Maltsev + Datalog $\implies$ Symmetric Datalog *

Victor Dalmau
Universitat Pompeu Fabra, Barcelona, Spain
victor.dalmau@upf.edu

Benoit Larose
Concordia University, Montréal, Canada
larose@mathstat.concordia.ca

Abstract

Let $\mathbf{B}$ be a finite, core relational structure and let $\mathbb{A}$ be the algebra associated to $\mathbf{B}$, i.e. whose terms are the operations on the universe of $\mathbf{B}$ that preserve the relations of $\mathbf{B}$. We show that if $\mathbb{A}$ generates a so-called arithmetical variety then $\mathrm{CSP}(\mathbf{B})$, the constraint satisfaction problem associated to $\mathbf{B}$, is solvable in Logspace; in fact $\neg\mathrm{CSP}(\mathbf{B})$ is expressible in symmetric Datalog. In particular, we obtain that if $\neg\mathrm{CSP}(\mathbf{B})$ is expressible in Datalog and the relations of $\mathbf{B}$ are invariant under a Maltsev operation then $\neg\mathrm{CSP}(\mathbf{B})$ is in symmetric Datalog.

1 Introduction and Statement of the Main Results

Constraint satisfaction problems (or CSPs) provide a natural and flexible framework to study the complexity of various combinatorial problems arising naturally in optimisation, graph theory, artificial intelligence and database theory. Simply put, an instance of a CSP consists of a finite set of variables together with constraints on these, and the problem consists of determining whether values from a specified domain can be assigned to the variables to satisfy all constraints. Although the general problem is NP-complete, restricting the nature of the constraints can lead to tractable cases. For convenience, we shall adopt the standard approach of viewing these so-called non-uniform CSP's as homomorphism problems: given a finite relational structure $\mathbf{B}$, let $\mathrm{CSP}(\mathbf{B})$ denote the class of all finite structures that admit a homomorphism to $\mathbf{B}$. The past few years have witnessed a flurry of activity in the study of these CSP's, fueled in part by a tantalising conjecture, due to Feder and Vardi [12], stating that for every structure $\mathbf{B}$, the problem $\mathrm{CSP}(\mathbf{B})$ is either solvable in polynomial time or NP-complete. The conjecture is known to hold in the 2-element [20] and 3-element [3] cases, and several other special cases (see [7]). Even though the conjecture is still open after almost 15 years, remarkable progress has been made in understanding the complexity of $\mathrm{CSP}(\mathbf{B})$ thanks to two complementary approaches that have now come to be inextricably linked.

The first of these approaches seeks to classify the CSP's according to the nature of the logics required to describe the set of structures that admit (or do not admit) a homomorphism to $\mathbf{B}$. In particular, it has been noticed that a number of tractable cases can be captured by definability of $\neg\mathrm{CSP}(\mathbf{B})$ in the database-inspired query language Datalog. If, furthermore, $\neg\mathrm{CSP}(\mathbf{B})$ is definable in linear Datalog then the corresponding problem can be solved in NL and some evidence was given in [8] that this condition is in fact necessary and sufficient. Going further down the hierarchy, symmetric Datalog is a fragment of linear Datalog that guarantees that, if $\neg\mathrm{CSP}(\mathbf{B})$ is definable in it then $\mathrm{CSP}(\mathbf{B})$ is solvable in Logspace [11]; all

*The first author is supported by grants TIN2004-04343 and TIN2006-15387-C03-03 funded by the MCyT. The second author's research is supported by grants from NSERC, FQRNT and CRM.

known CSP's in Logspace are of this form. Finally, it is known that every CSP which is not hard for Logspace must be in non-uniform AC^0 and in fact has finite duality [15]. It is known that the above scheme gives a full account of Datalog and its fragments for Boolean CSP's [15].

The second approach associates to every finite structure $\mathbf{B}$ an algebra $\mathbb{A} = \mathbb{A}(\mathbf{B})$, whose universe A is that of $\mathbf{B}$ and whose basic operations are those that are compatible with the relations of $\mathbf{B}$. Viewed this way, the constraint relations of the CSP are subuniverses of subalgebras of powers of $\mathbb{A}$, and thus the equational properties of the algebra control in some sense the "geometry" of the constraints; the presence of terms of the algebra obeying stringent identities, such as a semilattice term for instance, ensures that the associated CSP is tractable (see [7] for a comprehensive survey.)

