Symmetric Linear Arc Monadic Datalog and Gadget Reductions

Manuel Bodirsky
Institut für Algebra, TU Dresden, Germany

Florian Starke
Institut für Algebra, TU Dresden, Germany

Abstract

A Datalog program *solves* a constraint satisfaction problem (CSP) if and only if it derives the goal predicate precisely on the unsatisfiable instances of the CSP. There are three Datalog fragments that are particularly important for finite-domain constraint satisfaction: *arc monadic Datalog*, *linear Datalog*, and *symmetric linear Datalog*, each having good computational properties. We consider the fragment of Datalog where we impose all of these restrictions simultaneously, i.e., we study *symmetric linear arc monadic (slam) Datalog*. We characterise the CSPs that can be solved by a slam Datalog program as those that have a gadget reduction to a particular Boolean constraint satisfaction problem. We also present exact characterisations in terms of a homomorphism duality (which we call *unfolded caterpillar duality*), and in universal-algebraic terms (using known minor conditions, namely the existence of *quasi Maltsev operations* and *k-absorptive operations of arity nk*, for all $n, k \geq 1$). Our characterisations also imply that the question whether a given finite-domain CSP can be expressed by a slam Datalog program is decidable.

2012 ACM Subject Classification Theory of computation → Finite Model Theory

Keywords and phrases Datalog, Gadget Reductions, Homomorphism Dualities, Minor Conditions

Digital Object Identifier 10.4230/LIPIcs.ICDT.2025.13

Related Version *Full Version*: `https://arxiv.org/abs/2407.04924` [10]

Funding Both authors have been funded by the European Research Council (Project POCOCOP, ERC Synergy Grant 101071674). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

1 Introduction

Datalog is an important concept linking database theory with the theory of constraint satisfaction. It is by far the most intensively studied formalism for polynomial-time tractability in constraint satisfaction. Datalog allows to formulate algorithms that are based on iterating local inferences, aka *constraint propagation* or *establishing local consistency*; this has been made explicit by Feder and Vardi in their groundbreaking work where they also formulate the finite-domain CSP dichotomy conjecture [25]. Following their convention, we say that a Datalog program Π *solves* a CSP if Π derives the goal predicate on an instance of the CSP if and only if the instance is unsatisfiable.

The class of CSPs that can be solved by Datalog is closed under so-called "gadget reductions" (a result due to Larose and Zádori [33]). In such a reduction, the variables in an instance of a constraint satisfaction problem are replaced by tuples of variables of some fixed finite length, and the constraints are replaced by *gadgets* (implemented by conjunctive

queries; a formal definition can be found in Section 2.3); many of the well-known reductions between computational problems can be phrased as gadget reductions. Datalog is sufficiently powerful to simulate such gadget reductions; this has been formalised by Atserias, Bulatov, and Dawar in [2] and the connection has been sharpened recently by Dalmau and Opršal [21].

Feder and Vardi showed that Datalog cannot solve systems of linear equations over finite fields, even though such systems can be solved in polynomial time [25]. They suggest that the ability to simulate systems of linear equations should essentially be the only reason for a CSP to not be in Datalog. This conjecture was formalised by Larose and Zádori [33]: they observed that if systems of linear equations admit a gadget reduction to a CSP, then the CSP is not in Datalog, and they conjectured that otherwise the CSP can be solved by Datalog. This conjecture was proved by Barto and Kozik in 2009 [4], long before the resolution of the finite-domain CSP dichotomy conjecture by Bulatov [12] and by Zhuk [41, 40].

Datalog programs can be evaluated in polynomial time; but even a running time in $O(n^3)$ on a sequential computer can be prohibitively expensive in practise. This is one of the reasons why syntactic *fragments* of Datalog have been studied, which often come with better computational properties.

1.1 Arc Monadic Datalog

In *monadic Datalog*, we restrict the arity of the inferred predicates of the Datalog program to one (i.e., all the *IDBs* are monadic). In *arc Datalog* we restrict each rule to a single input relation symbol (i.e., the body contains a single *EDB*; for formal definitions, see Section 2.4).

An important Datalog fragment is *arc monadic Datalog*, which is still powerful enough to express the famous *arc consistency procedure* in constraint satisfaction. The arc consistency procedure has already been studied by Feder and Vardi [25], and has many favorable properties: it can be evaluated in linear time and linear space. It is used as an important pre-processing step in the algorithms for both of the mentioned CSP dichotomy proofs, and it is also used in many practical implementations of algorithms in constraint satisfaction. The arc consistency procedure is still extremely powerful, and can for instance solve the P-complete HornSat Problem.

Feder and Vardi characterised the power of the arc consistency procedure in terms of *tree duality* (see Section 2.6), a natural combinatorial property which has been studied intensively in the graph homomorphism literature in the 90s (see, e.g., [26, 27]). Their characterisation has several remarkable consequences: one is that also the class of CSPs that can be solved by an arc monadic Datalog program is closed under gadget reductions. Another one is a *collapse result* for Datalog when it comes to finite-domain CSPs, namely that monadic Datalog collapses to arc monadic Datalog: in fact, if a finite-domain CSP can be solved by a monadic Datalog program, then it can already be solved by the arc consistency procedure (i.e., by a program in arc monadic Datalog). This statement is false without the restriction to finite-domain CSPs; in fact, there are infinite-domain CSPs that can be solved by a monadic Datalog program, but not by a program in arc monadic Datalog (Bodirsky and Dalmau [8]).

1.2 Linear Datalog

Besides arc monadic Datalog, there are other natural fragments of Datalog. The most notable one is *linear Datalog* [29]. Linear Datalog programs can be evaluated in non-deterministic logarithmic space (NL), and hence cannot express P-hard problems (unless

P=NL). Dalmau [19] asked whether the converse is true as well, i.e., whether every finite-domain CSP which is in NL can be solved by a linear Datalog program [19]. This is widely treated as a conjecture, to which we refer as the *linear Datalog conjecture*; it is one of the biggest open problems in finite-domain constraint satisfaction.

There are some sufficient conditions for solvability by linear Datalog (see Bulatov, Kozik, and Willard [3] and Carvalho, Dalmau, and Krokhin [15]) and some necessary conditions (Larose and Tesson [32]) but the results still leave a large gap. For examples of CSPs of orientations of trees that fall into this gap, see Bodirsky, Bulín, Starke, and Wernthaler [7]. Again, linear Datalog is closed under gadget reductions [38]. And indeed, if HornSat has a gadget-reduction in a finite-domain CSP, then the finite-domain CSP cannot be solved by a linear Datalog program [1].

1.3 Symmetric Linear Datalog

A further restriction is *symmetric linear Datalog*, introduced by Egri, Larose, and Tesson [23]. Symmetric linear Datalog programs can be evaluated in deterministic logspace (L). Egri, Larose, and Tesson conjecture that every finite-domain CSP which is in L can be be solved by a symmetric linear Datalog program [23]; we refer to this conjecture as the *symmetric linear Datalog conjecture*.

Symmetric linear Datalog is closed under gadget reductions [38]. Since directed reachability is not in symmetric linear Datalog [24], it follows that every CSP that admits a gadget reduction from directed reachability cannot be solved by a symmetric linear Datalog program. Egri, Larose, and Tesson also suggest that this might be the only additional condition for containment in symmetric linear Datalog, besides the known necessary conditions to be in linear Datalog. Kazda [30] confirms the symmetric linear Datalog conjecture conditionally on the truth of the linear Datalog conjecture, i.e., he shows that if a finite-domain CSP is in linear Datalog and does not admit a gadget reduction from a CSP that corresponds to the directed reachability problem, then it is in symmetric linear Datalog (generalizing an earlier result of Dalmau and Larose [20]).

1.4 Our Contributions

In this paper, we study the Datalog fragment that can be obtained by combining all the previously considered restrictions, namely *symmetric linear arc monadic (slam) Datalog*. Before stating our result we illustrate this fragment with some examples. For $n \geq 1$, let $\mathfrak{P}_n$ be the directed path with n vertices and $n-1$ edges. An example of a slam Datalog program which solves $\mathrm{CSP}(\mathfrak{P}_2)$ is

$$A(x) :- E(x,y) \qquad \text{goal} :- E(x,y), A(y)$$

(in this case, the program is even recursion-free). An example of a slam Datalog program which solves $\mathrm{CSP}(\mathfrak{P}_3)$, this time with recursion and IDBs A and B, is

$$\begin{aligned} A(x) &:- E(x,y) & B(x) &:- A(y), E(x,y) \\ A(y) &:- B(x), E(x,y) & \text{goal} &:- B(y), E(x,y). \end{aligned}$$

The idea why this program is correct is that a finite digraph $\mathfrak{A}$ has a homomorphism to $\mathfrak{P}_3$ if and only if certain orientations of paths (those of *net length three*; for a formal description, see Example 8) do not have a homomorphism to $\mathfrak{A}$; and the program derives the goal predicate on $\mathfrak{A}$ precisely if there is a homomorphism from such a path to $\mathfrak{A}$.

It follows from our results that the class of CSPs that can be solved by slam Datalog programs is closed under gadget reductions, despite the many restrictions that we imposed.

We provide a *full description* of the power of a Datalog fragment in terms of gadget reductions: we show that a CSP can be solved by a slam Datalog program if and only if it has a gadget reduction to $\mathrm{CSP}(\mathfrak{P}_2)$.[1] The particular role of the structure $\mathfrak{P}_2$ is explained by the fact that it is a representative of the unique class of CSPs which is non-trivial and *weakest* with respect to gadget reductions – a formalisation of this can be found in Section 2.3.[2] This shows that slam Datalog is the smallest non-trivial fragment of Datalog that is closed under gadget reductions.

