. As usual, the *restriction* of a function h to a subset S of its range is the mapping h' with range S where $h'(x) = h(x)$ for all $x \in S$; h is called an *extension* of h'.

2.2. First-order logic

First-order formulas φ over the signature τ (or, in short, τ-formulas) are 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 usual Tarskian style. A τ-formula without free variables is called a τ-sentence. We write $\Gamma \models \varphi$ iff the τ-structure Γ is a model for the τ-sentence φ; this notation is lifted to sets of sentences in the usual way. A good introduction to logic and model theory is [31].

We can use first-order formulas over the signature τ to define relations over a given τ-structure Γ: for a formula $\varphi(x_1, \dots, x_k)$ where $x_1, \dots, x_k$ are the free variables of φ the corresponding relation R is the set of all k-tuples $(t_1, \dots, t_k) \in D_\Gamma^k$ such that $\varphi(t_1, \dots, t_k)$ is true in Γ.

A first-order formula φ is said to be *primitive positive* (we say φ is a *pp-formula*, for short) iff it is of the form $\exists \bar{x}(\varphi_1(\bar{x}) \wedge \dots \wedge \varphi_k(\bar{x}))$ where $\varphi_1, \dots, \varphi_k$ are atomic formulas (which might be equality relations of the form $x = y$).

2.3. Canonical queries

A basic concept to link structure homomorphisms and logic is the *canonical conjunctive query* ϕ^A of a finite relational structure A, which is a first-order formula of the form $\exists v_1, \dots, v_n(\psi_1 \wedge \dots \wedge \psi_m)$, where $v_1, \dots, v_n$ are the vertices of A, and $\{\psi_1, \dots, \psi_m\}$ is the set of atomic formulas of the form $R(v_{i_1}, \dots, v_{i_j})$ that hold in A.

It is a fundamental property of the canonical query ϕ^A that ϕ^A holds in a structure Γ if and only if there is a homomorphism from A to Γ [17].

2.4. Finite variable logics

The class of sentences that have at most k variables and are obtained from atomic formulas using infinitary conjunction, infinitary disjunction, and existential quantification is denoted by $\exists\mathcal{L}^k_{\infty\omega}$. The class $\bigcup_{k\geqslant 0} \exists\mathcal{L}^k_{\infty\omega}$ is denoted by $\exists\mathcal{L}^\omega_{\infty\omega}$.

We want to bring another parameter l into the picture, and define the following refinement of $\exists\mathcal{L}^k_{\infty\omega}$. A conjunction $\bigwedge \Psi$ is called *l-bounded* if Ψ is a collection of $\exists\mathcal{L}^\omega_{\infty\omega}$ formulas ψ that are quantifier-free or have at most l free variables. Similarly, a disjunction $\bigvee \Psi$ is called *l-bounded* if Ψ is a collection of $\exists\mathcal{L}^\omega_{\infty\omega}$ formulas ψ that are quantifier-free or have at most l free variables. The set of $\exists\mathcal{L}^{l,k}_{\infty\omega}$ formulas is defined as the restriction of $\exists\mathcal{L}^k_{\infty\omega}$ obtained by only allowing infinitary l-bounded conjunction and l-bounded disjunction instead of full infinitary conjunction and disjunction. Note that $\bigcup_{0\leqslant l<k} \exists\mathcal{L}^{l,k}_{\infty\omega}$ equals $\exists\mathcal{L}^k_{\infty\omega}$. The logic $\exists\mathcal{L}^k_{\infty\omega}$ was introduced (under a different name) by Kolaitis and Vardi as an existential negation-free variant of well-studied infinitary logics to study the expressive power of Datalog [34]; in subsequent work, they used the name $\exists\mathcal{L}^k_{\infty\omega}$ to denote this logic [35] and we follow this convention.

We denote by $L^{l,k}$ the logic $\exists\mathcal{L}^{l,k}_{\infty\omega}$ without disjunctions and with just *finite* l-bounded conjunctions. In other words, a formula in $L^{l,k}$ is composed out of existential quantification and finitary l-bounded conjunction, and uses only k distinct variables. We also call the logic $L^{l,k}$ *infinitary l-bounded existential positive k-variable logic*. The language $L^k := \bigcup_{l\geqslant 0} L^{l,k}$, where only the parameter k, but not the parameter $l \leqslant k$ is specified, has been studied for example by Kolaitis and Vardi [35] and later by Dalmau, Kolaitis and Vardi [22].

2.5. Datalog

We now formally define Datalog. Our definition will be purely operational; for the standard semantical approach to the evaluation of Datalog programs see [25,35]. Let τ be a relational signature. A Datalog program (with signature τ) is a finite set of *rules* of the form $\psi :- \phi_1, \ldots, \phi_r$, where $r \geqslant 0$ and where $\psi, \phi_1, \ldots, \phi_r$ are atomic τ-formulas. The formula ψ is called the *head* of the rule, and $\phi_1, \ldots, \phi_r$ is called the *body*. The relation symbols occurring in the head of some clause are called *intentional database predicates* (or *IDBs*), and all other relation symbols in the clauses are called *extensional database predicates* (or *EDBs*). A Datalog program has *width* (l, k) if all IDBs are at most l-ary, and if all rules have at most k distinct variables. A Datalog program has *width* l if it has width (l, k) for some k.

An *evaluation* of a Datalog program Π on a finite structure S proceeds in steps $i = 0, 1, \ldots$; at each step i we maintain a $(\tau \cup \sigma)$-structure S^i. The relations for the symbols from τ are always equal to the relations from S, i.e., for every $i \geqslant 0$ and every $R \in \tau$ we have $R^{S^i} = R^S$. For every relation symbol $R \in \sigma$ we have that $R^{S^i} \subseteq R^{S^{i+1}}$ for all $i \geqslant 0$. Initially, we start with the expansion S^0 of S where all symbols from σ denote the empty relation. Now suppose that $R_1(u_1^1, \ldots, u_{k_1}^1), \ldots, R_l(u_1^r, \ldots, u_{k_r}^r)$ hold in S^i, and that

$$R_0\big(y_1^0, \ldots, y_{k_0}^0\big) :- R_1\big(y_1^1, \ldots, y_{k_1}^1\big), \ldots, R_l\big(y_1^r, \ldots, y_{k_r}^r\big)$$

is a rule from Π, where $u_j^i = u_{j'}^{i'}$ if $y_j^i = y_{j'}^{i'}$. Then we add the tuple $(u_1^0, \ldots, u_{k_0}^0)$ to R in S^{i+1}, where $u_j^0 = u_{j'}^i$ if and only if $y_j^0 = y_{j'}^i$. We also say that the Datalog program *derives* $R(u_1^0, \ldots, u_{k_0}^0)$ from $R_1(u_1^1, \ldots, u_{k_1}^1), \ldots, R_l(u_1^r, \ldots, u_{k_r}^r)$. The procedure stops if no new tuples can be derived.

On an input structure with n elements a Datalog program of width l can derive at most n^l tuples, and it is clear that a fixed Datalog program can be evaluated on a given structure in polynomial time in the size of the structure.

We might use Datalog programs to solve constraint satisfaction problems $\mathrm{CSP}(\Gamma)$ for a template Γ with signature τ as follows. Let Π be a Datalog program whose set of EDBs is τ, and let σ be the set of IDBs of Π. We assume that there is one distinguished 0-ary intentional relation symbol *false*. The program Π is *sound* for $\mathrm{CSP}(\Gamma)$ if every finite τ-structure S does not homomorphically map to Γ whenever Π derives *false* on S. We say that Π *solves* $\mathrm{CSP}(\Gamma)$ if Π derives *false* (i.e., adds the 0-ary tuple to the relation for the symbol *false*) on S if and only if S does not homomorphically map to Γ.

Feder and Vardi showed that deciding whether a given finite template T has width 1 is decidable (see also [23]). Only recently it has been shown that the question whether a finite structure T has width (l, k) is decidable as well [4]; surprisingly, it turns out that $\mathrm{CSP}(T)$ has width l, for $l \geqslant 2$, if and only if it has width 2.

2.6. The existential pebble game

The existential k-pebble game has been introduced to the context of constraint satisfaction in [22,26,35]. As in [26], we study this game with a second parameter, and first define the *existential* (l, k)*-pebble game*. The usual existential k-pebble game is exactly the existential $(k-1, k)$-pebble game in our sense. Again, the second parameter is necessary to obtain the strongest formulations of our results.

The game is played by the players Spoiler and Duplicator on (possibly infinite) structures A and B of the same relational signature. Each player has k pebbles, $p_1, \ldots, p_k$ for Spoiler and $q_1, \ldots, q_k$ for Duplicator. Spoiler places his pebbles on elements of A, Duplicator her pebbles on elements of B. Initially, no pebbles are placed. In each round of the game Spoiler picks $k - l$ pebbles. If some of these pebbles are already placed on A, then Spoiler removes them from A, and Duplicator responds by removing the corresponding pebbles from B. Spoiler places the $k - l$ pebbles on elements of A, and Duplicator responds by placing the corresponding pebbles on elements of B. Let $i_1, \ldots, i_m$ be the indices of the pebbles that are placed on A (and B) after the i-th round. Let $a_{i_1}, \ldots, a_{i_m}$ $(b_{i_1}, \ldots, b_{i_m})$ be the elements of A (B) pebbled with the pebbles $p_{i_1}, \ldots, p_{i_m}$ $(q_{i_1}, \ldots, q_{i_m})$ after the i-th round. If the partial mapping h from A to B defined by $h(a_{i_j}) = b_{i_j}$, for $j \in \{1, \ldots, m\}$, is not a partial homomorphism from A to B, then the game is over, and Spoiler wins. Duplicator wins if the game continues forever.

It is convenient and customary [25] to define the existential pebble game in terms of winning positions.

Definition 1. A (*positional*) *winning strategy for Duplicator* for the existential (l, k)-pebble game on A, B is a non-empty set $\mathcal{H}$ of partial homomorphisms from A to B such that

- $\mathcal{H}$ is closed under restrictions of its members, and
- for all functions h in $\mathcal{H}$ with $|dom(h)| = d \leqslant l$ and for all $a_1, \ldots, a_{k-d} \in A$ there is an extension $h' \in \mathcal{H}$ of h such that h' is also defined on $a_1, \ldots, a_{k-d}$.

Following the usual convention we take Definition 1 as the definition of the existential pebble game. Consequently, we shall not give a formalization of the existential pebble game in terms of sequences of rounds and we shall not prove the equivalence between such a formalization and Definition 1. Such a formalization and equivalence proof can be found for a closely related game, the Ehrenfeucht–Fraïssé game, in full formal detail in [31, Lemma 3.2.2]; the modifications of the

presentation in [31] to the existential pebble game are straightforward. Instead of giving these modifications here, we shall work directly with Definition 1. In some of our proofs, however, it will be convenient to think about games in terms of a sequence of moves instead of winning positions. We have taken the liberty of describing some of those arguments in terms of a sequence of moves of Spoiler and Duplicator with the understanding that the argument could be easily transformed into the language of winning positions.

2.7. Treewidth

In the remainder of this section we define the notion of *treewidth* for relational structures. As in [26], we need to extend the ordinary notion of treewidth for relational structures in such a way that we can introduce the additional parameter l.

Let $0 \leqslant l < k$ be positive integers. An (l,k)-*tree* is defined inductively as follows:

- A k-clique is an (l,k)-tree.
- For every (l,k)-tree G and for every l-clique induced by nodes $v_1,\ldots,v_l$ in G, the graph G' obtained by adding $k-l$ new nodes $v_{l+1},\ldots,v_k$ to G and adding edges (v_i,v_j) for all $i \neq j$ with $i \in \{1,\ldots,k\}$, $j \in \{l+1,\ldots,k\}$ (so that $v_1,\ldots,v_k$ forms a k-clique) is also an (l,k)-tree.

A *partial* (l,k)-*tree* is a (not necessarily induced) subgraph of an (l,k)-tree.

Definition 2. Let $0 \leqslant l < k$ and let τ be a relational signature. We say that a τ-structure S has *treewidth at most* (l,k) if the Gaifman graph of S is a partial (l,k)-tree.

If a structure has treewidth at most $(k,k+1)$ we also say that it has *treewidth at most* k, and it is not difficult to see that these structures are precisely the structures of treewidth at most k in the sense of [35]. It is also possible to define partial (l,k)-trees by using tree-decompositions.

Definition 3. A *tree-decomposition* of a graph G is a tree T such that

1. the nodes of T are sets of nodes of G;
2. every edge of G is entirely contained in some node of T;
3. if a node v belongs to two nodes x, y of T it must also be in every node in the unique path from x to y.