Tame Congruence Theory, first developed in the mid 80's, is a powerful tool to analyse equational classes (a.k.a. varieties) generated by finite algebras [13]. To each finite algebra is associated a set of among 5 *types*: (1) the unary type, (2) the affine type, (3) the Boolean type, (4) the lattice type and (5) the semilattice type. The typeset of an algebra reflects the local behaviour of its polynomial operations; the typeset of a variety is the union of all typesets of its finite algebras, and is an indicator of how well-behaved the variety is. Four natural classes of varieties are delineated by the so-called omitting-types theorems of [13]. Assuming without loss of generality the structures involved are cores, the following lists some consequences on the associated CSP whose algebra $\mathbb{A}$ generates a variety $\mathcal{V}(\mathbb{A})$ whose typeset contains certain types:

- if $\mathcal{V}(\mathbb{A})$ admits type 1, then $\mathrm{CSP}(\mathbf{B})$ is NP-complete and its complement is not Datalog definable;
- if $\mathcal{V}(\mathbb{A})$ admits type 1 or 2, then $\mathrm{CSP}(\mathbf{B})$ is mod_p L-hard for some p and its complement is not Datalog definable;
- if $\mathcal{V}(\mathbb{A})$ admits type 1, 2 or 5, then $\mathrm{CSP}(\mathbf{B})$ is P-hard and its complement is not definable in linear Datalog;
- if $\mathcal{V}(\mathbb{A})$ admits type 1, 2, 4 or 5, then $\mathrm{CSP}(\mathbf{B})$ is NL-hard and its complement is not definable in symmetric Datalog.

It is known that this result essentially gives the whole picture in the Boolean case (see [15]); for instance every Boolean CSP whose variety has typeset $\{3, 4\}$ has its complement in linear Datalog and hence is in NL. It is conceivable that this state of affairs could extend to arbitrary CSP's: in particular, that varieties omitting type 1 should give rise to tractable CSP's is the algebraic version of the Feder Vardi dichotomy conjecture and was first stated by Bulatov, Jeavons and Krokhin in [2]; it was conjectured in [16] that CSP's whose variety omits types 1 and 2 should have their complement in Datalog (the *bounded width conjecture*).

Early on algebraists recognised the importance of the nature of the congruence lattices of algebras in the classification of varieties: among the chief conditions are congruence-modularity, -distributivity and -permutability. It is known that the congruence-permutability of a variety is characterised by the presence of a so-called *Maltsev term*, i.e. a ternary term m satisfying the identities

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

in the prototypical case of groups, this is easily witnessed by the term $m(x, y, z) = xy^{-1}z$. CSP's with a congruence-permutable variety are solvable in polynomial time [4, 5]; one of the striking achievements of the algebraic method is a far-reaching extension of this result to so-called algebras with few subpowers [14]. There is a strong link between typesets of varieties and their congruence properties. For instance, it is known that (idempotent, locally finite) varieties whose typeset is $\{2, 3\}$ are precisely those that are n-permutable for some $n \geq 2$, where n-permutability is a natural generalisation of congruence (2)-permutability [13].

The present paper is a first step in the direction of a solution to the fourth conjecture, namely that if a CSP has a variety with typeset $\{3\}$, then its complement is definable in symmetric Datalog. A variety $\mathcal{V}$ is *arithmetical* if every algebra in it is congruence-permutable and congruence-distributive. Since congruence-distributive varieties omit types 1, 2 (and 5), according to the bounded width conjecture they should give rise to CSP's with complement in Datalog; this has been verified only in the so-called CD(4) case [6]. As pointed out earlier, a CSP with a congruence-permutable variety with typeset $\{2, 3\}$ is tractable, but the presence of type 2 prevents its complement being in Datalog. However if both congruence conditions are combined, we are able to show much more:

Theorem 1 *Let $\mathbf{B}$ be a finite, core structure. If $\mathbb{A}(\mathbf{B})$ generates an arithmetical variety then $\neg\mathrm{CSP}(\mathbf{B})$ is definable in symmetric Datalog. In particular, $\mathrm{CSP}(\mathbf{B})$ is in Logspace.*

We will show that this has the following consequence:

Corollary 2 *Let $\mathbf{B}$ be a finite, core structure. If $\mathbb{A}(\mathbf{B})$ generates a congruence-permutable variety and $\neg\mathrm{CSP}(\mathbf{B})$ is definable in Datalog, then $\neg\mathrm{CSP}(\mathbf{B})$ is definable in symmetric Datalog. In particular, $\mathrm{CSP}(\mathbf{B})$ is in Logspace.*

Assuming the bounded width conjecture is correct, if a CSP has a type 3 variety then its complement is in Datalog and the variety is n-permutable for some $n \geq 2$. The corollary can then be viewed as the case $n = 2$ towards showing the general type 3 conjecture.