Our main result (Theorem 18) establishes a tight connection between the power of slam Datalog and various central themes in structural combinatorics and universal algebra. Specifically, the power of slam Datalog can be characterised by a combinatorial duality which we call *unfolded caterpillar duality* (restricting the concept of *caterpillar duality* of Carvalho, Dalmau, and Krokhin [16], and using ideas that appear implicitly in the literature on symmetric Datalog [20, 23, 30]), and by the existence of a *quasi Maltsev polymorphism* (a central concept in universal algebra) in combination with *kn-ary k-absorbing polymorphisms for every $k, n \geq 1$* (introduced in [16] as well). Our result also implies that the following *meta-problem* can be decided algorithmically: given a finite structure $\mathfrak{B}$, can $\mathrm{CSP}(\mathfrak{B})$ be solved by a slam Datalog program?

1.5 Related Results

Solvability of finite-domain CSPs by (unrestricted) Datalog was first studied by Feder and Vardi; they proved that $\mathrm{CSP}(\mathfrak{B})$ can be solved by Datalog if and only if $\mathfrak{B}$ has *bounded treewidth duality*, and they showed that CSPs for systems of linear equations over finite Abelian groups cannot be solved by Datalog. Larose and Zadori [33] showed that solvability by Datalog is preserved by gadget reductions and they asked whether having a gadget reduction from CSPs for systems of linear equations is not only a sufficient, but also a necessary condition for not being solvable by Datalog. This questions was answered positive by Barto and Kozik [4]. Kozik, Krokhin, and Willard [31] gave a characterisation of Datalog in terms of minor conditions.

Linear (but not necessarily symmetric) monadic arc Datalog has been studied by Carvalho, Dalmau, and Krokhin [16]; our proof builds on their result. In their survey on Datalog fragments and dualities in constraint satisfaction [13] Bulatov, Krokhin, and Larose write *"it would be interesting to find (...) an appropriate notion of duality for symmetric (Linear) Datalog (...)"*. We do find such a notion for symmetric linear arc monadic Datalog, namely unfolded caterpillar duality (Theorem 18).

Another fragment of Datalog consists of the set of conjunctive queries; the CSPs that can be solved by such Datalog programs are precisely the CSPs that are first-order expressible, by Rossman's theorem [36]. This is also known to be equivalent to CSPs having *finite duality*. However, note that first-order definability is not preserved under gadget reductions (as we will see in Section 2.6).

---

[1] The statement even holds for infinite-domain CSPs, since being solved by an arc monadic Datalog program implies the existence of a finite template [8] and admitting a gadget reduction to a finite-domain CSP implies the existence of a finite template as well [21].

[2] We mention that $\mathfrak{P}_2$ is a Boolean structure which simultaneously satisfies the Schaefer conditions of being Horn, dual Horn, affine, and bijunctive [37].

2 Preliminaries

We write $[n]$ for the set $\{1, \dots, n\}$ and $[m, n]$ for the set $\{m, m+1, \dots, n\}$. We say that a tuple $a \in A^k$, for $k \in \mathbb{N}$, is *injective* if a is injective when viewed as a function from $[k]$ to A.

2.1 Structures and Graphs

We assume familiarity with the concepts of relational structures and first-order formulas from mathematical logic, as introduced for instance in [28]. The arity of a relation symbol R is denoted by $\mathrm{ar}(R)$. If $\mathfrak{A}$ is a τ-structure, then we sometimes use the same symbol for $R \in \tau$ and the respective relation $R^{\mathfrak{A}}$ of $\mathfrak{A}$. We write $\mathfrak{A}[S]$ for the substructure of $\mathfrak{A}$ induced on S.

A *(directed) graph* is a relational structure with a single binary relation E. For instance the *clique with n vertices* is the graph $\mathfrak{K}_n$ with domain $[n]$ and edges $E^{\mathfrak{K}_n} := \{(a, b) \mid a \neq b\}$. Let $\mathfrak{G}$ be a graph. An *(undirected) path from a to b in $\mathfrak{G}$* is a tuple $P = (a_1, \dots, a_n)$ such that $a_1, \dots, a_n$ are pairwise distinct, $a_1 = a$, $a_n = b$, and for all $i \in [n-1]$ there is an edge between a_i and a_{i+1} (from a_i to a_{i+1} or from a_{i+1} to a_i). If $i \in [2, n-1]$, then we say that P *passes through* a_i. A graph $\mathfrak{G}$ is called *connected* if for any two elements a, b there exists a path from a to b in $\mathfrak{G}$. A *cycle* is a path from a to b of length n at least three such that there is an edge between a and b. A graph is called *acyclic* if it does not contain any cycle. A graph is called a *tree* if it is connected and acyclic. A graph has *girth* k if the length of the shortest cycle is k.

2.2 Homomorphisms and CSPs

Let τ be a relational signature and let $\mathfrak{A}$ and $\mathfrak{B}$ be τ-structures. Then a *homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$* is a map $h\colon A \to B$ such that for $R \in \tau$, say of arity k, we have $(h(a_1), \dots, h(a_k)) \in R^{\mathfrak{B}}$ whenever $(a_1, \dots, a_k) \in R^{\mathfrak{A}}$. An embedding of $\mathfrak{A}$ into $\mathfrak{B}$ is an injective map $e\colon A \to B$ such that $(e(a_1), \dots, e(a_k)) \in R^{\mathfrak{B}}$ if and only if $(a_1, \dots, a_k) \in R^{\mathfrak{A}}$. We write $\mathfrak{A} \to \mathfrak{B}$ if there exists a homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$ and $\mathfrak{A} \not\to \mathfrak{B}$ if there exists no homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$.

If τ is a finite relational signature and $\mathfrak{B}$ is a τ-structure, then $\mathrm{CSP}(\mathfrak{B})$ denotes the class of all finite τ-structures $\mathfrak{A}$ such that $\mathfrak{A} \to \mathfrak{B}$. It can be viewed as a computational problem. For example, $\mathrm{CSP}(\mathfrak{K}_n)$ consists of the set of all finite n-colourable graphs, and can therefore be viewed as the n-colorability problem. Clearly, for finite structures $\mathfrak{B}$, this problem is always in NP. Note that from the database perspective, by the work of Chandra and Merlin [17], $\mathrm{CSP}(\mathfrak{B})$ can be viewed as the expression complexity of conjunctive queries over $\mathfrak{B}$.

A τ-structure $\mathfrak{B}$ is *homomorphically equivalent* to a τ-structure $\mathfrak{C}$ if there are homomorphisms from $\mathfrak{B}$ to $\mathfrak{C}$ and vice versa. Clearly, homomorphically equivalent structures have the same CSP. A relational τ-structure $\mathfrak{C}$ is called a *core* if all endomorphisms of $\mathfrak{C}$ are embeddings. It is well-known and easy to see that every finite structure $\mathfrak{B}$ is homomorphically equivalent to a core $\mathfrak{C}$, and that all core structures $\mathfrak{C}$ that are homomorphically equivalent to $\mathfrak{B}$ are isomorphic; therefore, we refer to $\mathfrak{C}$ as *the* core of $\mathfrak{B}$.

2.3 Primitive Positive Constructions

A τ-formula ϕ is called a *conjunctive query* (in constraint satisfaction and model theory such formulas are called *primitive positive*, or short *pp*) if it is built from atomic formulas (including atomic formulas of the form $x = y$) using only conjunction and existential quantification. If $\mathfrak{B}$ is a τ-structure, and ϕ is a conjunctive query over the signature τ, then the relation $R = \{(t_1, \dots, t_k) \mid \mathfrak{B} \models \phi(t_1, \dots, t_k)\}$ is called the *relation defined by ϕ*.

▶ **Definition 1.** *The* canonical database *of a conjunctive query ϕ over the signature τ is the τ-structure $\mathfrak{B}$ that can be constructed as follows: Let ϕ' be obtained from ϕ be renaming all existentially quantified variables such that no two quantified variables have the same name. Let ϕ'' be obtained from ϕ' by removing all conjuncts of the form $x = y$ in ϕ' and by identifying variables x and y if there is a conjunct $x = y$ in ϕ'. Then $\mathfrak{B}$ is the τ-structure whose domain is the set of variables of ϕ'' such that for every $R \in \tau$ we have*

$$R^{\mathfrak{B}} = \{(v_1, \dots, v_k) \mid R(v_1, \dots, v_k) \text{ is a conjunct of } \phi''\}.$$

The canonical conjunctive query *of a structure $\mathfrak{B}$ with signature τ is the primitve positive τ-formula with variables B that contains for every $R \in \tau$ and every $t \in R^{\mathfrak{B}}$ the conjunct $R(t_1, \dots, t_{\mathrm{ar}(R)})$.*

Observe that the canonical database of the canonical conjunctive query of a structure $\mathfrak{B}$ equals $\mathfrak{B}$. The following concepts have been introduced by Barto, Opršal, and Pinsker [5].

▶ **Definition 2.** *A* (d-th) pp-power *of a τ-structure $\mathfrak{B}$ is a structure $\mathfrak{C}$ with domain B^d such that every relation of $\mathfrak{C}$ of arity k is definable by a conjunctive query in $\mathfrak{B}$ as a relation of arity dk. A structure has a* primitive positive (pp) construction from $\mathfrak{B}$ *if it is homomorphically equivalent to a pp-power of $\mathfrak{B}$.*

Primitive positive constructions turned out to the *the* essential tool for classifying the complexity of finite-domain CSPs, because if $\mathfrak{C}$ has a pp-construction from $\mathfrak{B}$, then there is a so-called *gadget reduction* from CSP($\mathfrak{C}$) to CSP($\mathfrak{B}$); in fact, the converse is true as well, see Dalmau and Opršal [21].