A tree-decomposition T is said to be of *width* (l,k) if every node of T contains at most k nodes of G and the intersection of two different nodes of T has size at most l. The following is a straightforward generalization of a well-known fact for single parameter k, and the proof can be obtained by adapting for instance the proof given in [45].

Proposition 1. *A graph is a partial* (l,k)*-tree if and only if it has a tree-decomposition of width* (l,k).

It was shown in [35, Lemma 5.2], that the canonical query for a structure S of treewidth at most k can be expressed in the logic L^{k+1}. We show an analogous statement for both parameters l and k.

Lemma 1. *Let A be a finite structure of treewidth at most (l,k). Then the canonical query ϕ^A for A is logically equivalent to a sentence from $L^{l,k}$.*

Proof. Let A be a finite structure of treewidth at most (l,k), let G be its Gaifman graph, and let T be a tree-decomposition of G. Let us view T as a rooted tree with root $t = \{a_1,\ldots,a_{k'}\}$, $k' \leqslant k$. We shall show by structural induction on T that there exists a formula $\phi^A(y_1,\ldots,y_{k'})$ in $L^{l,k}$ with free variables $y_1,\ldots,y_{k'}$ such that for every structure B and elements $b_1,\ldots,b_{k'}$ in B the following two sentences are equivalent:

(1) The partial mapping from A to B that maps a_i to b_i for $1 \leqslant i \leqslant k'$ can be extended to a homomorphism from A to B;
(2) $Q(b_1,\ldots,b_{k'})$ holds in B.

The base case is when the tree contains only one node t. In this case ϕ^A is obtained by removing the existential quantifiers in the canonical conjunctive query of A. For the inductive step, let $t_1,\ldots,t_m$ be the children of the root t in T. Consider the m subtrees $T_1,\ldots,T_m$ of T obtained by removing t. For every $i = 1,\ldots,m$ we root T_i at t_i and consider the substructure A_i of A induced by the set of all nodes of A contained in some node of T_i. Then, T_i is a tree-decomposition of A_i and the induction hypothesis provides a formula ϕ^{A_i} for which (1) and (2) are equivalent. Let $\phi^A(y_1,\ldots,y_{k'})$ be the formula $\bigwedge \Phi$ where Φ is the following set of formulas:

(a) For each $i = 1, \ldots, m$ the set Φ contains the formula obtained by existentially quantifying all free variables y_j in ϕ^{A_i} where $a_j \notin t$. Note that the resulting formula has at most l free variables.
(b) The set Φ contains the conjuncts from the canonical query of the substructure of A induced by the nodes in t (as in the base case).

To show that (1) and (2) are equivalent, let B be an arbitrary structure. By the properties of the tree-decomposition we know that h is a homomorphism from A to B that maps a_i to b_i if and only if for all $i = 1, \ldots, m$ the restriction of h to A_i is a homomorphism from A_i to B and the restriction of h to the elements of t is a partial homomorphism from A to B as well. The former condition is equivalent to the fact that the assignment $y_{a_i} \mapsto b_i$ satisfies every formula of Φ included in (a). The latter condition is equivalent to the fact that the very same assignment satisfies the formula introduced in (b). □

3. State of the art for finite domains

In this section we recall the known connection between the existential pebble game, finite variable logics, and bounded treewidth duality for finite structures [22,34,35], and show that the connection fails if Γ is an arbitrary structure with an infinite domain. The equivalence of (1) and (2) has been shown in [34] (Theorem 4.8 there). The equivalence of (2) and (3) follows from results in [22]; a proof of the entire theorem will appear in [3].

A finite relational structure S is a *core* if every endomorphism of S is an automorphism of S. It is easy to see that every finite relational structure is homomorphically equivalent to a core, and that this core is unique up to isomorphism (see e.g. [29]).

Theorem 1. *Let A, B be finite relational structures over the same signature τ. Then the following are equivalent.*

1. *Duplicator has a winning strategy for the existential k-pebble game on A and B.*
2. *All τ-sentences in $\exists\mathcal{L}^k_{\infty\omega}$ that hold in A also hold in B.*
3. *Every finite τ-structure whose core has treewidth at most $k-1$ that homomorphically maps to A also homomorphically maps to B.*

We show that for infinite structures B it is in general not true that 3 implies 1.

Proposition 2. *There are infinite τ-structures A and B such that*

- *Duplicator does not have a winning strategy for the existential 2-pebble game on A and B;*
- *every finite τ-structure C of treewidth at most 1 that homomorphically maps to A also maps to B.*

Proof. Let B be the disjoint union of all non-isomorphic directed paths of finite length. Consider $A = C_3^{\rightarrow}$, the directed cycle on three vertices. Every finite τ-structure C of treewidth at most 1 is a finite oriented tree, and therefore homomorphically maps to A and to B. However, Spoiler clearly has a winning strategy. After Spoiler places his first pebble, Duplicator has to place his first pebble on a path of length l in B. By walking with his two pebbles in one direction on the directed cycle A, Spoiler can trap Duplicator after l rounds of the game. □

The following theorem combines results obtained in [22,26,34].

Theorem 2. *Let Γ be a τ-structure over a finite domain. Then for every k the following statements are equivalent.*

1. *There is a $(k-1,k)$-Datalog program that solves* CSP(Γ).
2. *For all finite τ-structures A, if Duplicator has a winning strategy for the existential k-pebble game on A and Γ, then A is in* CSP(Γ).
3. *The complement of* CSP(Γ) *can be formulated in $\exists\mathcal{L}^k_{\infty\omega}$.*
4. *For all finite τ-structures A, if every finite τ-structure C of treewidth at most k that homomorphically maps to A also maps to Γ, then A homomorphically maps to Γ.*

Proof. The equivalence between items 1, 2, and 3 has been shown in [35] (Theorem 4.8 there). The equivalence between items 1 and 4 is due to [26] (Theorem 23 there). □

Also Theorem 2 fails for structures Γ over an infinite domain. Intuitively, the reason is that the expressive power of infinitary disjunction is relatively larger for CSP(Γ) if Γ has an infinite domain.

Proposition 3. *There are an infinite τ-structure Γ and a finite τ-structure A such that*

- *the complement of* CSP(Γ) can *be formulated in* $\exists\mathcal{L}^2_{\infty\omega}$, *and*
- *Duplicator wins the existential 2-pebble game on A and Γ, but A is not in* CSP(Γ).

Proof. We choose Γ to be $(\mathbb{Q}, <)$. Duplicator wins the existential 2-pebble game on $C_3^{\rightarrow}$ and Γ, but there is no homomorphism from $C_3^{\rightarrow}$ to Γ.

The complement of CSP$(\mathbb{Q}, <)$ can be formulated in $\exists\mathcal{L}^2_{\infty\omega}$. Let Φ be an $\exists\mathcal{L}^2_{\infty\omega}$-sentence that expresses that a structure contains copies of $(\{1, \ldots, n\}, <)$ for arbitrarily large n. The finite directed graphs that do not homomorphically map to $(\mathbb{Q}, <)$ are precisely the directed graphs containing a directed cycle. Clearly, Φ holds precisely on those finite directed graphs that contain a directed cycle. □

4. Datalog for ω-categorical structures

The concept of ω-categoricity is of central interest in model theory [16,31]. We show that many facts that are known about Datalog programs for finite structures extend to ω-categorical structures.

4.1. Countably categorical structures

A countable structure Γ is called *ω-categorical* if all countable models of the first-order theory of Γ are isomorphic to Γ. The following is a well-known and fundamental connection that shows that ω-categoricity of Γ is a property of the automorphism group of Γ, without reference to concepts from logic (see [31]). The *orbit* of an n-tuple $\bar{a}$ from Γ is the set $\{\alpha(\bar{a}) \mid \alpha \text{ is an automorphism of } \Gamma\}$.

Theorem 3 *(Engeler, Ryll-Nardzewski, Svenonius). (See e.g. [31].) The following properties of a countable structure Γ are equivalent*:

1. *the structure Γ is ω-categorical*;
2. *for each $n \geqslant 1$, there are finitely many orbits of n-tuples in the automorphism group of Γ*;
3. *for each $n \geqslant 1$, there are finitely many inequivalent first-order formulas with n free variables over Γ.*

Examples. An example of an ω-categorical directed graph is the set of rational numbers with the dense linear order $(\mathbb{Q}, <)$ [31]. The CSP for this structure is digraph acyclicity.

Another important example is the *universal triangle free graph* $\not\triangle$. This structure is the up to isomorphism unique countable K_3-free graph with the following *extension property*: whenever S is a subset and T is a disjoint independent subset of the vertices in $\not\triangle$, then $\not\triangle$ contains a vertex $v \notin S \cup T$ that is linked to no vertex in S and to all vertices in T. Since the extension property can be formulated by an (infinite) set of first-order sentences, it follows that $\not\triangle$ is ω-categorical [31]. The structure $\not\triangle$ is called the *universal* triangle free graph, because every other countable triangle free graph embeds into $\not\triangle$. The problem CSP($\not\triangle$) is the problem to decide whether a given graph does not contain triangles. This problem is clearly polynomial-time tractable; however, it cannot be formulated as a constraint satisfaction problem with a finite template [26,42].

The following lemma states an important property of ω-categorical structures needed several times later. The proof contains a typical proof technique for ω-categorical structures.

Lemma 2. *Let Γ be a finite or infinite ω-categorical structure with relational signature τ, and let Δ be a countable relational structure with the same signature τ. If there is no homomorphism from Δ to Γ, then there is a finite substructure of Δ that does not homomorphically map to Γ.*

Proof. Suppose every finite substructure of Δ homomorphically maps to Γ. We show the contraposition of the lemma, and prove the existence of a homomorphism from Δ to Γ. Let $a_1, a_2, \ldots$ be an enumeration of Δ. We construct a directed acyclic graph with finite out-degree, where each node lies on some level $n \geqslant 0$. The nodes on level n are equivalence classes of homomorphisms from the substructure of Δ induced by $a_1, \ldots, a_n$ to Γ. Two such homomorphisms f and g are equivalent, if there is an automorphism α of Γ such that $f\alpha = g$. Two equivalence classes of homomorphisms on level n and $n+1$ are adjacent, if there are representatives of the classes such that one is a restriction of the other. Theorem 3 asserts that Γ has only finitely many orbits of k-tuples, for all $k \geqslant 0$ (clearly, this also holds if Γ is finite). Hence, the constructed directed graph has finite out-degree. By assumption, there is a homomorphism from the structure induced by $a_1, a_2, \ldots, a_n$ to Γ for all $n \geqslant 0$, and hence the directed graph has vertices on all levels. König's lemma asserts the existence of an infinite path in the graph, which can be used to inductively define a homomorphism h from Δ to Γ as follows.

The restriction of h to $\{a_1, \ldots, a_n\}$ will be an element from the n-th node of the infinite path in G. Initially, this is trivially true if h is restricted to the empty set. Suppose h is already defined on $a_1, \ldots, a_n$, for $n \geqslant 0$. By construction of the infinite path, we find representatives h_n and h_{n+1} of the n-th and the $(n+1)$-st element on the path such that h_n is a restriction of h_{n+1}. The inductive assumption gives us an automorphism f of Γ such that $f(h_n(x)) = h(x)$ for all $x \in \{a_1, \ldots, a_n\}$. We

set $h(a_{n+1})$ to be $f(h_{n+1}(a_{n+1}))$. The restriction of h to $a_1, \ldots, a_{n+1}$ will therefore be a member of the $(n+1)$-st element of the infinite path. The operation f defined in this way is indeed a homomorphism from Δ to Γ. □

4.2. Canonical Datalog programs

In this section we define the canonical Datalog program of an ω-categorical structure Γ with finite relational signature τ. We will later prove in Section 4.3 that CSP(Γ) can be solved by an (l,k)-Datalog program if and only if the canonical (l,k)-Datalog program solves the problem.

For *finite* τ-structures T the canonical Datalog program for T was defined in [26]. Our definition generalizes this definition to ω-categorical structures Γ. The *canonical* (l,k)*-Datalog program* for Γ contains an IDB for every at most l-ary primitive positive definable relation in Γ. The empty 0-ary relation serves as *false* (this relation is primitive positive definable unless CSP(Γ) is trivial in the sense that every instance has a solution; our definition applies to non-trivial CSPs only). The input relation symbols are precisely the relation symbols from τ.