Algebras that generate arithmetical varieties include Boolean algebras and Heyting algebras; our result also generalises the case of quasiprimal algebras first proved in [11].

The next section contains the basic facts about relational structures and various fragments of Datalog we require; it will be followed by an overview of the algebraic results necessary to state and prove our main theorem and its corollary.

2 Constraint Satisfaction Problems and Fragments of Datalog

For basic terminology and notation concerning relational structures we refer the reader to [17]. Given relational structures $\mathbf{A}$, $\mathbf{B}$, ... we denote their respective universes by A, B, ... Let $\tau = \{R_1, \dots, R_n\}$ be a vocabulary and let $\mathbf{A}$ and $\mathbf{B}$ be τ-structures. A *homomorphism* f from $\mathbf{A}$ to $\mathbf{B}$ is a map $f : A \to B$ such that $f(R_i^{\mathbf{A}}) \subseteq R_i^{\mathbf{B}}$ for every $1 \leq i \leq n$. We write $\mathbf{A} \to \mathbf{B}$ to indicate that there exists a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. A structure $\mathbf{B}$ is a *core* if the only homomorphisms from $\mathbf{B}$ to itself are onto. Given a structure $\mathbf{B}$, $\mathrm{CSP}(\mathbf{B})$ denotes the class of all similar structures $\mathbf{A}$ such that $\mathbf{A} \to \mathbf{B}$; $\neg\mathrm{CSP}(\mathbf{B})$ denotes the class of all $\mathbf{A}$ such that $\mathbf{A} \not\to \mathbf{B}$. It is easy to see that for every finite structure $\mathbf{B}$, there exists a core $\mathbf{B}'$ such that $\mathrm{CSP}(\mathbf{B}) = \mathrm{CSP}(\mathbf{B}')$.

For basic facts concerning Datalog and its relevant fragments we refer the reader to [8, 9, 11, 12]). Fix a vocabulary τ. A *Datalog program* is a finite set of rules of the form

$$T_0 : -\, T_1, \dots, T_n$$

where each T_i is an atomic formula $R(x_{i_1}, \dots, x_{i_k})$. Then T_0 is called the *head* of the rule, and the sequence $T_1, \dots, T_n$ the *body* of the rule. There are two kinds of relational predicates occurring in the program: predicates R that occur at least once in the head of a rule are called *intensional database predicates* (IDBs) and are not part of τ. The other predicates which occur only in the body of a rule are called *extensional database predicates* and must all lie in τ. One special IDB, which is 0-ary, is the *goal predicate* of the program. Each Datalog program is a recursive specification of the IDBs, with semantics obtained via least fixed-points of monotone operators. The goal predicate is initially set to `false`, and the Datalog program *accepts* a τ-structure $\mathbf{A}$ if its goal predicate evaluates to `true` on $\mathbf{A}$.

For $0 \leq j \leq k$, a (j, k)-*Datalog program* is a Datalog program with at most j variables in the head and at most k variables per rule. A Datalog program is *linear* if every of its rules has at most one occurrence of an IDB in its body. Given a rule t of the form

$$I :- J, T_1, \ldots, T_n$$

of a linear Datalog program where I and J are IDB's, its *symmetric* complement $t_{\rm sym}$ is the rule

$$J :- I, T_1, \ldots, T_n;$$

if t has no IDB in the body then we let $t_{\rm sym} = t$. A linear program is said to be *symmetric* if it contains the symmetric complement of each of its rules. A class $\mathcal{C}$ of structures is said to be *definable in (linear, symmetric)* (j,k)*-Datalog* if there is a (linear, symmetric) (j,k)-Datalog program which accepts precisely the structures from $\mathcal{C}$.