▶ **Definition 3.** *Let $\mathcal{B}$ be a class of finite τ-structures and let $\mathcal{C}$ be a class of finite ρ-structures. Then a* (d-dimensional) gadget reduction *from $\mathcal{C}$ to $\mathcal{B}$ consists of a conjunctive query ϕ_R of arity dk over the signature τ for every $R \in \rho$ of arity k. This defines the following map r from finite ρ-structures $\mathfrak{C}$ to finite τ-structures:*

- *replace each element c of $\mathfrak{C}$ by the d-tuple $((c,1), \dots, (c,d))$;*
- *for every $R \in \rho$ of arity k and every tuple $(t_1, \dots, t_k) \in R^{\mathfrak{C}}$, introduce a new element for every existentially quantified variable in ϕ_R and define relations for the relation symbols from τ such that the substructure induced by the new elements and $\{(t_1,1), \dots, (t_1,d), \dots, (t_k,1), \dots, (t_k,d)\}$ induce a copy of the canonical database of ϕ'_R in the natural way, where ϕ'_R is obtained from ϕ_R by removing all conjuncts of the form $x = y$. Let $\mathfrak{C}'$ be the resulting structure.*
- *let S be the smallest equivalence relation that contains all ordered pairs of elements $((c,i),(d,j))$ such that there exists $R \in \rho$ and $(t_1, \dots, t_k) \in R^{\mathfrak{C}}$ with $t_p = c$, $t_q = d$ such that $\phi_R(x_{1,1}, \dots, x_{k,d})$ contains the conjunct $x_{p,i} = x_{q,j}$. Then $r(\mathfrak{C}) := \mathfrak{C}'/S$.*

For instance, the solution to the Feder-Vardi conjecture mentioned in the introduction states that CSP($\mathfrak{B}$), for a finite structure $\mathfrak{B}$, is NP-hard if and only if $\mathfrak{K}_3$ has a pp-construction from $\mathfrak{B}$ (unless P = NP); by what we have stated above, this is true if and only if the 3-coloring problem has a gadget reduction to CSP($\mathfrak{B}$).

▶ **Example 4.** Let $\tau = \{E\}$ be the signature that consists of a single binary relation symbol E whose elements we call *edges*. Let $\mathfrak{P}_n$ be the τ-structure with the domain $\{1, 2, \dots, n\}$ and edges $\{(1,2),(2,3),\dots,(n-1,n)\}$. Then $\mathfrak{P}_2$ pp-constructs $\mathfrak{P}_3$ [9], so CSP($\mathfrak{P}_3$) has a gadget reduction to CSP($\mathfrak{P}_2$).

▶ Remark 5. It is known that pp-constructibility is transitive [5], and the corresponding poset has a largest element (which is represented by $\mathfrak{P}_1$), and all other elements are below the element which is represented by $\mathfrak{P}_2$; see, e.g., [11]).

Given the fundamental importance of conjunctive queries and homomorphisms in database theory, we believe that pp-constructions and gadget reductions are an interesting concept for database theory as well.

2.4 Datalog

Let τ and ρ be finite relational signatures such that $\tau \subseteq \rho$. A *Datalog program* is a finite set of rules of the form $\phi_0 :- \phi_1, \ldots, \phi_n$ where each ϕ_i is an atomic τ-formula. The formula ϕ_0 is called the *head* of the rule, and the sequence $\phi_1, \ldots, \phi_n$ is called the *body* of the rule. The symbols in τ are called *EDBs* (*extensional database predicates*) and the other symbols from ρ are called *IDBs* (*intensional database predicates*). In the rule heads, only IDBs are allowed. There is one special IDB of arity 0, which is called the *goal predicate*. IDBs might also appear in the rule bodies. We view the set of rules as a recursive specification of the IDBs in terms of the EDBs – for a detailed introduction, see, e.g., [8]. A Datalog program is called

- *linear* if in each rule, at most one IDB appears in the body (we then assume without loss of generality that in every rule whose body contains an IDB, the IDB is listed first).
- *arc* if each rule involves at most one EDB.
- *symmetric* if it is linear and for every rule $\phi_0 :- \phi_1, \phi_2, \ldots, \phi_n$ where ϕ_0 and ϕ_1 are build from IDBs, the Datalog program also contains the *reversed rule* $\phi_1 :- \phi_0, \phi_2, \ldots, \phi_n$.[3]

If $\mathfrak{B}$ is a τ-structure, then we say that CSP($\mathfrak{B}$) is *solved* by a Datalog program Π with EDBs τ if the following holds: the goal predicate is derived by Π on a finite τ-structure $\mathfrak{A}$ if and only if there is *no* homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$. We say that a Datalog program has *width* (ℓ, k) if all IDBs have arity at most ℓ, and if every rule has at most k variables. For given (ℓ, k) and a structure $\mathfrak{B}$, there exists a Datalog program Π of width (ℓ, k) with the remarkable property that if some Datalog program of width (ℓ, k)) solves CSP($\mathfrak{B}$), then Π solves CSP($\mathfrak{B}$). This Datalog program is referred to as the *canonical Datalog program for* $\mathfrak{B}$ *of width* (ℓ, k), and is constructed as follows [25]: For every relation R over B of arity at most ℓ, we introduce a new IDB. The empty relation of arity 0 is the goal predicate. Then Π contains all rules $\phi :- \phi_1, \ldots, \phi_n$ with at most k variables such that the formula $\forall \bar{x}(\phi_1 \wedge \cdots \wedge \phi_n \Rightarrow \phi)$ holds in the expansion of $\mathfrak{B}$ by all IDBs. If the canonical Datalog program for $\mathfrak{B}$ derives the goal predicate on a finite structure $\mathfrak{A}$, then there is no homomorphism from $\mathfrak{A}$ to $\mathfrak{B}$ (see, e.g., [8]).

If k is the maximal arity of the EDBs, we may restrict the canonical Datalog program of width $(1, k)$ to those rules with only unary IDBs and at most one EDB; in this case, we obtain the canonical arc monadic Datalog program, which is also known as the *arc consistency procedure*. Analogously, we may define the canonical *linear*, and the canonical *symmetric* Datalog program. We may also combine these restrictions, and in particular obtain a definition the *canonical slam Datalog program*, i.e., the canonical symmetric linear arc monadic Datalog program, which has not yet been studied in the literature before. The following lemma can be shown analogously to the well-known fact for unrestricted canonical Datalog programs of width (ℓ, k) (see, e.g., [8]).

[3] Note that we do not have to exclude that ϕ_0 is the goal predicate, because we may always add its symmetric version without changing the set of structures on which the goal predicate is derived.

▶ **Lemma 6.** *Let $\mathfrak{B}$ be a finite structure with a finite relational signature, and let Π be the canonical slam Datalog program for $\mathfrak{B}$. If $\mathfrak{A}$ is a finite structure with a homomorphism to $\mathfrak{B}$, then Π does not derive the goal predicate on $\mathfrak{A}$.*

2.5 The Incidence Graph

Several results from graph theory concerning acyclicity and high girth can be generalised to general structures. To formulate these generalisations, we need the following notion. The *incidence graph* of a structure $\mathfrak{A}$ with the relational signature τ is the bipartite graph where one color class is A, and the other consists of all pairs of the form (t, R) such that $t \in R^{\mathfrak{A}}$ and $R \in \tau$. We put an edge between a and (t, R) if $t_i = a$ for some i. The *girth* of an (undirected) graph $\mathfrak{G}$ is the length of the shortest cycle in $\mathfrak{G}$. We say that a relational structure is a *generalised tree* if its incidence graph is a tree. A *leaf* of a generalised tree $\mathfrak{T}$ is an element of T which has degree one in the incidence graph of $\mathfrak{T}$. A structure $\mathfrak{B}$ is called *injective* if all tuples that are in some relation in $\mathfrak{B}$ are injective (i.e., have no repeated entries). A structure is called an *(injective) tree* if it is injective and its incidence graph is a tree.

▶ **Theorem 7** (Sparse incomparability lemma for structures [25])**.** *Let τ be a finite relational signature. Let $\mathfrak{A}$ and $\mathfrak{B}$ be τ-structure with finite domains such that $\mathfrak{A} \not\to \mathfrak{B}$. Then for every $m \in \mathbb{N}$ there exists an injective finite structure $\mathfrak{A}'$ whose incidence graph has girth at least m, such that $\mathfrak{A}' \to \mathfrak{A}$ and $\mathfrak{A}' \not\to \mathfrak{B}$.*

2.6 Dualities

For a τ-structure $\mathfrak{B}$ and a class of τ-structures $\mathcal{F}$ the pair $(\mathcal{F}, \mathfrak{B})$ is called a *duality pair* if a finite structure $\mathfrak{A}$ has a homomorphism to $\mathfrak{B}$ if and only if no structure $\mathfrak{F} \in \mathcal{F}$ has a homomorphism to $\mathfrak{A}$. Several forms of duality pairs will be relevant here, depending on the class of structures $\mathcal{F}$. A τ-structure $\mathfrak{B}$ has *finite duality* if there exists a finite set of τ-structures $\mathcal{F}$ such that $(\mathcal{F}, \mathfrak{B})$ is a duality pair. The property of having finite duality is among the very few notions studied in the context of constraint satisfaction which is *not* preserved under gadget reductions, as illustrated in the following example.