Let Γ' be the expansion of Γ by all at most l-ary primitive positive definable relations in Γ. It is a well-known consequence of Theorem 3 that first-order expansions of ω-categorical structures, and hence in particular the structure Γ', are also ω-categorical. The new relations of Γ' will be the IDBs and the relations that were already present in Γ are the EDBs of the canonical Datalog program. Theorem 3 also asserts that over Γ' there is a finite number of inequivalent formulas $\Psi(\bar{x})$ of the form

$$\big(\exists \bar{y}\big(\psi_1(\bar{x},\bar{y}) \wedge \cdots \wedge \psi_j(\bar{x},\bar{y})\big)\big) \to R(\bar{x})$$

having at most k variables, where $\psi_1, \ldots, \psi_j$ are atomic formulas of the form $R_1(\bar{z}_1), \ldots, R_j(\bar{z}_j)$ for IDBs or EDBs $R_1, \ldots, R_j$ and an IDB R. For each of these inequivalent implications $\Psi(\bar{x})$ we introduce a rule

$$R(\bar{x}) :- R_1(\bar{z}_1), \ldots, R_j(\bar{z}_j)$$

into the canonical Datalog program if $\forall \bar{x}.\, \Psi(\bar{x})$ is valid in Γ'. In other words, we introduce this rule if $R(\bar{x})$ is implied by $\exists \bar{y}(\psi_1(\bar{x},\bar{y}) \wedge \cdots \wedge \psi_j(\bar{x},\bar{y}))$ in Γ'. Since there are finitely many implications Ψ that are pairwise inequivalent in Γ', the canonical (l,k)-Datalog program is finite.

Observe that the final stage of the evaluation of the canonical Datalog program Π on a given instance S of CSP(Γ) gives rise to an instance S' of CSP(Γ') (where Γ' is as defined in the previous paragraph), namely the expansion of S computed in the last step of the evaluation of Π on S: since the IDBs of Π are relations from Γ', the structure computed at the last step of the evaluation of Π on S is an instance of CSP(Γ').

The following is easy to see.

Proposition 4. *Let Γ be an ω-categorical structure with finite relational signature. Then the canonical (l,k)-Datalog program for Γ is sound for* CSP(Γ).

Proof. We have to show that if the canonical (l,k)-Datalog program derives *false* on a given instance S, then S is unsatisfiable. We claim that when the canonical Datalog program derives $R(\bar{c})$ for some tuple $\bar{c} = (c_1, \ldots, c_d)$ of elements of S, and the IDB R has been introduced for the relation with the primitive positive formula $\phi(x_1, \ldots, x_d)$ over Γ, then for all homomorphisms f from S to Γ we have that Γ satisfies $\phi(f(c_1), \ldots, f(c_d))$. This follows by a straightforward induction over the evaluation of canonical Datalog programs, using the fact that the rules of the canonical Datalog program have been introduced for valid implications in the expansion Γ' of Γ by all at most l-ary primitive positive definable relations in Γ. Now, if the canonical (l,k)-program for Γ derives *false* on an instance S of CSP(Γ), then this shows that there is no homomorphism from S to Γ, and hence that S is unsatisfiable. □

4.3. Datalog for countably categorical structures

The following theorem is the promised link between Datalog, the existential pebble game, finite variable logics, and hypertree duality for ω-categorical structures. We present it in its most general form with both parameters l and k. The assumption of ω-categoricity will be used for the transition from item 2 to item 3 below (note that the canonical Datalog program is only defined for ω-categorical structures).

Theorem 4. *Let Γ be a ω-categorical structure with a finite relational signature τ, and let A be a finite τ-structure. Then for all l, k with $l \leqslant k$ the following statements are equivalent.*

1. *Every sound (l,k)-Datalog program for CSP(Γ) does not derive false on A.*
2. *The canonical (l,k)-Datalog program for Γ does not derive false on A.*
3. *Duplicator has a winning strategy for the existential (l,k)-pebble game on A and Γ.*
4. *All sentences in $L^{l,k}$ that hold in A also hold in Γ.*
5. *Every finite τ-structure with a core of treewidth at most (l,k) that homomorphically maps to A also homomorphically maps to Γ.*

Proof. The implication from 1 to 2 follows from Proposition 4.

To show that 2 implies 3, we define a winning strategy for Duplicator as follows. Let Γ' be the expansion of Γ by all at most l-ary primitive positive definable relations, and let A' be the instance of $\mathrm{CSP}(\Gamma')$ computed by the canonical (l,k)-Datalog program for Γ on input A. Then the strategy for Duplicator contains all those partial mappings $f : A \to \Gamma$ with domain D of size at most k such that for every relation $R(x_1,\dots,x_d)$ that holds in A' on elements $x_1,\dots,x_d \in D$, the tuple $(f(x_1),\dots,f(x_d))$ belongs to R in Γ'.

By construction, $\mathcal{H}$ contains only partial homomorphisms and is non-empty (since *false* is not derived, $\mathcal{H}$ contains the partial mapping with the empty domain). We shall prove that $\mathcal{H}$ has the (l,k)-extension property, and omit the easier proof that $\mathcal{H}$ is closed under restrictions. Let h be a function with domain $v_1,\dots,v_{l'}$ of size at most l and let $D = \{v_1,\dots,v_{l'},v_{l'+1},\dots,v_{k'}\}$ be a superset of $\{v_1,\dots,v_{l'}\}$ of size at most k. Let T be the k'-ary relation that contains all those tuples $(b_1,\dots,b_{k'}) \in D^{k'}_{\Gamma}$ such that $(b_{i_1},\dots,b_{i_r}) \in R^{\Gamma'}$ for every $R(v_{i_1},\dots,v_{i_r})$ derived in A' on variables $v_{i_1},\dots,v_{i_r}$ from D. Consider the following rule with variables $x_1,\dots,x_{k'}$ of the canonical Datalog program. The body of the rule contains for each IDB R and for all tuples $(v_{i_1},\dots,v_{i_r}) \in R^{A'}$ such that $v_{i_1},\dots,v_{i_r} \in D$ the atomic predicate $R(x_{i_1},\dots,x_{i_r})$. The head of the rule is $S(x_1,\dots,x_{l'})$ where $S^{\Gamma'}$ is the projection of the relation T to the first l' arguments. The instantiation $x_i \to v_i$, $i = 1,\dots,k'$, allows to derive $S(v_1,\dots,v_{l'})$ by this rule, and, by the definition of $\mathcal{H}$, $(h(v_1),\dots,h(v_{l'}))$ belongs to $S^{\Gamma'}$. By the definition of $S^{\Gamma'}$, there exist $b_{l'+1},\dots,b_{k'}$ such that $(h(v_1),\dots,h(v_{l'}),b_{l'+1},\dots,b_{k'})$ belongs to T. Hence, if we extend h by $v_i \to b_i$ for i from $l'+1,\dots,k'$ we obtain the desired function.

Next, we show the implication from 3 to 4. The proof closely follows the corresponding proof for finite structures given in [35], with the important difference that we have both parameters l and k in our proof, whereas previously the results have only been stated with the parameter k.

Suppose Duplicator has a winning strategy $\mathcal{H}$ for the existential (l,k)-pebble game on A and Γ. Let ϕ be a τ-sentence from $L^{l,k}$ that holds in A. We have to show that ϕ also holds in Γ. For that, we prove by induction on the syntactic structure of $L^{l,k}$ formulas that

if $\psi(v_1,\dots,v_m)$ is an $L^{l,k}$ formula that is an l-bounded conjunction or has at most l free variables (i.e., $m \leqslant l$), then for all $h \in \mathcal{H}$ and all elements $a_1,\dots,a_m$ from the domain of h, if A satisfies $\psi(a_1,\dots,a_m)$, then Γ satisfies $\psi(h(a_1),\dots,h(a_m))$.

Clearly, choosing $m = 0$, this implies that ϕ holds in Γ. The base case of the induction is obvious, since atomic formulas are preserved under homomorphisms. Next, suppose that $\psi(v_1,\dots,v_m)$ is an l-bounded conjunction of a set of formulas Ψ. Then each formula in Ψ either has at most l free variables, or is quantifier-free. In both cases we can use the inductive hypothesis, and the inductive step follows directly.

Assume that the formula $\psi(v_1,\dots,v_m)$ is of the form $\exists u_1,\dots,u_n.\,\chi(v_1,\dots,v_m,u_1,\dots,u_n)$. Since ψ is from $L^{l,k}$, we know that $n+m \leqslant k$. If $m > l$, there is nothing to show. Otherwise, we choose χ and n such that n is largest possible. Therefore, χ is either an l-bounded conjunction or an atomic formula. We will use the inductive hypothesis for the formula $\chi(v_1,\dots,v_m,u_1,\dots,u_n)$. Let h be a homomorphism in $\mathcal{H}$. We have to show that if $a_1,\dots,a_m$ are arbitrary elements from the domain of h such that A satisfies $\psi(a_1,\dots,a_m)$, then Γ satisfies $\psi(h(a_1),\dots,h(a_m))$.

Since A satisfies $\exists u_1,\dots,u_n.\,\chi(a_1,\dots,a_m)$, there exist $a_{m+1},\dots,a_{m+n}$ such that A satisfies $\chi(a_1,\dots,a_m,a_{m+1},\dots,a_{m+n})$. Consider the restriction h^* of h to the subset $\{a_1,\dots,a_m\}$ of the domain of h. Because of the first property of winning strategies $\mathcal{H}$, the homomorphism h^* is in $\mathcal{H}$. Since $m \leqslant l$, we can apply the forth property of $\mathcal{H}$ to h^* and $a_{m+1},\dots,a_{m+n}$, and there are $b_1,\dots,b_n$ such that the extension h' of h^* with domain $\{a_1,\dots,a_{m+n}\}$ that maps a_{m+i} to b_i is in $\mathcal{H}$. By applying the induction hypothesis to $\chi(v_1,\dots,v_m,u_1,\dots,u_n)$ and to h', we infer that Γ satisfies $\chi(h'(a_1),\dots,h'(a_{m+n}))$, and hence Γ satisfies $\psi(h(a_1),\dots,h(a_m))$.

4 implies 5: Let T be a finite τ-structure T whose core T' has treewidth at most (l,k) such that T homomorphically maps to A. By Lemma 1 there exists an $L^{l,k}$-sentence ϕ such that ϕ holds in a structure B if and only if T' homomorphically maps to B. In particular, ϕ must hold in A. Then 4 implies that ϕ holds in Γ, and therefore T' homomorphically maps to Γ. But then we can compose the homomorphism from T to T' and the homomorphism from T' to Γ to obtain the desired homomorphism from T to Γ.

We finally show that 5 implies 1. Assume 5, and suppose for contradiction that there is a sound (l,k)-Datalog program Π for Γ that derives *false* on A. The idea is to use the 'derivation tree of *false*' to construct a τ-structure S of treewidth at most (l,k) that homomorphically maps to A, but not to Γ. The construction proceeds by induction over the evaluation of Π on A. Suppose that $R_0(y^0_1,\dots,y^0_{k_0})$ is an atomic formula derived by Π on A from previously derived atomic formulas $R_1(y^1_1,\dots,y^1_{k_1}),\dots,R_s(y^s_1,\dots,y^s_{k_s})$. We will prove that there exists a structure S_0 with distinguished vertices $v^0_1,\dots,v^0_{k_0}$ and an (l,k)-tree G_0 such that

1. the Gaifman graph of S_0 is a (not necessarily induced) subgraph of G_0,
2. $v^0_1,\dots,v^0_{k_0}$ induce a clique in G_0,
3. there is a homomorphism from S_0 to A that maps v^0_i to y^0_i for every $1 \leqslant i \leqslant k_0$, and
4. the program Π derives $R_0(v^0_1,\dots,v^0_{k_0})$ on S_0.

Let $i \in \{1,\dots,s\}$. If R_i is an IDB, then let S_i, $v_1^i,\dots,v_{k_i}^i$, and G_i be given by the inductive hypothesis. If R_i is an EDB, we create fresh vertices $v_1^i,\dots,v_{k_i}^i$, and define S_i to be the following structure with vertices $v_1^i,\dots,v_{k_i}^i$. The relation R_i in S_i equals $\{(v_1^i,\dots,v_{k_i}^i)\}$, and all other relations in S_i are empty. Clearly, $\{v_1^i,\dots,v_{k_i}^i\}$ induces a clique in the Gaifman graph of S_i, and the Gaifman graph of S_i is a partial (l,k)-tree.

Now, the structure S_0 has the distinguished vertices $v_1^0,\dots,v_{k_0}^0$, and is obtained from the τ-structures $S_1,\dots,S_s$ as follows. We start from the disjoint union of $S_1,\dots,S_s$. When $y_j^i = y_s^r$ for $i,r \in \{0,\dots,s\}$, $j \in \{1,\dots,k_i\}$, and $s \in \{1,\dots,k_r\}$, then we identify v_j^i and v_s^r. To define G_0 we form a disjoint union of $G_1,\dots,G_s$ and the isolated nodes $v_1^0,\dots,v_{k_0}^0$, and do the same node identifications as before. We finally add an edge for every pair of distinct vertices in $v_1^0,\dots,v_{k_0}^0$. The resulting graph, G_0, satisfies the requirements of the claim. Observe that since Π derives $R_1(v_1^1,\dots,v_{k_1}^1),\dots,R_s(v_1^s,\dots,v_{k_s}^s)$ on S_0 by inductive assumption, it also derives $R_0(v_1^0,\dots,v_{k_0}^0)$ on S_0.