We shall require the notion of a *canonical* (linear, symmetric) (j,k)-Datalog program for $\mathbf{B}$. Let $\tau = \{R_1, \ldots, R_n\}$. For each r-ary relation S on B with $1 \le r \le j$, introduce an r-ary IDB I_S. Then the canonical (j,k)-Datalog program for $\mathbf{B}$ involves the IDBs I_S and EDBs $R_1, \ldots, R_n$, and contains all the rules with at most k variables with the following property: if every I_S in the rule is replaced by S and every R_s by $R_s^{\mathbf{B}}$, then every assignment of elements of B to the variables that satisfies the conjunction of atomic formulas in the body must also satisfy the atomic formula in the head. Finally, introduce one 0-ary IDB G together with the rule $G :- I_\emptyset(x_1, \ldots, x_j)$, and make G the goal predicate of the program. The *canonical linear* (j,k)-Datalog program for $\mathbf{B}$ consists of all linear rules from the canonical program described above. Finally, define the *canonical symmetric* (j,k)-Datalog program for $\mathbf{B}$ as the largest set of rules from the canonical linear program which is closed under symmetry. It was shown in [12] that if $\neg\mathrm{CSP}(\mathbf{B})$ is definable in (j,k)-Datalog then it is defined by the canonical (j,k)-Datalog program. The following result was first stated for plain Datalog in [12]:

Lemma 3 *Suppose that* $\neg\mathrm{CSP}(\mathbf{B})$ *is definable in (linear, symmetric)* (j,k)*-Datalog. Then* $\neg\mathrm{CSP}(\mathbf{B})$ *is precisely the set of structures accepted by the canonical (linear, symmetric)* (j,k)*-Datalog program for* $\mathbf{B}$.

We do not include the proof here as it is a direct adaptation of that of [12]. Although we shall not make use of Lemma 3 in our proofs we will need a simple argument that is used in its proof which we now state and prove. For every IDB I of any program $\mathcal{P}$, let $I(\mathbf{C})$ be the relation on C produced by the program at the end of its run on $\mathbf{C}$.

Lemma 4 *Suppose that* $\mathbf{A} \to \mathbf{B}$*. Then for every* $0 \le j \le k$*, the canonical (and hence the linear, symmetric)* (j,k)*-Datalog program for* $\mathbf{B}$ *does not accept* $\mathbf{A}$.

Proof: Since the body of every rule is a primitive positive formula, it is easy to verify that if f is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$, then $f(I(\mathbf{A})) \subseteq I(\mathbf{B})$ for every IDB I of $\mathcal{P}$. In particular, we have that $f(I_\emptyset(\mathbf{A})) \subseteq I_\emptyset(\mathbf{B}) = \emptyset$ hence $I_\emptyset(\mathbf{A}) = \emptyset$ which shows that $\mathcal{P}$ does not accept $\mathbf{A}$. ∎

Finally we shall use the following result from [11]:

Lemma 5 *Suppose that* $\neg\mathrm{CSP}(\mathbf{B})$ *is definable in symmetric* (j,k)*-Datalog. Then* $\mathrm{CSP}(\mathbf{B})$ *is in Logspace.*

3 Algebraic Preliminaries

For basic algebraic results we refer the reader to [18]. A *finitary operation* f on a set A is a map $f : A^n \to A$ for some $n \ge 1$, called the *arity* of f. An *algebra* is a pair $\mathbb{A} = \langle A; F\rangle$ where A, the *universe* of $\mathbb{A}$, is a non-empty set and F is a set of finitary operations on A, called the *basic* or *fundamental* operations of $\mathbb{A}$.

Let θ be an r-ary relation on A and let f be an n-ary operation on A. We say that f *preserves* θ, (θ is *invariant* under f) if, given n tuples $t_1, \ldots, t_n$ from R, applying f to the rows of the matrix whose columns are the t_i yields a tuple in θ. The set of relations invariant under all operations in F is denoted by $Inv(F)$, and the set of operations preserving all relations in Γ is denoted by $Pol(\Gamma)$.

The algebra associated to a relational structure $\mathbf{B}$ is $\mathbb{A}(\mathbf{B}) = \langle A; Pol(\Gamma)\rangle$ where A and Γ are the universe and the set of basic relations of $\mathbf{B}$ respectively. If a signature is specified, then one may define subalgebras, homomorphic images and products of algebras. A *variety* is a class of similar algebras closed under these three constructions. The variety $\mathcal{V}(\mathbb{A})$ *generated* by an algebra $\mathbb{A}$ is the smallest variety containing $\mathbb{A}$.