▶ **Example 8.** As in Example 4, let $\tau = \{E\}$. Let $\mathfrak{Z}_n$ be the τ-structure with the domain $\{1, 2, \ldots, 2n+3\}$ and edges $\{(1,2),(2,3),(4,3),(4,5),\ldots,(2n-2,2n-1),(n-1,n)\}$. Then

- $\mathfrak{P}_2$ has finite duality, witnessed by the duality pair $(\{\mathfrak{P}_3\}, \mathfrak{P}_2)$,
- recall from 4 that $\mathfrak{P}_2$ pp-constructs $\mathfrak{P}_3$ [9], so CSP($\mathfrak{P}_3$) has a gadget reduction to CSP($\mathfrak{P}_2$), but
- $\mathfrak{P}_3$ does not have finite duality: this is witnessed by the fact that $(\{\mathfrak{Z}_n \mid n \in \mathbb{N}\}, \mathfrak{P}_3)$ is a duality pair [27], and that there is no homomorphism from $\mathfrak{Z}_n$ to $\mathfrak{Z}_m$ for $n < m$.

▶ **Example 9.** Let $\rho = \{E, Z\}$ be the signature that consists of a binary relation symbol E and a unary relation symbol Z. Let $\mathfrak{B}_2$ be the structure with domain $\{0, 1\}$ where

$$E^{\mathfrak{B}_2} := \{(1,1),(0,1),(1,0)\} \qquad \text{and} \qquad Z^{\mathfrak{B}_2} := \{0\}$$

Let $\mathfrak{P}_2'$ be the ρ-expansion of $\mathfrak{P}_2$ where $Z^{\mathfrak{P}_2'} := \{0, 1\}$. Then $(\{\mathfrak{P}_2'\}, \mathfrak{B}_2)$ is a duality pair.

A more robust form of duality is *tree duality*, which plays a central role in constraint satisfaction, and is studied in the graph homomorphism literature in the 90s. A structure $\mathfrak{B}$ has *tree duality* if there exists a (not necessarily finite) set of trees $\mathcal{F}$ such that $(\mathcal{F}, \mathfrak{B})$ is a duality pair. The following is well known; see Theorem 7.4 in [14]. The equivalence of 1. and 3. is from [25]; also see [6].

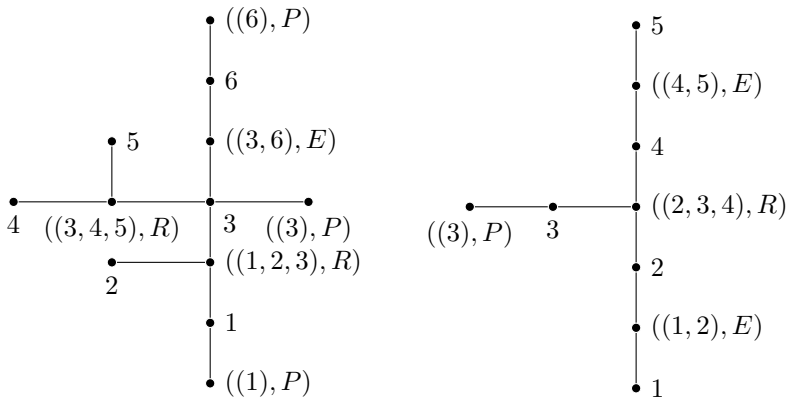

Figure 1 An example of the incidence graph of a caterpillar (left) and of a structure that is **not** a caterpillar (right).

▶ **Theorem 10.** *Let $\mathfrak{B}$ be a finite τ-structure. Then the following are equivalent:*

1. *$\mathfrak{B}$ has tree duality;*
2. *$\mathfrak{B}$ has a pp-construction from $(\{0,1\}; \{0\}, \{1\}, \{0,1\}^3 \setminus \{(1,1,0)\})$;*
3. *$\mathfrak{B}$ can be solved by arc consistency.*

There are finite structures with tree duality that have a P-complete CSP, such as the structure $(\{0,1\}; \{0\}, \{1\}, \{0,1\}^3 \setminus \{(1,1,0)\})$, which is essentially the Boolean HornSAT problem. In the following, we therefore introduce more restrictive forms of dualities.

▶ **Definition 11.** *A relational structure $\mathfrak{A}$ is called a* generalised caterpillar *if it is a generalised tree and its incidence graph $\mathfrak{G}$ contains a path $P = (a_1, \dots, a_n)$ such that every vertex in $G \setminus \{a_1, \dots, a_n\}$ of the form (t, R) is connected (in $\mathfrak{G}$) to a vertex in P.*

See Figure 1 (left) for an example. A relational structure $\mathfrak{A}$ is called an *(injective) caterpillar* if it is injective and a generalised caterpillar (this definition of a caterpillar is equivalent to the one given in [16]). A structure $\mathfrak{B}$ has *caterpillar duality* if there exists a set of caterpillars $\mathcal{F}$ such that $(\mathcal{F}, \mathfrak{B})$ is a duality pair. The structures $\mathfrak{B}$ with caterpillar duality have been characterised by [16] (Theorem 17) in terms of linear arc Datalog.

To capture the power of *symmetric* linear arc Datalog, we present a more restrictive form of duality. Let $\mathfrak{T}$ be an (injective) tree and let a and b be distinct elements of $\mathfrak{T}$. Write the canonical query of $\mathfrak{T}$ as $\phi_a \wedge \phi_{a,b} \wedge \phi_b$ where ϕ_a contains all conjuncts of the form $R(\bar{u})$ such that in the incidence graph, all paths from the vertex $(\bar{u}, R)$ to the vertex b pass though a. Similarly, we define ϕ_b, switching the roles of a and b. Note that ϕ_a and ϕ_b do not share any conjuncts. All the remaining conjuncts of the canonical query form ψ. Let ϕ_1 be obtained from ϕ_a (ϕ_b) by existentially quantifying all variables except for a (b), and let ψ be obtained from $\phi_{a,b}$ by existentially quantifying all variables except for a and b. We introduce the following concept; see Figure 2 for an example.

▶ **Definition 12.** *Let $\mathfrak{T}$ be an (injective) tree and let a and b be distinct elements of $\mathfrak{T}$ that are not leaves. The* (a,b)-unfolding of $\mathfrak{T}$ *is the canonical database of the formula*

$$\phi_1(a) \wedge \psi(a, b') \wedge \psi(a', b') \wedge \psi(a', b) \wedge \phi_2(b)$$

where ϕ_1, ϕ_2, and ψ are defined as above. A unfolding *of $\mathfrak{T}$ is a structure $\mathfrak{T}'$ that is obtained by a sequence $\mathfrak{T} = \mathfrak{T}_1, \mathfrak{T}_2, \dots, \mathfrak{T}_n = \mathfrak{T}'$ such that $\mathfrak{T}_i$ is an (a_i, b_i)-unfolding of $\mathfrak{T}_{i-1}$, for all $i \in \{2, \dots, n\}$.*

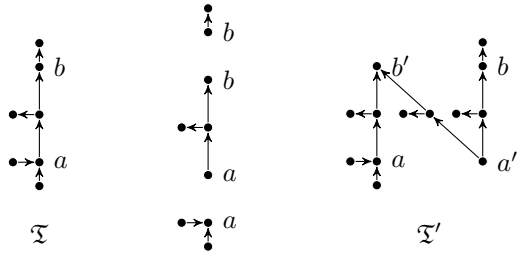

Figure 2 Example of an (a, b)-unfolding $\mathfrak{T}'$ of a tree $\mathfrak{T}$.

Note that an unfolding of a tree $\mathfrak{T}$ is again a tree and has a homomorphism to $\mathfrak{T}$. It can also be shown that the unfolding of a caterpillar is a caterpillar as well (Lemma 24). We say that a structure $\mathfrak{B}$ has *unfolded caterpillar duality* if there exists a set $\mathcal{F}$ of caterpillars such that $(\mathcal{F}, \mathfrak{B})$ is a duality pair, and $\mathcal{F}$ contains every unfolding of a caterpillar in $\mathcal{F}$. Clearly, unfolded caterpillar duality implies caterpillar duality. *Unfolded generalised caterpillar duality* is defined analogously.

2.7 Minor Conditions

If $\mathfrak{B}$ is a structure and $k \geq 1$, then a *polymorphism of* $\mathfrak{B}$ *of arity* k is a homomorphism from $\mathfrak{B}^k$ to $\mathfrak{B}$. The set of all polymorphisms of $\mathfrak{B}$ is denoted by $\mathrm{Pol}(\mathfrak{B})$. An *(operation) clone* is a set of operations which contains the projections and is closed under composition. Note that $\mathrm{Pol}(\mathfrak{B})$ is a clone. An operation $f\colon B^n \to B$ is called *idempotent* if $f(x, \dots, x) = x$ for all $x \in B$. A clone is called *idempotent* if all of its operations are idempotent.