In this fashion we proceed for all inference steps of the Datalog program. Let S be the resulting structure for the final derivation of false. It has treewidth at most (l,k), and maps to S, but does not map to Γ, since Π (which is sound) derives also *false* on S. □

4.4. Application to constraint satisfaction

We discuss an important consequence of Theorem 4 with many concrete applications: we prove that CSP(Γ) for ω-categorical Γ is tractable if the input is restricted to instances of treewidth at most (l,k). In fact, for the tractability result we only have to require that the *cores* of the input structures have bounded treewidth. The statement where we only require that the core of input structures has treewidth at most (l,k) is considerably stronger (also see [28]); the corresponding statement for finite structures and single parameter k has been observed in [22].

Corollary 1. *Let Γ be an ω-categorical structure with finite relational signature τ. Then every instance A of* CSP(Γ) *whose core has treewidth at most (l,k) can be solved in polynomial time by the canonical (l,k)-Datalog program.*

Proof. It is clear that an (l,k)-Datalog program can be evaluated on a (finite) instance A of CSP(Γ) in polynomial time. If the canonical (l,k)-Datalog program derives *false* on A, then, because the canonical Datalog program is always sound, the instance A is not homomorphic to Γ. Now, suppose that the canonical Datalog program does not derive *false* on a finite structure A whose core has treewidth at most (l,k). Then, by Theorem 4, every τ-structure whose core has treewidth at most (l,k) that homomorphically maps to A also homomorphically maps to Γ. This holds in particular for A itself, and hence A is homomorphic to Γ. □

The following direct consequence of Theorem 4 yields other characterizations of bounded Datalog width.

Theorem 5. *Let Γ be a ω-categorical structure with a finite relational signature τ. Then for all l, k with $l \leqslant k$ the following statements are equivalent.*

1. *There is an (l,k)-Datalog program that solves* CSP(Γ).
2. *The canonical (l,k)-Datalog program solves* CSP(Γ).
3. *For all finite τ-structures A, if Duplicator has a winning strategy for the existential (l,k)-pebble game on A and Γ, then A is in* CSP(Γ).
4. *For all finite τ-structures A, if all sentences in $L^{l,k}$ that hold in A also hold in Γ, then A homomorphically maps to Γ.*
5. *For all finite τ-structures A, if every finite τ-structure S of treewidth at most (l,k) that homomorphically maps to A also homomorphically maps to Γ, then A homomorphically maps to Γ.*
6. *There is a set $\mathcal{N}$ of finite structures of treewidth at most (l,k) such that every finite τ-structure A is homomorphic to Γ if and only if no structure in $\mathcal{N}$ is homomorphic to A.*

Proof. To prove the implication from 1 to 2, suppose that an (l,k)-Datalog program Π solves CSP(Γ), and let S be an instance of CSP(Γ). If the canonical (l,k)-Datalog program derives *false* on S, then by Proposition 4 the structure S is not homomorphic to Γ. Otherwise, since Π is sound, the implication from 2 to 1 in Theorem 4 shows that the canonical (l,k)-Datalog program does not derive *false* on S as well. Hence, the canonical Datalog program solves CSP(Γ).

The implications $2 \Rightarrow 3 \Rightarrow 4 \Rightarrow 5 \Rightarrow 1$ are straightforward consequences of Theorem 4.

To show that 5 implies 6, let $\mathcal{N}$ be the set of all those structures of treewidth at most (l,k) that does not homomorphically map to Γ. Let A be a finite τ-structure. If A homomorphically maps to Γ, then clearly there is no structure C in $\mathcal{N}$ that maps to A, because then C would also map to Γ, a contradiction to the definition of $\mathcal{N}$. Conversely, suppose that no structure in $\mathcal{N}$ homomorphically maps to A. In other words, every structure that homomorphically maps to A also maps to Γ. Using 5, this implies that A homomorphically maps to Γ.

Finally, 6 implies 5. Let $\mathcal{N}$ be such that it satisfies the conditions of item 6. It follows that all structures in $\mathcal{N}$ do not map homomorphically to Γ. Let A be a finite τ-structure such that every finite τ-structure S of treewidth at most (l,k) that homomorphically maps to A also homomorphically maps to Γ. In particular, no structure in $\mathcal{N}$ homomorphically maps to A. Therefore, A homomorphically maps to Γ. □

5. 1-Datalog, MMSNP, and constraint satisfaction

A Datalog program of width one accepts a class of structures that can be described by a sentence of a fragment of existential second-order logic called *monotone monadic SNP without inequalities* (*MMSNP*). We show that every problem in MMSNP that is closed under disjoint unions can be formulated as the constraint satisfaction problem for an ω-categorical template. It follows that for every infinite structure Γ with finite relational signature, if CSP(Γ) has Datalog width one, then CSP(Γ) can also be formulated as a constraint satisfaction problem with an ω-categorical template.

An *SNP sentence* is an existential second-order sentence with a universal first-order part. The first-order part might contain the existentially quantified relation symbols and additional relation symbols from a given signature τ (the *input* relations). We shall assume that SNP formulas are written in *negation normal form*, i.e., the first-order part is in prenex normal form, and the quantifier-free part is in conjunctive normal form where each disjunction is written as a negated conjunction of positive and negative literals. It is well known that every first-order formula is logically equivalent to a formula of this form. SNP sentences can be used to *describe* computational problems in the sense that an SNP sentence Φ is valid on a structure A if and only if A is a yes-instance of the respective computational problem. The *class* SNP consists of all problems on relational τ-structures that can be described by an SNP sentence.

The class *MMSNP*, defined by Feder and Vardi, is the class of problems that can be described by an SNP sentence Φ that satisfies three additional requirements:

- the existentially quantified relations in Φ are *monadic*, that is, unary,
- Φ is *monotone*, i.e., every input relation symbol occurs *negatively* in Φ, and
- Φ does not contain inequalities.

Every problem in MMSNP is equivalent under randomized Turing reductions to a constraint satisfaction problem with a finite template [26]; a deterministic reduction was announced by Kun [36]. It is easy to see that MMSNP contains all constraint satisfaction problems with finite templates. Thus, MMSNP has a complexity dichotomy (meaning that every problem in MMSNP is polynomial-time solvable or NP-complete) if and only if the class of all finite-domain CSPs has a dichotomy.

It has already been observed by Feder and Vardi [26] that $(1,k)$-Datalog is contained in the class MMSNP. For a proof, introduce an existentially quantified unary predicate for each of the unary IDBs in the Datalog program. It is then straightforward to translate the rules of the Datalog program into universal first-order formulas with at most k first-order variables.

We now want to prove that every problem in MMSNP can be formulated as a constraint satisfaction problem with a countably categorical template. In full generality, this cannot be true because constraint satisfaction problems are always closed under disjoint union. A simple example of an MMSNP problem not closed under disjoint union is the one defined by the formula $\forall x, y\ \neg(P(x) \wedge Q(x))$. Hence, we shall assume that we are dealing with a problem in MMSNP that is *closed under disjoint union*.

To prove the claim under this assumption, we need a recent model-theoretic result of Cherlin, Shelah and Shi [18]. Let $\mathcal{N}$ be a finite set of finite structures with a relational signature τ. In this paper, a τ-structure Δ is called $\mathcal{N}$-*free* if there is no homomorphism from any structure in $\mathcal{N}$ to Δ. A structure Γ in a class of countable structures $\mathcal{C}$ is called *universal* for $\mathcal{C}$, if it contains all structures in $\mathcal{C}$ as an induced substructure.

Theorem 6 *(of [18]). Let $\mathcal{N}$ be a finite set of finite connected τ-structures. Then there is an ω-categorical structure Δ that is universal for the class of all countable $\mathcal{N}$-free structures.*

Cherlin, Shelah and Shi proved this statement for (undirected) graphs, but the proof does not rely on this assumption on the signature, and works for arbitrary relational signatures. The statement in its general form also follows from a result in [19]. We use the ω-categorical structure Δ to prove the following.

Theorem 7. *Every problem in MMSNP that is closed under disjoint unions can be formulated as* CSP(Γ) *with an ω-categorical template Γ.*

Proof. Let Φ be an MMSNP sentence with signature τ, written in negation normal form, whose class $\mathcal{M}$ of finite models is closed under disjoint unions. We have to find an ω-categorical τ-structure Γ such that $\mathcal{M}$ equals CSP(Γ). Let $P_1, \ldots, P_k$ be the existential monadic predicates in Φ. For each existential monadic relation P_i we introduce a relation symbol P_i', and replace negative literals of the form $\neg P_i(x)$ in Φ by $P_i'(x)$. We shall denote the formula obtained after this transformation by Φ'. Let τ' be the signature containing the input relations from τ, the existential monadic relations P_i, and the symbols P_i' for the negative occurrences of the existential relations. We define $\mathcal{N}$ to be the set of τ'-structures containing for each

clause $\neg(L_1 \wedge \cdots \wedge L_m)$ in Φ' the canonical database [17] of $(L_1 \wedge \cdots \wedge L_m)$. We shall use the fact that a τ'-structure S satisfies a clause $\neg(L_1 \wedge \cdots \wedge L_m)$ if and only if the canonical database of $(L_1 \wedge \cdots \wedge L_m)$ is not homomorphic to S.

We can assume without loss of generality that Φ is minimal in the sense that if we remove a literal from some of the clauses the formula obtained is inequivalent. We shall show that then all structures in $\mathcal{N}$ are connected. Let us suppose that this is not the case. Then there is a clause C in Φ that corresponds to a non-connected structure in $\mathcal{N}$. The clause C can be written as $\neg(E \wedge F)$ where the set X of variables in E and the set Y of variables in F do not intersect. Consider the formulas Φ_E and Φ_F obtained from Φ by replacing C by $\neg E$ and C by $\neg F$, respectively. By minimality of Φ there is a structure M_E that satisfies Φ but not Φ_E, and similarly there exists a structure M_F that satisfies Φ but not Φ_F. By assumption, the disjoint union M of M_E and M_F satisfies Φ. Then there exists a τ''-expansion M'' of M where $\tau'' := \tau \cup \{P_1, \ldots, P_k\}$ that satisfies the first-order part of Φ. Consider the substructures M''_E and M''_F of M'' induced by the vertices of M_E and M_F. We have that M''_E does not satisfy the first-order part of Φ_E (otherwise M_E would satisfy Φ_E). Consequently, there is an assignment s_E of the universal variables that falsifies some clause. This clause must necessarily be $\neg E$ (since otherwise M'' would not satisfy the first-order part of Φ). By similar reasoning we can infer that there is an assignment s_F of the universal variables of Φ to elements of M_F that falsifies $\neg F$. Finally, fix any assignment s that coincides with s_E over X and with s_F over Y (such an assignment exists because X and Y are disjoint). Clearly, s falsifies C and M does not satisfy Φ, a contradiction. Hence, we shall assume that every structure in $\mathcal{N}$ is connected.

Then Theorem 6 asserts the existence of an $\mathcal{N}$-free ω-categorical τ'-structure Δ that is universal for all $\mathcal{N}$-free structures. We use Δ to define the template Γ for the constraint satisfaction problem. To do this, let Δ' be the restriction of Δ to those elements that have the property that for all existential monadic predicates P_i either P_i or P'_i holds (but not both P_i and P'_i). Let Γ be the reduct of Δ' that only contains the input relations from τ. It is well known (see e.g. Theorem 7.3.8 in [31]) that reducts and first-order restrictions of ω-categorical structures are again ω-categorical. Hence, Γ is ω-categorical.

We claim that a τ-structure S satisfies Φ if and only if S homomorphically maps to Γ. Suppose there is a homomorphism h from S to Γ. Let S' be the τ'-expansion of S such that for each $i \in \{1, \ldots, k\}$ the relation $P_i(x)$ holds in S' if and only if $P_i(h(x))$ holds in Δ', and $P'_i(x)$ holds in S' if and only if $P'_i(h(x))$ holds in Δ'. Clearly, h defines a homomorphism from S' to Δ' and also from S to Δ. In consequence, none of the structures from $\mathcal{N}$ maps to S'. Hence, the τ''-reduction of S' satisfies all the clauses of the first-order part of Φ and hence S satisfies Φ.