A ternary operation p ia a *majority* operation if it satisfies the identities $p(x,x,y) = p(x,y,x) = p(y,x,x) = x$ for all x,y; a ternary operation M is a *Maltsev* operation if it satisfies the identities $M(y,x,x) = y = M(x,x,y)$ for all x,y.

Recall from the introduction that a variety is *arithmetical* if it is congruence-permutable and congruence-distributive. It will be more convenient for us to use the following characterisation of arithmetical varieties:

Lemma 6 *[19] A variety is arithmetical if and only if it has a majority term and a Maltsev term.*

We now state some of the basic facts about arithmetical varieties we will require in the proof.

Lemma 7 *Let $\mathbf{\Gamma}$ be a core structure whose basic relations are invariant under a majority operation. Then there exists a core relational structure $\mathbf{B}$ (on the same universe as $\mathbf{\Gamma}$) such that*

1. *the basic relations of $\mathbf{B}$ are at most binary;*
2. $\mathbb{A}(\mathbf{B}) = \mathbb{A}(\mathbf{\Gamma})$*;*
3. *if $\neg\mathrm{CSP}(\mathbf{B})$ is in (linear, symmetric) Datalog then so is $\neg\mathrm{CSP}(\mathbf{\Gamma})$.*

Proof: Let G_0 denote the set of basic relations of $\mathbf{\Gamma}$. Let G consist of all the relations of arity at most 2 in $Inv(Pol(G_0))$. Because the relations in G_0 are invariant under a majority operation, it follows that $Pol(G_0) = Pol(G)$ [1]. In particular, the structure $\mathbf{B}$ with basic relations G is a core if $\mathbf{\Gamma}$ is, and $\mathbb{A}(\mathbf{\Gamma}) = \mathbb{A}(\mathbf{B})$. It then follows from Theorem 2.1 of [15] that $\neg\mathrm{CSP}(\mathbf{B})$ is in (linear, symmetric) Datalog if $\neg\mathrm{CSP}(\mathbf{B})$ is. ∎

A binary relation θ is *rectangular* if it satisfies the following: if (a,b), (c,b) and (c,d) are all in θ then so is (a,d). The following is well-known (and easy to verify):

Lemma 8 *If a binary relation is invariant under a Maltsev operation then it is rectangular.*

Lemma 9 *Let $\mathbf{B}$ be a core structure such that $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is congruence-permutable and $\neg\mathrm{CSP}(\mathbf{B})$ is in Datalog. Then $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is arithmetical.*

Proof: Since $\neg\mathrm{CSP}(\mathbf{B})$ is in Datalog, it follows from Theorem 4.2 of [16] that $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ omits types 1 and 2, and since this variety is congruence-permutable, it also omits types 4 and 5 by Theorem 9.14 of [13]. Congruence-permutability easily implies congruence-modularity, hence it follows from Theorems 8.5 and 8.6 of [13] that the variety is actually congruence-distributive, and thus arithmetical. ∎

4 Proof of the Main Results

We are now ready to launch into the proof of Theorem 1. Notice that Corollary 2 follows immediately from Theorem 1 and Lemma 9. Let $\mathbf{B}$ be a core relational structure such that $\mathcal{V}(\mathbb{A}(\mathbf{B}))$ is arithmetical. By Lemma 6 it follows that $\mathbb{A}(\mathbf{B})$ has both a majority and a Maltsev term operation, and hence the basic relations of $\mathbf{B}$ are invariant these operations. We may assume by Lemma 7 that the basic relations of $\mathbf{B}$ are at most binary. By Lemma 5 it will suffice to prove that $\neg\mathrm{CSP}(\mathbf{B})$ is in symmetric Datalog.

4.1 Some facts about canonical symmetric Datalog programs

For any structure $\mathbf{B}$, and for all $k > 1$ we denote the canonical symmetric $(k-1,k)$-Datalog program for $\mathbf{B}$ by $\mathcal{P}^k_{\mathbf{B}}$ (or simply by $\mathcal{P}^k$ if $\mathbf{B}$ is clear from the context) and call it simply the *canonical symmetric k-Datalog program* for $\mathbf{B}$.

Let $\mathbf{A}$ be any structure similar to $\mathbf{B}$. If $a_1,\dots,a_s \in A$ and I is an IDB of $\mathcal{P}^k_{\mathbf{B}}$, we say that $\mathcal{P}^k_{\mathbf{B}}$ *derives* $I(a_1,\dots,a_s)$ *on* $\mathbf{A}$ if $(a_1,\dots,a_s)$ is in the relation defined by I at the end of the run of the program $\mathcal{P}^k_{\mathbf{B}}$ on $\mathbf{A}$.