If $\mathfrak{A}$ and $\mathfrak{B}$ are a structures and $k \geq 1$, then a *polymorphism of* $(\mathfrak{A}, \mathfrak{B})$ *of arity* k is a homomorphism from $\mathfrak{A}^k$ to $\mathfrak{B}$. The set of all polymorphisms of $(\mathfrak{A}, \mathfrak{B})$ is denoted by $\mathrm{Pol}(\mathfrak{A}, \mathfrak{B})$. Let $f\colon A^n \to B$ be a function and let $\sigma\colon [n] \to [m]$, then the map

$$f_\sigma\colon A^m \to B, (a_1, \dots, a_m) \mapsto f(a_{\sigma(1)}, \dots, a_{\sigma(n)})$$

is called a *minor* of f. A *minion* is a set or functions from A^n to B that is closed under taking minors. Note that $\mathrm{Pol}(\mathfrak{A}, \mathfrak{B})$ is a minion. Let $\mathscr{M}$ and $\mathscr{N}$ be minions. A map $\xi\colon \mathscr{M} \to \mathscr{N}$ is called a *minion homomorphism* if ξ preserves arity and for every $f \in \mathscr{M}$ of arity n and every map $\sigma\colon [n] \to [m]$ we have $\xi(f_\sigma) = (\xi(f))_\sigma$.

Let τ be a *function signature*, i.e., a set of function symbols, each equipped with an arity. A *minor condition* is a finite set Σ of *minor identities*, i.e., expressions of the form

$$f(x_1, \dots, x_n) \approx g(y_1, \dots, y_m)$$

where f is an n-ary function symbol from τ, g is an m-ary function symbol from τ, and $x_1, \dots, x_n, y_1, \dots, y_m$ are (not necessarily distinct) variables. If $\mathscr{M}$ is a minion, then a map $\xi\colon \tau \to \mathscr{M}$ *satisfies* a minor condition Σ if for every minor identity $f(x_1, \dots, x_n) \approx g(y_1, \dots, y_m) \in \Sigma$ and for every assignment $s\colon \{x_1, \dots, x_n, y_1, \dots, y_n\} \to B$ we have

$$\xi(f)(s_1(x_1), \dots, s(x_n)) = \xi(g)(s(y_1), \dots, s(y_m)).$$

We say that a minion $\mathscr{M}$ *satisfies* Σ if there exists a map $\xi\colon \tau \to \mathscr{M}$ that satisfies Σ.[4] If Σ and Σ' are minor conditions, then we say that Σ implies Σ' if every clone that satisfies Σ also satisfies Σ'. We present some concrete minor conditions that are relevant in the following.

[4] It is convenient and standard practise to notationally drop the distinction between $f \in \tau$ and $\xi(f) \in \mathscr{M}$.

▶ **Definition 13.** *An operation* $m\colon B^3 \to B$ *is called a* quasi Maltsev operation *if it satisfies the minor condition*

$$m(x,x,y) \approx m(y,x,x) \approx m(y,y,y).$$

A Maltsev operation *is an idempotent quasi Maltsev operation. A* quasi minority operation *is a quasi Maltsev operation* m *that additionally satisfies*

$$m(x,y,x) \approx m(x,x,x)$$

and a minority operation *is an idempotent quasi minority operation.*

▶ **Definition 14.** *Let* $k, n \in \mathbb{N}_{>0}$*. An operation* $f\colon B^{kn} \to B$ *is called* k-block symmetric *if it satisfies the following condition*

$$f(x_{11},\dots,x_{1k},\dots,x_{n1},\dots,x_{nk}) \approx f(y_{11},\dots,y_{1k},\dots,y_{n1},\dots,y_{nk}) \tag{1}$$

whenever $\{S_1,\dots,S_n\} = \{T_1,\dots,T_n\}$ *where* $S_i = \{x_{i1},\dots,x_{ik}\}$ *and* $T_i = \{y_{i1},\dots,y_{ik}\}$*. If* $k = 1$ *or* $n = 1$ *then* f *is called* totally symmetric*.*

If f *is* k*-block symmetric and* $S_1,\dots,S_n$ *are subsets of* B *of size at most* k*, then we also write* $f(S_1,\dots,S_n)$ *instead of* $f(x_{11},\dots,x_{1k},\dots,x_{n1},\dots,x_{nk})$ *where* $\{x_{i1},\dots,x_{ik}\} = S_i$*. We say that* f *is* k-absorptive *if it satisfies*

$$f(S_1,S_2,\dots,S_n) \approx f(S_2,S_2,S_3,\dots,S_n) \text{ whenever } S_2 \subseteq S_1.$$

▶ Remark 15. Note that every structure with a 2-absorptive polymorphism f of arity 6 also has the quasi majority polymorphism m given by $m(x,y,z) := f(x,y,z,x,y,z)$, because $\{x\} \subseteq \{x,z\}$ and hence

$$m(x,x,z) = f(x,x,z,x,x,z) = f(x,x,x,x,x,x) = m(x,x,x)$$

and similarly $m(x,z,x) = m(z,x,x) = m(x,x,x)$.

The list of equivalent statements from Theorem 10 can now be extended as follows.

▶ **Theorem 16** ([22, 25])**.** *Let* $\mathfrak{B}$ *be a finite* τ*-structure. Then* $\mathfrak{B}$ *has tree duality if and only if* $\mathfrak{B}$ *has totally symmetric polymorphisms of all arities.*

We will make crucial use of the following theorem.

▶ **Theorem 17** (Theorem 16 in [16])**.** *Let* $\mathfrak{B}$ *be a finite relational* τ*-structure. Then the following are equivalent.*

1. $\mathfrak{B}$ *has caterpillar duality.*
2. $\mathrm{CSP}(\mathfrak{B})$ *can be solved by a linear arc monadic Datalog program.*
3. $\mathrm{Pol}(\mathfrak{B})$ *contains for every* $k, n \geq 1$ *an* k*-absorbing operation of arity* kn*.*
4. $\mathfrak{B}$ *is homomorphically equivalent to a structure* $\mathfrak{B}'$ *with binary polymorphisms* $\sqcup$ *and* $\sqcap$ *such that* $(B', \sqcup, \sqcap)$ *is a (distributive) lattice.*

3 Results

In this section we state and prove our main result (Theorem 18), which characterises the power of slam Datalog in many different ways, including descriptions in terms of pp-constructability in $\mathfrak{P}_2$, minor conditions, unfolded caterpillar duality, and homomorphic equivalence to a structure with both lattice and quasi Maltsev polymorphisms.

▶ **Theorem 18.** *Let $\mathfrak{B}$ be a structure with a finite domain and a finite relational signature τ. Then the following are equivalent.*

1. $\mathrm{Pol}(\mathfrak{B})$ *contains a quasi Maltsev operation and k-absorptive operations of arity nk, for all $n, k \geq 1$.*
2. *The canonical slam Datalog program for $\mathfrak{B}$ solves* $\mathrm{CSP}(\mathfrak{B})$.
3. *Some slam Datalog program solves* $\mathrm{CSP}(\mathfrak{B})$.
4. *$\mathfrak{B}$ has unfolded caterpillar duality.*
5. *If* $\mathrm{Pol}(\mathfrak{B})$ *does not satisfy a minor condition Σ, then Σ implies $f(x) \approx f(y)$.*
6. *Every minor condition that holds in* $\mathrm{Pol}(\mathfrak{P}_2)$ *also holds in* $\mathrm{Pol}(\mathfrak{B})$.
7. *There is a minion homomorphism from* $\mathrm{Pol}(\mathfrak{P}_2)$ *to* $\mathrm{Pol}(\mathfrak{B})$.
8. *There is a pp-construction of $\mathfrak{B}$ from $\mathfrak{P}_2$.*
9. *$\mathfrak{B}$ is homomorphically equivalent to a structure $\mathfrak{B}'$ such that* $\mathrm{Pol}(\mathfrak{B}')$ *contains a quasi Maltsev operation and operations $\sqcup$ and $\sqcap$ such that $(B', \sqcup, \sqcap)$ forms a (distributive) lattice.*

We first prove the equivalence of (1)-(6) in cyclic order. We then explain how the equivalence of (6)-(8) follows from general results in the literature, and show the equivalence of (1) and (9). The proof of the theorem stretches over the following subsections.

▶ Remark 19. If one of the items of Theorem 18 holds for a structure $\mathfrak{B}$, then there exists a structure $\mathfrak{B}'$ with a binary relational signature such that $\mathrm{Pol}(\mathfrak{B}') = \mathrm{Pol}(\mathfrak{B})$, and all the statements hold for $\mathfrak{B}'$ in place of $\mathfrak{B}$ as well.

▶ **Example 20.** The structure $\mathfrak{T}_n$ is the transitive tournament with n vertices, i.e., it has the domain $[n]$ and the binary relation $<$. Note that $\mathfrak{T}_2$ equals $\mathfrak{P}_2$. It is easy to see that $(\{\mathfrak{P}_{n+1}\}, \mathfrak{T}_n)$ is a duality pair. Since $\mathfrak{P}_{n+1}$ is a caterpillar Theorem 17 implies that $\mathrm{CSP}(\mathfrak{T}_n)$ can be solved by a linear arc monadic Datalog program. However, $\mathfrak{T}_n$ does not have a quasi Maltsev polymorphism for $n \geq 3$, and hence Theorem 18 implies that $\mathrm{CSP}(\mathfrak{T}_n)$ cannot be solved by slam Datalog.

3.1 Symmetrizing Linear Arc Monadic Datalog

The following lemma is used for the implication from (1) to (2) in the proof of Theorem 18. Note that in the canonical linear arc monadic Datalog program Π we can use the "strongest possible rules"[5] when deriving the goal predicate. However, the canonical slam Datalog program Π_S might need to use weaker rules in order to be able to apply symmetric rules later on in the derivation. See Example 22.

▶ **Lemma 21.** *Let $\mathfrak{B}$ be a finite structure with relational signature τ such that* $\mathrm{Pol}(\mathfrak{B})$ *contains a quasi Maltsev operation. Let Π be the canonical linear arc monadic Datalog program for $\mathfrak{B}$ and Π_S be the canonical slam Datalog program for $\mathfrak{B}$. Then Π can derive the goal predicate on a finite τ-structure $\mathfrak{A}$ if and only if Π_S can derive the goal predicate on $\mathfrak{A}$.*

A proof of the lemma can be found in the long version of the article which is available on ArXiv [10].