Conversely, let S be a structure satisfying Φ. Thus, there exists a τ'-expansion S' of S that satisfies the first-order part of Φ' and where for every element x exactly one of $P_i(x)$ or $P'_i(x)$ holds. Clearly, no structure in $\mathcal{N}$ is homomorphic to the expanded structure, and by universality of Γ the τ'-structure S' is an induced substructure of Δ. Since for every point of S' exactly one of P_i and P'_i holds, S' is also an induced substructure of Δ'. Therefore, S is homomorphic to Γ. This completes the proof. □

In particular, we proved the following.

Corollary 2. *Every problem in $(1, k)$-Datalog that is closed under disjoint unions can be formulated as a constraint satisfaction problem with an ω-categorical template.*

Example 1. The following computational problem is an example of a CSP in MMSNP that cannot be described with a finite template [41] and that is not in (l, k)-Datalog for all $1 \leqslant l \leqslant k$. We are given a finite graph S, and we want to test whether we can partition the vertices of S into two parts such that each part is triangle-free. It is easy to formulate this problem in MMSNP. Hence, Corollary 2 implies that it can also be formulated as a CSP with an ω-categorical template. To illustrate, we describe such a template explicitly: Take two copies C_1 and C_2 of $\not\triangle$, and add an undirected edge between all vertices in C_1 and all vertices in C_2. The corresponding CSP is NP-hard [1].

6. Bounded width

In this section we characterize some families of ω-categorical templates whose CSPs have bounded width. Our results generalize known algebraic characterizations of Datalog width for constraint satisfaction with finite templates. However, not all results remain valid for infinite templates: it is well known [26] that the constraint satisfaction of a finite template has Datalog width one if and only if the so-called *arc-consistency procedure* solves the problem. This is no longer true for infinite templates. We characterize both width one and the expressive power of the arc-consistency procedure for infinite ω-categorical templates, and present an example that shows that the two concepts are different. We also present an algebraic characterization of *strict width l*, a concept introduced by Feder and Vardi [26].

6.1. Width zero

An example of a template whose constraint satisfaction problem has width 0 is the universal triangle-free graph $\not\triangle$. Since there is a primitive positive sentence that states the existence of a triangle in a graph, and since every graph without a triangle is homomorphic to $\not\triangle$, there is a Datalog program of width 0 that solves CSP($\not\triangle$). In general, it is easy to see that a constraint satisfaction problem has width 0 if and only if there is a finite set of *homomorphic obstructions* for CSP(Γ), i.e.,

a finite set $\mathcal{N}$ of finite τ-structures such that every finite τ-structure A is homomorphic to Γ if and only if no substructure in $\mathcal{N}$ is homomorphic to A.

When Γ is a structure with finite relational signature τ, we say that CSP(Γ) is *first-order definable* if there exists a first-order τ-sentence Φ such that a finite τ-structure S homomorphically maps to Γ if and only if S satisfies Φ. It turns out that CSP(Γ) is first-order definable if and only if it has width 0. This can be seen as a reformulation of Rossman's theorem [44], which says that a first-order sentence ϕ is equivalent to an existential positive sentence if and only if the class of finite models of ϕ is closed under homomorphisms. For finite templates a characterization of first-order definable constraint satisfaction problems has been obtained in [38] building on work in [2,43]. Our discussion is summarized by the following theorem.

Theorem 8. *For every (not necessarily ω-categorical) template Γ the following are equivalent.*

1. CSP(Γ) *has a finite obstruction set*;
2. CSP(Γ) *has Datalog width* 0;
3. CSP(Γ) *is first-order definable.*

Moreover, if CSP(Γ) *is first-order definable we can always find an ω-categorical structure Γ' that has the same constraint satisfaction problem as Γ.*

Proof. The equivalence between items 1 and 2 has been discussed above. For the equivalence of 2 and 3, note that the complement of a CSP is closed under homomorphisms. Hence, Rossman's theorem implies that a CSP with an arbitrary infinite template has Datalog width 0 if and only if it is first-order definable. The last part of the statement is a special case of Corollary 2. □

6.2. Width one

Let Γ be an ω-categorical structure with relational signature τ, and Π be the canonical $(1,k)$-Datalog program for Γ, for some $k \geqslant 1$. By Corollary 2, the class of τ-structures accepted by Π is itself a CSP with an ω-categorical template, which we denote by $\Gamma(1,k)$.

Theorem 9. *Let Γ be ω-categorical. Then* CSP(Γ) *can be solved by a $(1,k)$-Datalog program if and only if there is a homomorphism from $\Gamma(1,k)$ to Γ.*

Proof. Let Π be the canonical $(1,k)$-Datalog program of Γ. Suppose first that there is a homomorphism from $\Gamma(1,k)$ to Γ. We show that Π solves CSP(Γ). Let A be an instance of CSP(Γ). If Π accepts A, then A homomorphically maps to $\Gamma(1,k)$, and therefore also to Γ. Otherwise, if Π does not accept A, then A does not map to Γ since Π is sound (Proposition 4).

For the opposite implication, suppose that there is a width $(1,k)$-Datalog program that solves CSP(Γ). By Theorem 5, the program Π also solves CSP(Γ). To show that $\Gamma(1,k)$ homomorphically maps to Γ, it suffices by Lemma 2 to show that every finite substructure A of the countable structure $\Gamma(1,k)$ homomorphically maps to Γ. Every finite substructure A of $\Gamma(1,k)$ is in particular homomorphic to $\Gamma(1,k)$, and thus accepted by Π. Since Π solves CSP(Γ), A homomorphically maps to Γ. □

6.3. Arc-consistency

The *arc-consistency procedure* (*AC*) is an algorithm for constraint satisfaction problems that is intensively studied in Artificial Intelligence (which is sometimes also called *hyperarc consistency* or *generalized arc consistency* to stress the fact that it can also deal with constraints of arity larger than two). It can be described as the subset of the canonical Datalog program of width one that consists of all rules with bodies containing at most one non-IDB. For finite templates T it is known that the arc-consistency procedure solves CSP(T) if and only if CSP(T) has width one [26]. For infinite structures, this is no longer true: consider for instance CSP($\not\triangleleft$), which has width 0, but cannot be solved by the arc-consistency procedure. The reason is that the width one canonical Datalog program for $\not\triangleleft$ has no non-trivial unary predicates, and we thus have to consider at least three relations in the input to infer that the input contains a triangle.

The following concept is crucial to understand the power of the arc-consistency procedure. Let Γ be an ω-categorical structure with finite relational signature τ. Since Γ is ω-categorical, there is only a finite number of primitive positive definable non-empty sets $O_1, \ldots, O_n$. We define the *definable subset structure* of Γ, which is the finite relational τ-structure with domain $\{O_1, \ldots, O_n\}$ where a k-ary relation R from τ holds on $O_{i_1}, \ldots, O_{i_k}$ iff for every $j \in \{1, \ldots, k\}$ and every vertex v_j in the orbit O_{i_j} there are vertices $v_1, \ldots, v_{j-1}, v_{j+1}, \ldots, v_k$ from $O_{i_1}, \ldots, O_{i_{j-1}}, O_{i_{j+1}}, \ldots, O_{i_k}$, respectively, such that R holds on $v_1, \ldots, v_k$ in Γ.

Lemma 3. *Let Γ be an ω-categorical structure with finite relational signature τ. Then for every instance S of* CSP(Γ) *the following two statements are equivalent*:

(1) *The arc-consistency procedure Π for Γ does not derive* false *on instance S.*
(2) *S is homomorphic to the definable subset structure of Γ.*

Proof. (1) → (2). Every unary relation that can be inferred by the arc-consistency procedure is definable by a primitive positive formula and hence is an element of $\{O_1, \dots, O_n\}$. For every variable u of S, let T^u be the subset of $\{O_1, \dots, O_n\}$ containing all those unary IDBs O_i, $1 \leqslant i \leqslant n$, such that $O_i(u)$ is derived by Π. By the structure of the rules of the arc-consistency algorithm, T^u is closed under intersection. Define h to be the mapping from D_S to $\{O_1, \dots, O_n\}$ that maps every variable u to the minimum element of T^u (with respect to set inclusion), which will be denoted by $\bigcap T^u$. We shall show that h is a homomorphism from S to the definable subset structure of Γ. Let $R \in \tau$, and let $(u_1, \dots, u_k)$ be a tuple of R^S. Then $(\bigcap T^{u_1}, \dots, \bigcap T^{u_k})$ is the image of this tuple under h. Fix any $j \in \{1, \dots, k\}$, and let O be the set containing all those v_j such that there are vertices $v_1, \dots, v_{j-1}, v_{j+1}, \dots, v_k$ from $\bigcap T^{u_1}, \dots, \bigcap T^{u_{j-1}}, \bigcap T^{u_{j+1}}, \dots, \bigcap T^{u_k}$, respectively, such that R holds on $v_1, \dots, v_k$ in Γ. Then O is primitive positive definable in Γ, and Π contains the rule

$$O(x_j) :- R(x_1, \dots, x_k), \bigcap T^{u_1}(x_1), \dots, \bigcap T^{u_{j-1}}(x_{j-1}), \bigcap T^{u_{j+1}}(x_{j+1}), \dots, \bigcap T^{u_k}(x_k)$$

which allows to derive $O(u_j)$. As $\bigcap T^{u_j} \subseteq O$ we conclude that $(\bigcap T^{u_1}, \dots, \bigcap T^{u_k})$ belongs to the relation R in the definable subset structure.

(2) → (1). Let h be a homomorphism from S to the definable subset structure of Γ. It is easy to prove by induction on the evaluation of Π on S that $h(u) \subseteq R$ for every $R(u)$ derived by Π. Hence, *false* cannot be derived by Π on S. □

Theorem 10. *Let Γ be an ω-categorical structure with finite relational signature. Then the arc-consistency procedure correctly decides* CSP(Γ) *if and only if the definable subset structure is homomorphic to Γ.*

Proof. Since the definable subset structure homomorphically maps to itself, the claim proven above shows that the arc-consistency procedure does not derive *false* on the definable subset structure. Hence, if the arc-consistency procedure solves CSP(Γ) then the definable subset structure homomorphically maps to Γ.

Conversely, suppose that there is a homomorphism h from the definable subset structure to Γ. To show that Π solves CSP(Γ), it suffices to show that an instance S where Π does not derive *false* homomorphically maps to Γ. By the claim proven above there is a homomorphism g from S to the definable subset structure of Γ. Composing g and h yields the desired homomorphism from S to Γ. □

Theorem 11. *Let Γ be an ω-categorical structure with finite relational signature. If* CSP(Γ) *is solved by the arc-consistency algorithm, then Γ is homomorphically equivalent to a finite structure.*

Proof. By Lemma 2 it suffices to show that arbitrary finite substructures S of Γ homomorphically map to the (finite!) definable subset structure. Since substructures of Γ are satisfiable instances of CSP(Γ), Π does not derive *false* on such a structure S. So by the claim above, S is homomorphic to the definable subset structure of Γ. □

7. Bounded strict width

The notion of strict width was introduced for finite domain constraint satisfaction problems by Feder and Vardi [26], and was defined in terms of the canonical Datalog program. In the terminology of the constraint satisfaction literature in Artificial Intelligence, strict width l is equivalent to *'strong l-consistency implies global consistency'*. Based on our generalization of the concept of canonical Datalog programs, we study the analogously defined concept of strict width l for ω-categorical structures.

The notion of strict width is defined as follows. Recall that the canonical (l, k)-Datalog program Π for CSP(Γ) receives as input an instance S of CSP(Γ) and returns an expansion S' of S over τ' where τ' is the vocabulary that contains τ as well as a predicate for every IDB of Π. The structure S' can be seen as an instance of CSP(Γ') where Γ' is the expansion of Γ by all at most l-ary primitive positive definable relations. The instance S' is called *globally consistent*, if every partial homomorphism, i.e., every homomorphism from an induced substructure of S' to Γ, can be extended to a homomorphism from S to Γ. If for some $k \geqslant l + 1 \geqslant 3$ all instances of CSP(Γ') that are computed by the canonical (l, k)-program are globally consistent, we say that Γ has *strict width l*. Note that strict width l implies width l, and hence CSP(Γ) can be solved in polynomial time when Γ has bounded strict width.

Also note that if Π derives *false* on input S, then S' does not have any partial homomorphisms to Γ', and hence S' is in this case by definition globally consistent. If the reader feels uneasy about calling unsatisfiable instances globally consistent, one might also define global consistence only for satisfiable instances; for strict width l we then require that the instances computed by the canonical (l, k)-program that do not contain the predicate *false* are globally consistent. These two definitions are clearly equivalent. With our definition we follow what is standard in the literature.