Every canonical Datalog program considered in the rest of the section will be for $\mathbf{B}$ and will be run on input $\mathbf{A}$.

Let R be a relation on B of arity r and let t be a rule of $\mathcal{P}^k$. Then the *multiplication* $R\cdot t$ is a rule defined in the following way: let $y_1,\dots,y_r$ be variables that do not occur in t.

- If the head of t is $I_S(x_1,\dots,x_s)$ then the head of $R\cdot t$ is defined to be $I_{R\times S}(y_1,\dots,y_r,x_1,\dots,x_s)$
- The body of t' is defined differently depending on whether the body of t has or not an IDB:
  - if the body of t does not have any IDB then the body of $R\cdot t$ is obtained by adding $I_R(y_1,\dots,y_r)$ to that of t.
  - if the body of t contains some IDB, say $I_U(z_1,\dots,z_u)$, then the body of $R\cdot t$ is obtained by replacing it (in the body of t) by the IDB $I_{R\times U}(y_1,\dots,y_r,z_1,\dots,z_u)$

The following fact about the multiplication is immediate:

Lemma 10 *If R is a relation of arity r and t is a rule of $\mathcal{P}^k$ then $R\cdot t$ is a rule of $\mathcal{P}^{k+r}$.*

Lemma 11 *Let $k,r>0$:*

1. *If $\mathcal{P}^{k+r}$ derives $I_R(a_1,\dots,a_r)$ and $\mathcal{P}^k$ derives $I_S(a'_1,\dots,a'_s)$ then $\mathcal{P}^{k+r}$ derives $I_{R\times S}(a_1,\dots,a_r,a'_1,\dots,a'_s)$*
2. *If the Datalog program $\mathcal{P}^{k+r}$ derives $I_{R\times S}(a_1,\dots,a_r,a'_1,\dots,a'_s)$ and $\mathcal{P}^k$ derives $I_S(a'_1,\dots,a'_s)$, then $\mathcal{P}^{k+r}$ derives $I_R(a_1,\dots,a_r)$.*

Proof: (1) For each rule t in the derivation of $I_S(a'_1,\dots,a'_s)$ in $\mathcal{P}^k$, the rule $R\cdot t$ is in $\mathcal{P}^{k+r}$ by Lemma 10: the sequence of rules thus obtained, preceded by the derivation of $I_R(a_1,\dots,a_r)$ in $\mathcal{P}^{k+r}$, is a derivation of $I_{R\times S}(a_1,\dots,a_r,a'_1,\dots,a'_s)$.

(2) Let $t_1,\dots,t_m$ be the sequence of rules in the derivation of $I_S(a'_1,\dots,a'_s)$ in $\mathcal{P}^k$, and consider the sequence

$$(R\cdot t_m)_{\mathrm{sym}},\dots,(R\cdot t_1)_{\mathrm{sym}}.$$

Using Lemma 10 again, this sequence, preceded by the derivation of $I_{R\times S}(a_1,\dots,a_r,a'_1,\dots,a'_s)$ in $\mathcal{P}^{k+r}$, is a derivation of $I_R(a_1,\dots,a_r)$. ∎

4.2 Two lemmas

A *path* P on a given structure $\mathbf{C}$ is any sequence $c_1,\dots,c_t$ of (possibly repeated) elements of C. The path P is a *cycle* if $c_1=c_t$.

Let $P=a_1,\dots,a_t$, and $Q=b_1,\dots,b_t$ be paths of the same length on structures $\mathbf{A}$ and $\mathbf{B}$, respectively. We will denote the mapping from $\{a,a'\}$ to $\{b,b'\}$ taking a to b and a' to b' by $a,a'\to b,b'$. If $t>1$, we say that Q *supports* (or 0-supports) P if, for all $1\le i<t$, the mapping $a_i,a_{i+1}\to b_i,b_{i+1}$ is a partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$ (i.e, a homomorphism from the substructure of $\mathbf{A}$ induced by $\{a_i,a_{i+1}\}$ to $\mathbf{B}$); for $t=1$, Q supports P if the mapping $a_1\to b_1$ is a partial homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Observe that several occurrences of the same element in the path on $\mathbf{A}$ need not be mapped to the same value in $\mathbf{B}$.