[5] These comments are intended to illustrate the challenges in the proof of next lemma; it will not be necessary to formalise what we mean by strongest possible rules.

▶ **Example 22.** Consider the structure $\mathfrak{B}$ with domain $\{0, 0', 1, a, b, b'\}$, binary relation $E = \{(0,1),(0',1),(a,b),(a,b')\}$, and all constants. Let $\mathfrak{A}$ be the structure with domain $\{0, b\}$, $E = \{(0, b)\}$, and all constants. Clearly, $\mathfrak{A} \not\to \mathfrak{B}$. The canonical linear arc monadic Datalog program Π for $\mathfrak{B}$ can derive the goal predicate using the derivation

$$\vdash_{R_0} \{0\}(0) \vdash_{R_1} \{1\}(b) \vdash_{R_2} G.$$

The canonical slam Datalog program Π_S for $\mathfrak{B}$ can also derive the goal predicate on $\mathfrak{A}$. Since the rule R_1 is not symmetric it cannot use R_1. However, it may use a different rule $\tilde{R}_1$:

$$\vdash_{\tilde{R}_0} \{0, 0'\}(0) \vdash_{\tilde{R}_1} \{1\}(b) \vdash_{R_2} G.$$

Note that R_0 is also a rule of Π_S but in order to apply $\tilde{R}_1$ the program needs to use the rule $\tilde{R}_0$ which is weaker than the rule R_0 (in the sense that the derived IDB is a strict superset).

3.2 Unfolded Caterpillar Duality

This section is devoted to the implication (3) to (4) in Theorem 18; the full proofs can be found in the long version of the article which is available on ArXiv [10]. We first present a general result about obstruction sets for finite-domain CSPs that closes a gap in the presentation of the proof of Lemma 21 in [16] and essentially follows from the sparse incomparability lemma (Theorem 7); we thank Víctor Dalmau for clarification.

▶ **Lemma 23.** *Let $\mathfrak{B}$ be a finite structure and $\mathcal{F}$ a class of finite structures such that $(\mathcal{F}, \mathfrak{B})$ is a duality pair. Define $\mathcal{F}' := \{\mathfrak{F} \in \mathcal{F} \mid \mathfrak{F}$ is injective$\}$. Then $(\mathcal{F}', \mathfrak{B})$ is a duality pair.*

Note that Theorem 17 implies that if CSP($\mathfrak{B}$) is solved by a linear arc monadic Datalog program, then $\mathfrak{B}$ has caterpillar duality; the proof given in [16] only shows generalised caterpillar duality. However, Lemma 23 implies that $\mathfrak{B}$ in this case also has (injective) caterpillar duality.

In order to prove the implication (3) to (4) in Theorem 18, it only remains to prove that $\mathfrak{B}$ also has *unfolded* caterpillar duality (Lemma 25). We need the following lemma, which has already been mentioned in Section 2.6.

▶ **Lemma 24.** *An unfolding of a caterpillar is a caterpillar as well.*

▶ **Lemma 25.** *If* CSP($\mathfrak{B}$) *is solved by a slam Datalog program* Π*, then $\mathfrak{B}$ has unfolded caterpillar duality.*

From this we obtain the implication (4) to (5) in Theorem 18.

▶ **Lemma 26.** *Let $\mathfrak{B}$ be a relational structure with unfolded caterpillar duality. If* Pol($\mathfrak{B}$) *does not satisfy a minor condition Σ, then Σ implies $f(x) \approx f(y)$.*

3.3 Proof of the Main Result

We finally prove Theorem 18.

Proof of Theorem 18. For the implication (1) $\Rightarrow$ (2) let Π_S be the canonical slam Datalog program for $\mathfrak{B}$ and let Π be the canonical linear arc monadic Datalog program for $\mathfrak{B}$. Since $\mathfrak{B}$ has k-absorptive operations of arity nk for all $n, k \geq 1$ we can apply Theorem 17 to

conclude that Π solves CSP($\mathfrak{B}$). Furthermore, $\mathfrak{B}$ has a quasi Maltsev polymorphism. Hence, Lemma 21 implies that Π and Π_S can derive the goal predicate on the same instances of CSP($\mathfrak{B}$). Therefore, Π_S solves CSP($\mathfrak{B}$).

The implication (2) ⇒ (3) is trivial, (3) ⇒ (4) is Lemma 25, and (4) ⇒ (5) is Lemma 26. For the implication from (5) to (6), suppose that Σ is a minor condition that holds in Pol($\mathfrak{P}_2$). Since all polymorphisms of $\mathfrak{P}_2$ are idempotent, Σ does not imply $f(x) \approx f(y)$. Hence, the contraposition of (5) implies that Pol($\mathfrak{B}$) does not satisfy Σ.

The implication from (6) to (1) is clear since Pol($\mathfrak{P}_2$) is preserved by the Boolean minority operation and by the k-absorptive nk-ary Boolean operation

$$(x_{11}, \dots, x_{1k}, \dots, x_{n1}, \dots, x_{nk}) \mapsto \bigvee_{i \in [n]} \bigwedge_{j \in [k]} x_{ij}$$

The equivalence between (6), (7), and (8) follows from Remark 5 and well-known general results [5].

The equivalence of (1) and (9) follows the equivalence of items (3) and (4) in Theorem 17 and from the fact that the existence of a quasi Maltsev polymorphism is preserved by homomorphic equivalence. ◀

▶ Remark 27. Consider the poset of all finite structures ordered by primitive positive constructability. It is well known that the structure $\mathfrak{C}_1 := (\{0\}, \{(0,0)\})$ is a representative of the top element of this poset and that it has exactly one lower cover with representative $\mathfrak{B}_2$. We claim that $\mathfrak{T}_3$ is a representative of a lower cover of $\mathfrak{B}_2$ in the poset of all finite structures ordered by primitive positive constructability. The structure $\mathfrak{T}_3 := (\{0,1,2\}, \{(0,1),(0,2),(1,2)\})$ satisfies all conditions Σ that do not imply the quasi Maltsev condition (see, e.g., [9]). Clearly, $\mathfrak{T}_3$ does not have a primitive positive construction from $\mathfrak{B}_2$, because $\mathfrak{T}_3$ does not have a quasi Maltsev polymorphism. Since min and max are polymorphisms of $\mathfrak{T}_3$, Theorem 17 implies that $\mathfrak{T}_3$ has kn-ary k-absorbing polymorphisms for all $n, k \geq 1$. Let $\mathfrak{B}$ be a structure with a primitive positive construction from $\mathfrak{T}_3$ such that $\mathfrak{T}_3$ does not have a pp-construction from $\mathfrak{B}$. Then $\mathfrak{B}$ must have a quasi Maltsev polymorphism and kn-ary k-absorbing polymorphisms for all $n, k \geq 1$. By Theorem 18, $\mathfrak{B}$ has a primitive positive construction from $\mathfrak{B}_2$, which proves the claim. It is still open what other lower covers $\mathfrak{B}_2$ has.

▶ Remark 28. Another condition on finite structures $\mathfrak{B}$ that is equivalent to the conditions in Theorem 18 has been found by Vucaj and Zhuk [39]: they prove that there is a minion homomorphism from Pol($\mathfrak{P}_2$) to Pol($\mathfrak{B}$) (item 7) if and only if Pol($\mathfrak{B}$) contains totally symmetric polymorphisms of all arities and *generalised quasi minority polymorphisms* of all odd arities $n \geq 3$. A operation $f\colon B^n \to B$, for odd $n \geq 3$, is called a *generalised quasi minority* if it satisfies

$$f(x_1, \dots, x_n) \approx f(x_{\pi(1)}, \dots, x_{\pi(n)}) \qquad \text{for every } \pi \in S_n$$
$$\text{and } f(x, x, x_3, \dots, x_n) \approx f(y, y, x_3, x_4, \dots, x_n).$$

However, it is not clear to us whether this characterisation can be used to prove the consequence of our main result from Remark 27.

▶ Remark 29. Yet another remarkable equivalent condition for solvability by slam Datalog was very recently discovered by Meyer and Starke [34]: the conditions from Theorem 18 hold if and only if $\mathfrak{B}$ can neither pp-construct $\mathfrak{T}_3$ nor any structure from a list of structures that correspond to the finite simple groups with a particular action; for details, we refer to [34].

4 Decidability of Meta-Problem

There are many interesting results and open problems about *algorithmic meta-problems* in constraint satisfaction; we refer to [18]. The natural algorithmic meta-problem in the context of our work is the one addressed in the following proposition.

▶ **Proposition 30.** *There is an algorithm which decides in deterministic doubly-exponential time whether the CSP of a given finite structure $\mathfrak{B}$ can be solved by a slam Datalog program, and if so, computes such a program.*

Proof. The following algorithm can be used to test whether $\mathfrak{B}$ has k-absorptive operations of arity nk, for all $n, k \geq 1$. It is well-known that the existence of a quasi Maltsev polymorphism can be decided in non-deterministic polynomial time (see, e.g., [18]). Hence, the statement then follows from Theorem 18.