In this section we present a universal-algebraic characterization of strict width l for ω-categorical templates Γ. The algebraic approach rests on the notion of *polymorphisms*. Let Γ be a relational structure with signature τ. A *polymorphism*

is a homomorphism from Γ^l to Γ, for some l, where Γ^l is a relational τ-structure defined as follows. The vertices of Γ^l are l-tuples over elements from V_Γ, and k such l-tuples $(v_1^i, \dots, v_l^i)$, $1 \leqslant i \leqslant k$, are joined by a k-ary relation R from τ if $(v_j^1, \dots, v_j^k)$ is in R^Γ, for all $1 \leqslant j \leqslant l$.

We say that an operation f is a *near-unanimity operation* (short, *nu-operation*) if it satisfies the identities $f(x, \dots, x, y, x, \dots, x) = x$, i.e., in the case that the arguments have the same value x except at most one argument, the operation has the value x. We say that f is a *near-unanimity operation on A* if it satisfies the identities $f(x, \dots, x, y, x, \dots, x) = x$ for all $x, y \in A$.

Feder and Vardi [26] proved that a finite template Γ has an $(l+1)$-ary near-unanimity operation (in this case, they say that Γ has the $(l+1)$-*mapping property*) if and only if CSP(Γ) has strict width l. Another proof of this theorem was given in [32]. It is stated there that the proof extends to arbitrary infinite templates, if we want to characterize bounded strict width on instances of the constraint satisfaction problem that might be infinite. However, we would like to describe the complexity of constraint satisfaction problems with finite instances.

In fact, there are structures that do not have a nu-operation, but where Γ has bounded strict width. One example of such a structure is the universal triangle-free graph $\not\triangle$. A theorem by Larose and Tardif shows that every finite or infinite graph with a nu-operation is bipartite [39]. Since the universal triangle-free graph contains all cycles of length larger than three, it therefore cannot have a nu-operation. However, the universal triangle-free graph has strict width 2. Indeed, for any instance S accepted by the canonical $(2, 3)$-Datalog program, every partial mapping from S to $\not\triangle$ satisfying all the facts derived by the program (and in particular not containing any triangle) can be extended to a complete homomorphisms from S to $\not\triangle$ – this follows from the extension properties of the template.

Theorem 12 characterizes strict width l, $l \geqslant 2$, for constraint satisfaction with ω-categorical templates. We first need an intermediate result.

Lemma 4. *Let Γ be a τ-structure such that* CSP(Γ) *has strict width l and let $\tau_\equiv$ be the superset of τ in which we add a new binary relation symbol $\equiv$. Let $\Gamma_\equiv$ be the $\tau_\equiv$-expansion of Γ in which $\equiv$ is interpreted by the usual equality relation $\{(x, x) \mid x \in D_\Gamma\}$. Then* CSP($\Gamma_\equiv$) *has also strict width l.*

Proof. Let $\Pi_\equiv$ be the canonical (l, k)-program for CSP($\Gamma_\equiv$), and S be an instance of CSP($\Gamma_\equiv$). Let S' be the structure computed by $\Pi_\equiv$ on S. Let E be the smallest equivalence relation on the universe of S that contains $\equiv^S$. Let S/E be the τ-reduct of S obtained by factoring S by the equivalence relation E. More precisely, the universe of S/E are the equivalence classes of R, $\{E_a \mid a \in D_S\}$, where E_a denotes the E-class of a, and for every $R \in \tau$, say r-ary, $R^S = \{(E_{a_1}, \dots, E_{a_r}) \mid (a_1, \dots, a_r) \in R^S\}$. We now consider S/E as an instance of CSP(Γ). Let Π be the canonical (l, k)-program of Γ. It is easy to prove by induction on the evaluation of Π on S/E that if R is an IDB, say r-ary, and $R(E_{a_1}, \dots, E_{a_r})$ is derived by Π on S/E, then $R(a_1, \dots, a_r)$ is derived by $\Pi_\equiv$ on S. We have to show that S' is globally consistent. So suppose that there is a partial homomorphism h from S' to $\Gamma_\equiv$. Since $l \geqslant 2$ and $k \geqslant 3$, $\Pi_\equiv$ will be able to derive that all elements in the same E-class have to get the same value and hence, if h is a partial homomorphism then this implies that for all elements a, b in the domain of h that are E-related, $h(a) = h(b)$. Define h/E to be the partial mapping that maps every E_a with a in the domain of h to $h(a)$. By the definition of S and analysis on the predicates derived by Π on S carried out above we know that h_E is a partial homomorphism from S/E to Γ. Hence h can be extended to a full homomorphism h' from S/E to Γ. Finally, the mapping h' defined to be $h'(a) = (h/E)(E_a)$ is a homomorphism from S to Γ and hence also from S' to Γ'. □

One of the key properties of structures Γ with near unanimity polymorphisms is the following.

Lemma 5. *Let Γ be a relational ω-categorical structure with maximal arity k and an $(l+1)$-ary polymorphism f for every finite subset A such that f is a nuf on A. Let Γ' be the expansion of Γ by all l-ary primitive positive definable relations, and let S' be an instance of* CSP(Γ') *computed by the canonical (l, k)-Datalog program for Γ. Then every partial homomorphism h from S' to Γ' has the property that for every fact $R(u_1, \dots, u_r)$ in S' there exists a tuple $(d_1, \dots, d_r) \in R^{\Gamma'}$ such that $h(u_i) = d_i$ for all u_i where h is defined.*

Proof. Let $i_1, \dots, i_s$ be a list of the indices $i \in \{1, \dots, r\}$ such that $u_i \in D_S$, and let $j_1, \dots, j_t$ be a list of the other indices in $\{1, \dots, r\}$ (so we have $s + t = r$). We prove the statement by induction on s. For $s \leqslant l$, let R' be the IDB associated to the $\exists u_{j_1}, \dots, u_{j_t}. R(u_1, \dots, u_r)$ with free variables $u_{i_1}, \dots, u_{i_s}$. Since $R'(u_{i_1}, \dots, u_{i_s}) :- R(u_1, \dots, u_r)$ is a rule in Π, we have $(h(u_{i_1}), \dots, h(u_{i_s})) \in R'^{\Gamma'}$. Then the witnesses for the existentially quantified variables $u_{j_1}, \dots, u_{j_t}$ in Γ' along with $h(u_{i_1}), \dots, h(u_{i_s})$ determine the tuple $(d_1, \dots, d_r) \in R^{\Gamma'}$ with the desired property.

For $s \geqslant l+1$, consider for all $j \in \{i_1, \dots, i_{l+1}\}$ the tuple $b^j = (b_1^j, \dots, b_r^j) \in R^{\Gamma'}$ given inductively for the restriction of h to $D_S \setminus \{u_j\}$. Let g be an $(l+1)$-ary polymorphism which is a nuf on the set containing all elements in all tuples b^j. Then the tuple $(g(b_1^1, \dots, b_1^{l+1}), \dots, g(b_r^1, \dots, b_r^{l+1}))$ has the desired properties. □

The proof of the following theorem is based on ideas from [26] and [32].

Theorem 12. *Let Γ be an ω-categorical structure with relational signature τ of bounded maximal arity. Then the following are equivalent, for $l \geqslant 2$:*

1. *$\mathrm{CSP}(\Gamma)$ has strict width l.*
2. *For every finite subset A of Γ there is an $(l+1)$-ary polymorphism of Γ that is a nuf on A.*

Proof. We first show that 1 implies 2.

We assume that $\mathrm{CSP}(\Gamma)$ has strict width l, and prove that for every finite subset A of Γ there is a polymorphism of Γ that is an $(l+1)$-ary nuf on A. Let τ^A be the superset of τ that additionally contains a unary relation symbol R_a for each element a of A. Let Γ^A be the τ^A-expansion of Γ in which R_a is interpreted by the singleton relation $\{a\}$. Consider the set B of tuples $(a_0, \dots, a_l)$ in A^{l+1} that have identical entries $a_i = a$ except possibly at one exceptional position. Let Δ be the τ^A-expansion of Γ^{l+1} where R_a denotes the set of all tuples $(a, \dots, a, b, a, \dots, a)$ in B where at most one entry is not a. Every homomorphism from Δ to Γ^A is by construction a polymorphism of Γ that is a nuf on A. Lemma 2 shows that if every finite substructure S^A of Δ homomorphically maps to Γ^A, then Δ homomorphically maps to Γ^A as well.

Let S^A be any finite substructure of Δ, and let S be the τ-reduct of S^A, which we see as an instance of $\mathrm{CSP}(\Gamma)$. We show that there exists a homomorphism h from S to Γ that sends every tuple of the form $(a, \dots, a, b, a, \dots, a)$ in $B \cap D_S$ to a. Hence, h is also a homomorphism from S^A to Γ^A.

Let $\tau_{\equiv}$ be the superset of τ that additionally contains a new binary predicate $\equiv$, and let $\Gamma_{\equiv}$ be the expansion of Γ in which $\equiv$ is interpreted by the equality relation. Let T be the $(\tau_{\equiv})$-structure with domain $D_S \times \{0,1\}$ where an r-ary predicate $P \in \tau$ denotes

$$P^T := \big\{\big((a_1,0),\dots,(a_r,0)\big) \bigm| (a_1,\dots,a_r) \in P^S\big\}.$$

Furthermore,

$$\equiv^T := \big\{\big((a,0),(a,1)\big) \bigm| a \in S\big\}.$$

By Lemma 4, $\Gamma_{\equiv}$ has strict width l. Let $\Gamma'_{\equiv}$ be the expansion of $\Gamma_{\equiv}$ by all at most l-ary primitive positive definable relations, and let k be such that all instances of $\mathrm{CSP}(\Gamma'_{\equiv})$ computed by the canonical (l,k)-program $\Pi_{\equiv}$ are globally consistent. Let T' be the instance of $\mathrm{CSP}(\Gamma'_{\equiv})$ computed by $\Pi_{\equiv}$ on T. Now consider the partial assignment g defined on $(B \cap D_S) \times \{1\}$ that sends every tuple of the form $((a,1),\dots,(a,1),(b,1),(a,1),\dots,(a,1))$ to a. We shall see that g is a partial homomorphism from T' to $\Gamma'_{\equiv}$. Indeed, let $(\bar{a}_1,\dots,\bar{a}_r) \in R^{T'}$ be any tuple entirely contained in the domain of g. For every $j \in \{1,\dots,r\}$, the tuple $\bar{a}_j$ is of the from $((a_j,1),\dots,(a_j,1),(b_j,1),(a_j,1),\dots,(a_j,1))$. This tuple has necessarily been placed there by the Datalog program, and hence R is an IDB and has cardinality at most l. The pigeon-hole principle guarantees that there exists an index $i \in \{1,\dots,l+1\}$ such that for every $1 \leqslant j \leqslant r$ the i-th entry of $\bar{a}_j$ is precisely $(a_j,1)$. Since the i-th projection is a homomorphism from S to Γ, it cannot violate any fact derived by the canonical (l,k)-Datalog program and hence $(a_1,\dots,a_l) \in R^{\Gamma'_{\equiv}}$. Since T' is globally consistent this implies that g can be extended to a full homomorphism g' from T' to $\Gamma'_{\equiv}$. Finally we obtain the desired homomorphism $h : S \to D_\Gamma$ as $h(a_1,\dots,a_l) := g'((a_1,0),\dots,(a_l,0))$.

Next we show that 2 implies 1. Let k be larger than the maximal arity of the relations in τ, and at least $l+1$. Let Π be the canonical (l,k)-program for Γ, let Γ' be the expansion of Γ by all at most l-ary primitive positive definable relations, and let S' be the instance of $\mathrm{CSP}(\Gamma')$ computed by Π on S. We shall prove that every partial homomorphism with domain $\{v_1,\dots,v_i\}$, for $i < |S'|$, has an extension to any other element v of S' such that the extension is still a partial homomorphism from S' to Γ'. We prove this by induction on the size i of the domain of s.

For the case that $i \leqslant l$, let Ψ be the set of all atomic formulas of the form $R(\bar{u})$ that hold in S' and where all entries of $\bar{u}$ are from $\{v_1,\dots,v_i,v\}$, and let R' be the IDB associated to the primitive positive formula $\exists v \bigwedge \Psi$ with free variables $v_1,\dots,v_i$. Since each formula in Ψ is derived by Π on S, the predicate $R'(v_1,\dots,v_i)$ is also derived by Π on S. Since h preserves R', we have that $(h(v_1),\dots,h(v_i))$ satisfies $\exists v \bigwedge \Psi$; hence, there exists an extension of h to v such that the extension is a partial homomorphism from S' to Γ'.

For the induction step where $i \geqslant l+1$, select elements $w_1,\dots,w_{l+1}$ in $\{v_1,\dots,v_i\}$, and let h_j be the restriction of h where w_j is undefined, for $j \in \{1,\dots,l+1\}$. By induction, h_j can be extended to a homomorphism h'_j from the structure induced by $\{v_1,\dots,v_i,v\} \setminus \{w_j\}$ in S' to Γ'. For each $(u_1,\dots,u_r) \in R^{S'}$, Lemma 5 asserts the existence of a tuple $(b_1^j,\dots,b_r^j) \in R^{\Gamma'}$ such that $h'_j(u_i) = b_i^j$ for all u_i where h is defined. Let A be the finite set that contains all those elements b_i^j of Γ', for all tuples $(u_1,\dots,u_r)$ in all relations R of S'. Let g be an $(l+1)$-ary polymorphism of Γ' that is a nuf on A (observe that Γ and Γ' have the same polymorphisms). Define b to be $g(h'_1(v),\dots,h'_{l+1}(v))$. We claim that the extension h' of h mapping v to b is a homomorphism from the substructure induced by $\{v_1,\dots,v_i,v\}$ in S' to Γ'.

Let $(u_1,\dots,u_r) \in R^{S'}$ be arbitrary; we want to show that $(h'(u_1),\dots,h'(u_r)) \in R^{\Gamma'}$. Recall that $(b_1^j,\dots,b_r^j) \in R^{\Gamma'}$ is such that $h'_j(u_i) = b_i^j$ for all u_i where h is defined. Then the tuple $(g'(b_1^1,\dots,b_1^{l+1}),\dots,g'(b_r^1,\dots,b_r^{l+1}))$ is from $R^{\Gamma'}$. Moreover, we claim that $g'(b_s^1,\dots,b_s^{l+1}) = h'(u_s)$: if $u_s \in \{v_1,\dots,v_i\}$, note that for all but at most one j from $\{1,\dots,l+1\}$ we have that $b_s^j = h'_j(u_s) = h(u_s)$, and since g' is a nuf on the entries of the tuples b^j we obtain that $g'(b_s^1,\dots,b_s^{l+1}) = h(u_s) = h'(u_s)$.

Otherwise, if $u_s = v$, then $g'(b_s^1, \dots, b_s^{l+1}) = g'(h'_1(v), \dots, h_{l+1}(v)) = b = h'(v)$ by definition of h'. We conclude that $(h'(u_1), \dots, h'(u_r)) \in R^{S'}$. □

Note that in several papers including [6,7] and the conference version that precedes this one, condition 2 has been stated in a different but essentially equivalent way using the notion of quasi near-unanimity operation.[3]

We say that an operation f is a *quasi near-unanimity operation* (short, *qnu-operation*), if it satisfies the identities $f(x, \dots, x, y, x, \dots, x) = f(x, \dots, x)$, i.e., in the case that the arguments have the same value x except at one argument position, the operation has the value $f(x, \dots, x)$. In other words, the value y of the exceptional argument does not influence the value of the operation f. Several well-known temporal and spatial constraint languages have polymorphisms that are qnu-operations [7].

For every subset A of Γ, we say that an operation is *idempotent on* A if $f(a, \dots, a) = a$ for all $a \in A$. Hence, if a qnu-operation f is idempotent on the entire domain, then f is a near-unanimity operation. If a polymorphism f of Γ has the property that for every finite subset A of Γ there is an automorphism α of Γ such that $f(x, \dots, x) = \alpha(x)$ for all $x \in A$, we say that f is *oligopotent*.

Corollary 3. *Let Γ be a ω-categorical structure with finite relational signature τ, and let $l \geqslant 2$. Then the following are equivalent*:

1. CSP(Γ) *has strict width l.*
2. *For every finite subset A of Γ there is an $(l+1)$-ary polymorphism of Γ that is a nuf on A.*
3. *Γ has an oligopotent $(l+1)$-ary polymorphism that is a qnu-operation.*
4. *Every primitive positive formula is in Γ equivalent to a conjunction of at most l-ary primitive positive formulas.*

Proof. The equivalence of items 1 and 2 has been shown in Theorem 12, and the equivalence of 2 and 3 follows from a direct application of Lemma 2. The equivalence of 3 and 4 is shown in [6]. □

Concerning the condition of oligopotency in statement 3 of Corollary 3, we want to remark that for every ω-categorical structure Γ there is a template that has the same CSP and where all polymorphisms are oligopotent. It was shown in [5] that every ω-categorical structure is homomorphically equivalent to a *model-complete core* Δ, i.e., Δ has the property that for every finite subset A of the domain of Δ and for every *endomorphism* e of Δ (an endomorphism is a unary polymorphism) there exists an automorphism a of Δ such that $a(x) = e(x)$ for all $x \in A$. (Moreover, it is also known that Δ is unique up to isomorphism, and ω-categorical.)

Corollary 4. *Suppose that Δ is an ω-categorical model-complete core. Then Δ has strict width l if and only if Δ has an $(l+1)$-ary qnu-polymorphism.*

8. Notational link with the relation algebra perspective

This section does not present any new results; instead, it demonstrates how to translate our results into the terminology of the literature that uses *relation algebras* to formalize infinite-domain constraint satisfaction problems, used in particular in temporal and spatial reasoning.

8.1. Proper relation algebras

In Artificial Intelligence, relation algebras are used as a framework to formalize and study qualitative reasoning problems [24,30,37]. In fact, the so-called *network consistency problem* for a fixed relation algebra turns out to be (up to the way how we formalize the instances of the problem) a CSP for a fixed infinite template Γ. Relation algebras are designed to handle binary relations in an algebraic way; we follow the presentation in [30].

Definition 4. A *proper relation algebra* is a domain D together with a set $\mathcal{B}$ of binary relations over D such that

- $\mathrm{Id} := \{(x, x) \mid x \in D\} \in \mathcal{B}$;
- if B_1 and B_2 are from $\mathcal{B}$, then $B_1 \vee B_2 := B_1 \cup B_2 \in \mathcal{B}$;
- $1 := \bigcup_{R \in \mathcal{B}} R \in \mathcal{B}$;
- $0 := \emptyset \in \mathcal{B}$;

[3] In the conference version of this paper, these operations were called *weak near-unanimity operations*. However, since another similar but much weaker relaxation of near-unanimity operations was introduced recently in universal algebra as well, we decide to call our operations *quasi* near-unanimity operations.

- if $B \in \mathcal{B}$, then $-B := 1 \setminus B_1 \in \mathcal{B}$;
- if $B \in \mathcal{B}$, then $B^\smile := \{(x, y) \mid (y, x) \in B\} \in \mathcal{B}$;
- if B_1 and B_2 are from $\mathcal{B}$, then $B_1 \circ B_2 \in \mathcal{B}$; where

$$B_1 \circ B_2 := \big\{(x, z) \;\big|\; \exists y\big((x, y) \in B_1 \wedge (y, z) \in B_2\big)\big\}.$$

We want to point out that in this standard definition of proper relation algebras it is *not* required that 1 denotes D^2. However, in most examples that we encounter, 1 indeed denotes D^2. The minimal non-empty elements of $\mathcal{B}$ with respect to set-wise inclusion are called the *atoms* of the relation algebra, or also the *basic relations*.

Example 2 *(The point algebra).* Let $D = \mathbb{Q}$ be the set of rational numbers, and consider

$$\mathcal{B} = \big\{<, >, =, \leqslant, \geqslant, \emptyset, \mathbb{Q}^2\big\}.$$

Those relations form a proper relation algebra (with atoms $<$, $>$, $=$) which is one of the most fundamental relation algebras and known under the name *point algebra*.

When $\mathcal{B}$ is finite, every relation in $\mathcal{B}$ can be written as a finite union of basic relations, and we abuse notation and sometimes write $R = \{B_1, \ldots, B_k\}$ when $B_1, \ldots, B_k$ are basic relations, $R \in \mathcal{B}$, and $R = B_1 \cup \cdots \cup B_k$. Note that composition of basic relations determines the composition of all relations in the relation algebra, since

$$R_1 \circ R_2 = \bigcup_{B_1 \in R_1,\, B_2 \in R_2} B_1 \circ B_2.$$

8.2. Abstract relation algebras

An *abstract* relation algebra (Definition 5 below) is an algebra with signature $\mathrm{Id}, 0, 1, -, ^\smile, \vee, \circ$ that satisfies laws that we expect from those operators in a proper relation algebra.

Definition 5 *(following [24,30,37]).* An (abstract) relation algebra $\mathbf{A}$ is an algebra with domain A and signature $\{\vee, -, 0, 1, \circ, ^\smile, \mathrm{Id}\}$ such that

- the structure $(A; \vee, \wedge, -, 0, 1)$ is a Boolean algebra where $\wedge$ is defined by $(x, y) \mapsto -(-x \vee -y)$ from $-$ and $\vee$;
- $\circ$ is an associative binary operation on A;
- $(a^\smile)^\smile = a$ for all $a \in A$;
- $\mathrm{Id} \circ a = a \circ \mathrm{Id} = a$ for all $a \in A$;
- $a \circ (b \vee c) = a \circ b \vee a \circ c$;
- $(a \vee b)^\smile = a^\smile \vee b^\smile$;
- $(-a)^\smile = -(a^\smile)$;
- $(a \circ b)^\smile = b^\smile \circ a^\smile$;
- $(a \circ b) \wedge c^\smile = 0 \Leftrightarrow (b \circ c) \wedge a^\smile = 0$.

We define $x \leqslant y$ by $x \wedge y = x$. A *representation* (D, i) of $\mathbf{A}$ consists of a set D and a mapping i from the domain A of $\mathbf{A}$ to binary relations over D such that the image of i induces a proper relation algebra $\mathbf{A}'$, and i is an isomorphism with respect to the functions $\{\vee, -, 0, 1, \circ, ^\smile, \mathrm{Id}\}$. In this case, we also say that $\mathbf{A}$ is the *abstract relation algebra of* $\mathbf{A}'$.

There are finite abstract relation algebras that do not have a representation [40]. Note that when (D, i) is a representation of $\mathbf{A}$, then $i(a)$ is a basic relation of the induced proper relation algebra if and only if $a \neq 0$, and for every $b \leqslant a$ we have $b = a$ or $b = 0$; we call a an *atom* of $\mathbf{A}$. Using the axioms of relation algebras, it can be shown that the composition operator is uniquely determined by the composition operator on the atoms. Similarly, the inverse of an element $a \in A$ is the disjunction of the inverses of all the atoms below a.

Example 3. The (abstract) point algebra is a relation algebra with 8 elements and 3 atoms, denoted by $=$, $<$, and $>$. The composition operator of the basic relations of the point algebra is shown in the table of Fig. 1. By the observation we just made, this table determines the full composition table. The inverse of $<$ is $>$, and Id denotes $=$ which is its own inverse. This fully determines the relation algebra.

We can obtain a representation with domain $\mathbb{Q}$ from the point algebra (Example 2) in the obvious way. Note that this is not the only representation of the abstract point algebra: another representation can be obtained by taking $[0, 1]$ in place of $\mathbb{Q}$. While in any representation the relation for $<$ has to be transitive and dense, it need not be unbounded.

$\circ$	$=$	$<$	$>$
$=$	$=$	$<$	$>$
$<$	$<$	$<$	1
$>$	$>$	1	$>$

Fig. 1. The composition table for the basic relations in the point algebra.

8.3. The network satisfaction problem

The central computational problem that has been studied for relation algebras is the *network satisfaction problem* [24,30,37].

Definition 6. Let **A** be a finite relation algebra with domain A. An (**A**-)*network* $N = (V, f)$ consists of a finite set of nodes V and a function $f : V \times V \to A$.

Two types of network satisfaction problems have been studied for **A**-networks. The first is the *network satisfaction problem for a (fixed) representation* (D, i) *of* **A**: here, the input is an **A**-network N, and the question is whether N is *satisfiable with respect to* (D, i), that is, whether there exists a mapping $s : V \to D$ such that for all $u, v \in V$

$$\big(s(u), s(v)\big) \in i\big(f(u, v)\big).$$

Another problem that has been studied is the (*general*) *network satisfaction problem for* **A**. Again, the input is an **A**-network N. This time the question is whether there *exists* a representation (D, i) of **A** such that N is satisfiable with respect to (D, i). It is not hard to show that for every finite relation algebra **A** that has a representation, there is also a representation (D, i) such that the network satisfaction problem for (D, i) is the same problem as the general network satisfaction problem for **A**. So we focus on the network satisfaction problem for fixed representations here.