Let m be the maximal arity of the relations of $\mathfrak{B}$. Let $n_0 := m\binom{|B|}{|B|/2}$ and $k_0 := m|B|$. Note that $\mathfrak{B}$ has k-absorptive polymorphisms of arity nk, for all k, n, if and only if it has k_0-absorptive polymorphisms of arity $n_0 k_0$ (similarly as the well-known fact that $\mathfrak{B}$ has totally symmetric polymorphisms of all arities if and only if it has totally symmetric polymorphisms of arity $m|B|$; the term $\binom{|B|}{|B|/2}$ bounds the size of antichains in the set of all subsets of B. Also see [15] for the case of k-absorptive polymorphisms). Let Σ be the minor condition for the existence of k_0-absorptive operations of arity $n_0 k_0$. Let $\mathfrak{C}$ be the indicator structure of Σ with respect to $\mathfrak{B}$ as defined in Section A.1; clearly, this structure can be computed in doubly exponential time. We may then find a non-deterministic algorithm with the same time bound that tests whether there exists a homomorphism from $\mathfrak{C}$ to $\mathfrak{B}$. The non-determinism for checking whether $\mathfrak{C} \to \mathfrak{B}$ can be eliminated by standard self-reduction techniques (again see, e.g., [18]). For the second part of the statement, note that there are for a given $\mathfrak{B}$ only finitely many potential rules of a slam Datalog program, and one can compute for a given rule whether it is part of the canonical slam Datalog program of CSP($\mathfrak{B}$). ◀

5 Conclusion and Open Problems

We characterised the unique submaximal element in the primitive positive constructability poset on finite structures, linking concepts from homomorphism dualities, Datalog fragments, minor conditions, and minion homomorphisms. It is now tempting to further descend in the pp-constructability poset of finite structures in order to obtain a more systematic understanding. Particularly attractive are other dividing lines in the poset that are relevant for the complexity of the constraint satisfaction problem. We propose the following problems for future research.

- Characterise all finite structures that are primitively positively constructible in a finite structure that has finite duality. Are these exactly the finite structures whose polymorphism clones have Hagemann-Mitschke chains of some length and *extended k-absorptive polymorphisms of arity $kn+1$*, for all $n, k \geq 1$, as defined in [16]? Is there a Datalog fragment that corresponds to this class?
- What is the precise computational complexity the Meta-Problem of deciding whether the CSP of a given finite structure $\mathfrak{B}$ can be solved by a slam Datalog program? Proposition 30 only provides a deterministic doubly exponential time algorithm.

References

1 Foto N. Afrati and Stavros S. Cosmadakis. Expressiveness of restricted recursive queries (extended abstract). In David S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA*, pages 113–126. ACM, 1989. doi:10.1145/73007.73018.

2 Albert Atserias, Andrei A. Bulatov, and Anuj Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410(18):1666–1683, 2009. doi:10.1016/J.TCS.2008.12.049.

3 L. Barto, M. Kozik, and R. Willard. Near unanimity constraints have bounded pathwidth duality. In *Proceedings of the 27th ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 125–134, 2012.

4 Libor Barto and Marcin Kozik. Constraint satisfaction problems of bounded width. In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*, pages 595–603, 2009. doi:10.1109/FOCS.2009.32.

5 Libor Barto, Jakub Opršal, and Michael Pinsker. The wonderland of reflections. *Israel Journal of Mathematics*, 223(1):363–398, 2018.

6 Manuel Bodirsky. Graph homomorphisms and universal algebra. Lecture Notes, https://wwwpub.zih.tu-dresden.de/~bodirsky/GH-UA.pdf, 2023.

7 Manuel Bodirsky, Jakub Bulín, Florian Starke, and Michael Wernthaler. The smallest hard trees. *Constraints*, abs/2205.07528, 2022. doi:10.48550/arXiv.2205.07528.

8 Manuel Bodirsky and Víctor Dalmau. Datalog and constraint satisfaction with infinite templates. *Journal on Computer and System Sciences*, 79:79–100, 2013. A preliminary version appeared in the proceedings of the Symposium on Theoretical Aspects of Computer Science (STACS'05). doi:10.1016/J.JCSS.2012.05.012.

9 Manuel Bodirsky and Florian Starke. Maximal digraphs with respect to primitive positive constructability. *Combinatorica*, 42:997–1010, 2022. doi:10.1007/S00493-022-4918-1.

10 Manuel Bodirsky and Florian Starke. Symmetric linear arc monadic datalog and gadget reductions. https://arxiv.org/abs/2407.04924, 2025.

11 Manuel Bodirsky, Florian Starke, and Albert Vucaj. Smooth digraphs modulo primitive positive constructability and cyclic loop conditions. *International Journal on Algebra and Computation*, 31(5):939–967, 2021. Preprint available at ArXiv:1906.05699.

12 Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.

13 Andrei A. Bulatov, Andrei Krokhin, and Benoit Larose. *Dualities for Constraint Satisfaction Problems*, pages 93–124. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. doi:10.1007/978-3-540-92800-3_5.

14 Jakub Bulín, Andrei A. Krokhin, and Jakub Opršal. Algebraic approach to promise constraint satisfaction. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 602–613, 2019. doi:10.1145/3313276.3316300.

15 Catarina Carvalho, Víctor Dalmau, and Andrei Krokhin. CSP duality and trees of bounded pathwidth. *Theoretical Computer Science*, 411:3188–3208, 2010. doi:10.1016/J.TCS.2010.05.016.

16 Catarina Carvalho, Víctor Dalmau, and Andrei A. Krokhin. Two new homomorphism dualities and lattice operations. *J. Log. Comput.*, 21(6):1065–1092, 2011. doi:10.1093/LOGCOM/EXQ030.

17 Ashok K. Chandra and Philip M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 77–90, 1977. doi:10.1145/800105.803397.

18 Hubie Chen and Benoît Larose. Asking the metaquestions in constraint tractability. *TOCT*, 9(3):11:1–11:27, 2017. doi:10.1145/3134757.

19 Víctor Dalmau. Linear Datalog and bounded path duality of relational structures. *Logical Methods in Computer Science*, 1(1), 2005. doi:10.2168/LMCS-1(1:5)2005.

20 Víctor Dalmau and Benoît Larose. Maltsev + Datalog → symmetric Datalog. In *Proceedings of the Twenty-Third Annual IEEE Symposium on Logic in Computer Science, LICS 2008, 24-27 June 2008, Pittsburgh, PA, USA*, pages 297–306. IEEE Computer Society, 2008.

21 Victor Dalmau and Jakub Opršal. Local consistency as a reduction between constraint satisfaction problems, 2023. arXiv:2301.05084.

22 Víctor Dalmau and Justin Pearson. Closure functions and width 1 problems. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP)*, pages 159–173, 1999. doi:10.1007/978-3-540-48085-3_12.

23 László Egri, Benoit Larose, and Pascal Tesson. Symmetric Datalog and constraint satisfaction problems in logspace. In *Proceedings of the Symposium on Logic in Computer Science (LICS)*, pages 193–202, 2007.

24 László Egri, Benoît Larose, and Pascal Tesson. Directed st-connectivity is not expressible in symmetric Datalog. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 172–183, 2008. doi:10.1007/978-3-540-70583-3_15.

25 Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM Journal on Computing*, 28:57–104, 1999. doi:10.1137/S0097539794266766.

26 P. Hell, J. Nešetřil, and X. Zhu. Duality and polynomial testing of tree homomorphisms. *TAMS*, 348(4):1281–1297, 1996.

27 Pavol Hell and Jaroslav Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, Oxford, 2004.

28 Wilfrid Hodges. *A shorter model theory*. Cambridge University Press, Cambridge, 1997.

29 P. C. Kanellakis. *Logic programming and parallel complexity*, pages 547–585. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. Book chapter in 'Foundations of Deductive Databases and Logic Programming'.

30 Alexandr Kazda. n-permutability and linear Datalog implies symmetric Datalog. *Logical Methods in Computer Science*, Volume 14, Issue 2, April 2018. doi:10.23638/LMCS-14(2:3)2018.

31 Marcin Kozik, Andrei Krokhin, Matt Valeriote, and Ross Willard. Characterizations of several Maltsev conditions. *Algebra universalis*, 73(3):205–224, 2015. doi:10.1007/s00012-015-0327-2.

32 Benoit Larose and Pascal Tesson. Universal algebra and hardness results for constraint satisfaction problems. *Theoretical Computer Science*, 410(18):1629–1647, 2009. doi:10.1016/J.TCS.2008.12.048.

33 Benoit Larose and László Zádori. Bounded width problems and algebras. *Algebra Universalis*, 56(3-4):439–466, 2007.

34 Sebastian Meyer and Florian Starke. Finite simple groups in the primitive positive constructability poset, 2024. arXiv:2409.06487.

35 Reinhard Pöschel and Lev A. Kalužnin. *Funktionen- und Relationenalgebren*. Deutscher Verlag der Wissenschaften, Berlin, 1979.

36 Benjamin Rossman. Homomorphism preservation theorems. *Journal of the ACM*, 55(3), 2008. doi:10.1145/1379759.1379763.

37 Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 216–226, 1978. doi:10.1145/800133.804350.

38 Florian Starke. Digraphs modulo primitive positive constructability. Preprint, 2024. PhD dissertation, Institute of Algebra, TU Dresden.

39 Albert Vucaj and Dmitriy Zhuk. Submaximal clones over a three-element set up to minor-equivalence, 2024. arXiv:2304.12807.

40 Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *J. ACM*, 67(5):30:1–30:78, 2020. doi:10.1145/3402029.

41 Dmitriy N. Zhuk. A proof of CSP dichotomy conjecture. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17*, pages 331–342, 2017. URL: https://arxiv.org/abs/1704.01914.