We now present the link between network satisfaction problems and constraint satisfaction problems as defined earlier in this paper. Let $\tau_{\mathbf{A}}$ be a signature consisting of binary relation symbols: $\tau_{\mathbf{A}}$ contains a binary relation symbol R_a for each element $a \in A$. When (D, i) is a representation of $\tau_{\mathbf{A}}$, then we associate to it a $\tau_{\mathbf{A}}$-structure $\Gamma_{D,i}$ in a natural way: the domain of the structure is D, and the relation symbol R_a is interpreted by $i(a)$. We sometimes also call the $\tau_{\mathbf{A}}$-structure $\Gamma_{D,i}$ a *representation* of **A**.

Also to each **A**-network $N = (V, f)$ we can associate a $\tau_{\mathbf{A}}$-structure S_N in a straightforward way: the domain of S_N is V, and for $u, v \in V$ we have $(u, v) \in R_a$ if and only if $f(u, v) = a$. Conversely, we can associate to each finite $\tau_{\mathbf{A}}$-structure S a network $N_S = (V, f)$ as follows. The node set V of N is D_S, the domain of S. Let $u, v \in V$, and list by $a_1, \ldots, a_k$ all those elements a of A such that $(u, v) \in R_a$. Then define $f(u, v) = a$ for $a = (a_1 \wedge a_2 \wedge \cdots \wedge a_k)$ (if $k = 0$, then $a = 0$ by definition).

The following link between the network satisfaction problem for a fixed representation (D, i) of **A**, and the constraint satisfaction problem for $\Gamma_{D,i}$ is straightforward from the definitions.

Proposition 5. *Let* **A** *be a finite relation algebra with representation* (D, i)*. Then an* **A***-network* N *is satisfiable with respect to* (D, i) *if and only if* S_N *homomorphically maps to* $\Gamma_{D,i}$*. Moreover, a finite* $\tau_{\mathbf{A}}$*-structure* S *homomorphically maps to* $\Gamma_{D,i}$ *if and only if* N_S *is satisfiable with respect to* (D, i)*.*

8.4. Datalog and path-consistency

One of the main algorithmic techniques used in the context of network satisfaction problems is the *path consistency procedure*. We will see that – under the translation of terminology presented in Section 8.3 – the path consistency procedure can be formulated with a Datalog program.

The path-consistency procedure for **A** takes as input an **A**-network N. The execution of the procedure on N only depends on **A** as an *abstract* relation algebra (and not on particular representations of **A**). (See Fig. 2.)

Proposition 6. *Let* **A** *be a finite relation algebra. Then there exists a Datalog program* Π *such that for every* **A***-network* N*, the program* Π *derives false on* S_N *if and only if the path-consistency procedure for* **A** *rejects* N*.*

Proof. The Datalog program Π is defined as follows. The signature $\tau_{\mathbf{A}}$ defined above is the set of EDBs; as IDBs, we have a binary relation S_a for each $a \in A$, and the distinguished 0-ary predicate *false*. Then Π contains for each $a \in A$ the rule

$$S_a(x) :- R_a(x),$$

and for all $a, b \in A$ the rules

$\mathrm{PC}_{\mathbf{A}}(N)$
Input: an **A**-network $N = (V, f)$.
Do
  For all distinct nodes $x, y, z \in V$:
    Replace $f(x, y)$ by $f(x, y) \wedge (f(x, z) \circ f(z, y))$
    If $f(x, y) = 0$ then **reject**
Loop until no further changes
Return (V, f).

Fig. 2. The path-consistency procedure for **A**-networks.

$$S_{a\circ b}(x, y) :- S_a(x, z), S_b(z, y),$$
$$S_{a\wedge b}(x, y) :- S_a(x, y), S_b(x, y).$$

The verification that Π has the required properties is straightforward and left to the reader. □

8.5. Discussion

We close this section by discussing the weaknesses of the relation algebra approach to constraint satisfaction. First of all, the class of problems that can be formulated as a network satisfaction problems is *severely* restricted. The relations that we allow in the input network are closed under unions; this introduces a sort of restricted disjunction that quickly leads to NP-hardness, and indeed the network satisfiability problem is tractable in only a few exceptional cases [30]. The typical work-around here is to introduce another parameter, namely a subset of B of the domain of **A**, and to study the network satisfaction problem for networks $N = (V, f)$ where the image of f is contained in B. Note that such an additional parameter is not necessary for CSPs as treated in this paper. Also note that the network satisfaction problem is restricted to *binary* relations, whereas many important CSPs can only be formulated in a natural way with higher-ary relations. As we have seen in Proposition 6, every network satisfaction problem for a fixed representation can be formulated as $\mathrm{CSP}(\Gamma)$ for an appropriate infinite structure Γ; but as the above remarks show, only a very small fraction of CSPs can be formulated as a network satisfaction problem.

Even though only very specific CSPs can be formulated as the network satisfaction problem for a finite relation algebra **A**, there are hardly any additional techniques available for studying the complexity of network satisfaction problems, since the tools we have for network satisfaction usually also apply to constraint satisfaction. For instance, the main computational technique that has been studied for the network satisfaction problem is local consistency (such as path consistency); however, this technique is also applicable to infinite-domain CSPs in general. As we have seen in this paper, local consistency is particularly powerful for problems of the form $\mathrm{CSP}(\Gamma)$ where Γ is ω-categorical. When the network satisfiability problem under consideration cannot be formulated as $\mathrm{CSP}(\Gamma)$ for an ω-categorical structure Γ, then not much is known about the power of consistency techniques for the network satisfiability problem, either.

The study of composition of relations in the context of the network satisfiability problem is usually justified by the fact that a network with constraints over the relation $R \circ S$ can be simulated by networks that only have constraints over the relation R and over the relation S. But the same holds for primitive positive definable relations. Apart from being more powerful, primitive positive definability has another advantage in comparison to relational composition in relation algebras: while the set of relations that can be obtained by composing and intersecting the binary relations from a subset of a relation algebra is intricate and not well understood, there is a powerful Galois-theory to study primitive positive definability of relations [10]. In fact, for many infinite structure Γ the question whether a given first-order formula has a primitive positive definition over Γ is decidable [12].

References

[1] D. Achlioptas, The complexity of G-free colourability, Discrete Math. 165 (1997) 21–30.
[2] A. Atserias, On digraph coloring problems and treewidth duality, in: Proceedings of LICS, 2005, pp. 106–115.
[3] A. Atserias, A. Bulatov, V. Dalmau, P.G. Kolaitis, M.Y. Vardi, Consistency, treewidth, and number of first-order variables, in preparation.
[4] L. Barto, M. Kozik, Constraint satisfaction problems of bounded width, in: Proceedings of FOCS, 2009, pp. 595–603.
[5] M. Bodirsky, Cores of countably categorical structures, Log. Methods Comput. Sci. 3 (1) (2007) 1–16.
[6] M. Bodirsky, H. Chen, Oligomorphic clones, Algebra Universalis 57 (1) (2007) 109–125.
[7] M. Bodirsky, H. Chen, Qualitative temporal and spatial reasoning revisited, in: Proceedings of CSL, 2007, pp. 194–207.
[8] M. Bodirsky, V. Dalmau, Datalog and constraint satisfaction with infinite templates, in: Proceedings of STACS, 2006, pp. 646–659.
[9] M. Bodirsky, J. Kára, The complexity of temporal constraint satisfaction problems, J. ACM 57 (2) (2009), 41 pp., an extended abstract appeared in the proceedings of STOC'08.
[10] M. Bodirsky, J. Nešetřil, Constraint satisfaction with countable homogeneous templates, J. Logic Comput. 16 (3) (2006) 359–373.
[11] M. Bodirsky, M. Pinsker, Schaefer's theorem for graphs, in: Proceedings of STOC, 2011, pp. 655–664; preprint of the long version available at arXiv: 1011.2894v2 [cs.CC].
[12] M. Bodirsky, M. Pinsker, T. Tsankov, Decidability of definability, in: Proceedings of LICS, 2011, pp. 321–328.
[13] A. Bulatov, Bounded relational width, manuscript, 2009.
[14] A. Bulatov, P. Jeavons, A. Krokhin, The complexity of constraint satisfaction: An algebraic approach (a survey paper), in: Structural Theory of Automata, Semigroups and Universal Algebra, Montreal, 2003, in: NATO Sci. Ser. II Math. Phys. Chem., vol. 207, 2005, pp. 181–213.

[15] A. Bulatov, A. Krokhin, P.G. Jeavons, Classifying the complexity of constraints using finite algebras, SIAM J. Comput. 34 (2005) 720–742.
[16] P.J. Cameron, Oligomorphic Permutation Groups, Cambridge University Press, Cambridge, 1990.
[17] A.K. Chandra, P.M. Merlin, Optimal implementation of conjunctive queries in relational data bases, in: Proceedings of STOC, 1977, pp. 77–90.
[18] G. Cherlin, S. Shelah, N. Shi, Universal graphs with forbidden subgraphs and algebraic closure, Adv. in Appl. Math. 22 (1999) 454–491.
[19] J. Covington, Homogenizable relational structures, Illinois J. Math. 34 (4) (1990) 731–743.
[20] N. Creignou, P.G. Kolaitis, H. Vollmer (Eds.), Complexity of Constraints – An Overview of Current Research Themes [Result of a Dagstuhl Seminar], Lecture Notes in Comput. Sci., vol. 5250, Springer, 2008.
[21] M. Cristiani, R. Hirsch, The complexity of the constraint satisfaction problem for small relation algebras, Artificial Intelligence J. 156 (2004) 177–196.
[22] V. Dalmau, P.G. Kolaitis, M.Y. Vardi, Constraint satisfaction, bounded treewidth, and finite-variable logics, in: Proceedings of CP, 2002, pp. 310–326.
[23] V. Dalmau, J. Pearson, Closure functions and width 1 problems, in: Proceedings of CP, 1999, pp. 159–173.
[24] I. Duentsch, Relation algebras and their application in temporal and spatial reasoning, Artificial Intelligence Rev. 23 (2005) 315–357.
[25] H.-D. Ebbinghaus, J. Flum, Finite Model Theory, 2nd edition, Springer, Berlin, Heidelberg, New York, 1999.
[26] T. Feder, M.Y. Vardi, The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory, SIAM J. Comput. 28 (1999) 57–104.
[27] M. Grohe, The structure of fixed-point logics, PhD thesis, Albert-Ludwigs Universität, Freiburg i. Br., 1994.
[28] M. Grohe, The complexity of homomorphism and constraint satisfaction problems seen from the other side, J. ACM 54 (1) (2007).
[29] P. Hell, J. Nešetřil, Graphs and Homomorphisms, Oxford University Press, Oxford, 2004.
[30] R. Hirsch, Expressive power and complexity in algebraic logic, J. Logic Comput. 7 (3) (1997) 309–351.
[31] W. Hodges, Model Theory, Cambridge University Press, 1993.
[32] P. Jeavons, D. Cohen, M. Cooper, Constraints, consistency and closure, Artificial Intelligence 101 (1–2) (1998) 251–265.
[33] P. Jeavons, D. Cohen, M. Gyssens, Closure properties of constraints, J. ACM 44 (4) (1997) 527–548.
[34] P.G. Kolaitis, M.Y. Vardi, On the expressive power of Datalog: Tools and a case study, J. Comput. System Sci. 51 (1) (1995) 110–134.
[35] P.G. Kolaitis, M.Y. Vardi, Conjunctive-query containment and constraint satisfaction, in: Proceedings of PODS, 1998, pp. 205–213.
[36] G. Kun, Constraints, MMSNP, and expander relational structures, available at arXiv:0706.1701, 2007.
[37] P.B. Ladkin, R.D. Maddux, On binary constraint problems, J. ACM 41 (3) (1994) 435–469.
[38] B. Larose, C. Loten, C. Tardif, A characterisation of first-order constraint satisfaction problems, Log. Methods Comput. Sci. 4 (2007) 6.
[39] B. Larose, C. Tardif, Strongly rigid graphs and projectivity, Multiple-Valued Logic 7 (2001) 339–361.
[40] R. Lyndon, The representation of relational algebras, Ann. of Math. 51 (3) (1950) 707–729.
[41] F. Madelaine, I.A. Stewart, Constraint satisfaction, logic and forbidden patterns, SIAM J. Comput. 37 (1) (2007) 132–163.
[42] F.R. Madelaine, I.A. Stewart, Some problems not definable using structure homomorphisms, Ars Combin. 67 (2003) 153–159.
[43] J. Nešetřil, C. Tardif, Duality theorems for finite structures, J. Combin. Theory Ser. B 80 (2000) 80–97.
[44] B. Rossman, Homomorphism preservation theorems, J. ACM 55 (3) (2008).
[45] J. van Leeuwen, Graph algorithms, in: Handbook of Theoretical Computer Science, vol. A: Algorithms and Complexity (A), Elsevier, MIT Press, 1990, pp. 525–631.