A Appendix

A.1 Indicator Structures

In this section we revisit a common theme in constraint satisfaction, the concept of an *indicator structure* of a minor condition. To simplify the presentation, we only define the indicator structure for minor conditions with only one function symbol. For our purposes, this is without loss of generality, because for clones over a finite domain, every minor condition is equivalent to such a restricted minor condition. If $f\colon C^n \to C$ and $g\colon C^m \to C$ are operations, then the *star product* $f * g$ is defined to be the operation defined as

$$(x_{1,1}, \dots, x_{n,m}) \mapsto f(g(x_{1,1}, \dots, x_{1,m}), \dots, g(x_{n,1}, \dots, x_{n,m})).$$

▶ **Lemma 31.** *Let Σ be a minor condition. Then there exists a minor condition Σ' with a single function symbol such that a clone over a finite domain satisfies Σ if and only if it satisfies Σ'.*

Proof. First note that for every clone $\mathscr{D}$ on a finite set there exists an *idempotent* clone $\mathscr{C}$ on a finite set which is equivalent to it with respect to minion homomorphisms, i.e., there are minion homomorphism from $\mathscr{D}$ to $\mathscr{C}$ and vice versa. It is well-known and easy to see that if $f_1, \dots, f_n$ are the function symbols that appear in Σ, and $\mathscr{C}$ satisfies Σ, $\mathscr{C}$ also contains an operation g of arity m such that for every $i \in [n]$ there exists $\alpha_i\colon [m] \to [k]$ such that $g_{\alpha_i} = f_i$ (use that $\mathscr{C}$ is closed under the star product and idempotent). Note that $\mathscr{C}$ satisfies a minor identity $(f_i)_\beta \approx (f_j)_\gamma$ if and only if $\mathscr{C}$ satisfies a minor identity $(g_{\alpha_i})_\beta \approx (g_{\alpha_j})_\gamma$. ◀

If $\mathfrak{B}$ is a relational τ-structure and $\sim$ is an equivalence relation on B, the $\mathfrak{B}/_\sim$ is the τ-structure whose domain are the equivalence classes of $\sim$, and where $R(C_1, \dots, C_k)$ holds if there exist $a_1 \in C_1, \dots, a_k \in C_k$ such that $R(a_1, \dots, a_k)$ holds in $\mathfrak{B}$.

▶ **Definition 32.** *Let $\mathfrak{B}$ be a relational τ-structure and let Σ be a minor condition with a single function symbol f of arity m. Let $\sim$ be the smallest equivalence relation on B^m such that $a \sim b$ if Σ contains $f(x_1, \dots, x_m) \approx f(y_1, \dots, y_m)$ such that there is a map $s\colon \{x_1, \dots, x_m, y_1, \dots, y_m\} \to B$ with $a = (s(x_1), \dots, s(x_m))$ and $b = (s(y_1), \dots, s(y_m))$. Then the* indicator structure of Σ with respect to $\mathfrak{B}$ *is the τ-structure $\mathfrak{B}^m/_\sim$.*

The following is straightforward from the definitions.

▶ **Lemma 33.** *Let $\mathfrak{B}$ be a structure and Σ be a minor condition with a single function symbol f. Then $\mathfrak{B}$ has a polymorphism satisfying Σ if and only if the indicator structure of Σ with respect to $\mathfrak{B}$ has a homomorphism to $\mathfrak{B}$.*

B Remarks on Related Results

The following remarks show that the results of Carvalho, Dalmau and Krokhin [15] can be extended in the same spirit as our Theorem 18.

▶ Remark 34. Let $\mathfrak{D}_2$ be the structure $(\{0,1\};\{0\},\{1\},\leq)$, also known as st-Con. Theorem 17 of Carvalho, Dalmau and Krokhin can be extended in the same spirit as our Theorem 18, by adding the following equivalent items:

5. Every minor condition that holds in $\mathrm{Pol}(\mathfrak{D}_2)$ also holds in $\mathrm{Pol}(\mathfrak{B})$.
6. There is a minion homomorphism from $\mathrm{Pol}(\mathfrak{D}_2)$ to $\mathrm{Pol}(\mathfrak{B})$.
7. $\mathfrak{B}$ has a primitive positive construction from $\mathfrak{D}_2$.

Proof. The equivalence of 5., 6., and 7. follows immediately from the general results in [5].

4. $\Rightarrow$ 7. It is well known that $\mathrm{Pol}(\mathfrak{D}_2)$ is generated by the two binary operations $\vee$ and $\wedge$.[6] Let $\mathfrak{B}$ be a structure that is homomorphically equivalent to a structure $\mathfrak{B}'$ with binary polymorphisms $\sqcup$ and $\sqcap$ such that $(B',\sqcup,\sqcap)$ is a distributive lattice. Note that $(\{0,1\},\vee,\wedge)$ is a distributive lattice as well. Let ι be the map that maps terms over $\vee,\wedge$ to terms over $\sqcup,\sqcap$ by replacing $\vee$ and $\wedge$ by $\sqcup$ and $\sqcap$, respectively. Define the map $\xi\colon \mathrm{Pol}(\mathfrak{D}_2)\to\mathrm{Pol}(\mathfrak{B}')$ as follows. Since $\mathrm{Pol}(\mathfrak{D}_2)$ is generated by $\vee$ and $\wedge$, for every $f\in\mathrm{Pol}(\mathfrak{D}_2)$ there is a $\{\wedge,\vee\}$-term t whose term operation is f. Define $\xi(f)$ as the term operation of $\iota(t)$. Note that this term operation is a polymorphism of $\mathfrak{B}'$. It is clear that ξ is a minion homomorphism (even a clone homomorphism). We still need to show that ξ is well defined. Let t and t' be two $\{\wedge,\vee\}$-terms that both have the term operation $f\in\mathrm{Pol}(\mathfrak{D}_2)$. Since $(\{0,1\},\vee,\wedge)$ is a distributive lattice, there is a set $\mathcal{I}$ of subsets of $[n]$ such that f is the term operation of $s:=\bigwedge_{I\in\mathcal{I}}\bigvee_{i\in I}x_i$. Furthermore, t and t' can both be rewritten (using associativity, commutativity, distributivity, and idempotence) into the term s. Therefore, $\iota(t)$ and $\iota(t')$ can also both be rewritten into the term $\iota(s)$. Since $(B',\sqcup,\sqcap)$ is a distribute lattice, the term operations of $\iota(t)$, $\iota(t')$, and $\iota(s)$ are the same. Hence, ξ is well defined.

5. $\Rightarrow$ 4. holds since $\mathfrak{D}_2$ has for every $n,k\geq 1$ a k-absorbing polymorphism of arity kn. ◀

▶ Remark 35. We consider the structure $\mathfrak{B}^{\leq}_{\infty}=(\{0,1\},\mathbf{0},\leq,R_1,R_2,\dots)$ where $\mathbf{0}:=\{0\}$, $\leq:=\{(0,0),(0,1),(1,1)\}$, and $R_n:=\{0,1\}^n\setminus\{(0,\dots,0)\}$ for every $n\geq 1$. It is well known that $\mathrm{Pol}(\mathfrak{B}^{\leq}_{\infty})$ is generated by the operation m given by $(x,y,z)\mapsto x\wedge(y\vee z)$. Carvalho, Dalmau and Krokhin also introduce another type of duality in their paper: jellyfish duality. Their characterization in Theorem 18 in [15] can be extended by the following items:

6. Every minor condition that holds in $\mathrm{Pol}(\mathfrak{B}^{\leq}_{\infty})$ also holds in $\mathrm{Pol}(\mathfrak{B})$.
7. There is a minion homomorphism from $\mathrm{Pol}(\mathfrak{B}^{\leq}_{\infty})$ to $\mathrm{Pol}(\mathfrak{B})$.
8. $\mathfrak{B}$ has a primitive positive construction from $\mathfrak{B}^{\leq}_{\infty}$.

The proof of Remark 35 is mostly analogous to the previous one. Here we only sketch the proof that $\mathrm{Pol}(\mathfrak{B}^{\leq}_{\infty})$ is generated by the operation m given by $(x,y,z)\mapsto x\wedge(y\vee z)$.

Clearly, every relation of $\mathfrak{B}^{\leq}_{\infty}$ is preserved by the operation m. For the converse inclusion, it suffices to verify that every relation that is preserved by m has a primitive positive definition in $\mathfrak{B}^{\leq}_{\infty}$ (see, e.g., [35]). First note that $m(x,y,y)=x\wedge y$, and hence every Boolean relation

[6] Proof sketch: clearly, $\vee$ and $\wedge$ preserve the relations of $\mathfrak{D}_2$. For the converse inclusion, it suffices to verify that every relation that is preserved by $\wedge$ and $\vee$ has a primitive positive definition in $\mathfrak{D}_2$ (see, e.g. [35]). Every Boolean relation preserved by $\vee$ and $\wedge$ has a definition in CNF which is both Horn and dual Horn, so consists of clauses that can be defined using the relations in $\mathfrak{D}_2$. This implies the claim.

R preserved by m has a Horn definition; pick such a definition ϕ which is shortest possible. Suppose for contradiction that a Horn clause in ϕ contains a positive literal ψ_1 and two negative literals ψ_2 and ψ_3. By the minimality assumption there are tuples $t_1, t_2, t_3 \in R$ such that t_i satisfies ϕ_i and no other literal in that clause. Then $m(t_1, t_2, t_3)$ satisfies none of ψ_1, ψ_2, ψ_3, a contradiction. It follows that each clause can be defined using the relations in $\mathfrak{B}^{\leq}_{\infty}$ and the statement follows.