For every element a in A and for every $n\ge 0$ we define a subset a^n of B inductively. First,

a^0 is B for all $a \in A$. If $n > 0$ then a^n is defined using a^{n-1} in the following way: Let $P = a_1, \dots, a_t$ and $Q = b_1, \dots, b_t$ be paths on $\mathbf{A}$ and $\mathbf{B}$. We say that Q $(n-1)$-*supports* P if Q supports P and for every $1 \leq i \leq t$, $b_i \in a_i^{n-1}$. Let b be an element of B, then $b \in a^n$ if every cycle P on $\mathbf{A}$ starting in a is $(n-1)$-supported by some cycle Q on $\mathbf{B}$ starting in b.

Although not explicitly stated there, the following result is implicit in [10].

Lemma 12 *[10] If all relations of $\mathbf{B}$ are binary and invariant under a majority operation, and for every $a \in A$, $a^{M+1} \neq \emptyset$, where M is the cardinality of B, then $\mathbf{A} \rightarrow \mathbf{B}$.*

Let $P = a_1, \dots, a_n$ be a path on $\mathbf{A}$ and let $N \geq 0$. We define R_P^N to be the binary relation on B that consists of all (b, b') such that there exists some path $Q = b_1, \dots, b_n$ in B that N-supports P with $b_1 = b$ and $b_n = b'$.

Our main result will follow from Lemma 12 if we can show this:

Lemma 13 *For every a and every $0 \leq N$, (1) $\mathcal{P}_{\mathbf{B}}^{2+4N}$ derives $I_{a^N}(a)$ on $\mathbf{A}$. Furthermore (2) for every path $P = a_1, \dots, a_n$ on $\mathbf{A}$, $\mathcal{P}_{\mathbf{B}}^{5+4N}$ derives $I_{R_P^N}(a_1, a_n)$ on $\mathbf{A}$.*

Indeed, if $N = M+1$ where $M = |B|$ then can prove that $\mathcal{P}_{\mathbf{B}}^{2+4N}$ defines $\neg\mathrm{CSP}(\mathbf{B})$. By Lemma 4 it is only necessary to show that if $\mathbf{A}$ is a structure not accepted by $\mathcal{P}_{\mathbf{B}}^{2+4N}$ then $\mathbf{A} \rightarrow \mathbf{B}$. This is so, because since $\mathcal{P}_{\mathbf{B}}^{2+4N}$ derives $I_{a^N}(a)$, if a^N were empty then the program would contain the rule $G :- I_{a^N}(x)$ and hence it would accept $\mathbf{A}$, a contradiction. Thus $a^N \neq \emptyset$ for every $a \in A$ and we are done by Lemma 12.

4.3 Proof of Lemma 13

Proof:

Let us first show that (1) implies (2).

In what follows K is set to $2 + 4N$. Associated to P we define a collection of paths, that we call the *symmetric paths of* P. Every such path is of the form $y_1, \dots, y_m$ where $y_1 = a_1$ and for every $2 \leq i \leq m$, if $y_{i-1} = a_j$ then $y_i \in \{a_{j-1}, a_{j+1}\}$.

For every $1 \leq i \leq n$, we define the binary relation R_i^N on B as the union of all $R_{P^*}^N$ where P^* ranges over all symmetric paths of P ending in a_i.

Item (2) follows immediately from the following two claims:

Claim 1 *For every $2 \leq i \leq n$, $\mathcal{P}^{K+3}$ derives $I_{R_i}^N(a_1, a_i)$.*

Claim 2 $R_n^N = R_P^N$.

Proof: (of Claim 1)

We shall need the following construction. Let $\mathbf{C}$ be a structure, let c_1, c_2 be elements of C and let x_1, x_2 be new elements that we shall regard as variables (of a Datalog program). Then $\mathbf{C}_{c_1,c_2 \rightarrow x_1,x_2}$ is the collection of atomic predicates of the form $\theta(x_{i_1}, \dots, x_{i_r})$ with $i_1, \dots, i_r \in \{1, 2\}$ where θ is in the vocabulary of $\mathbf{C}$ and $(c_{i_1}, \dots, c_{i_r}) \in \theta^{\mathbf{C}}$.

We shall prove the claim by induction on i.

Case $(\mathbf{i = 2})$

Since by (1) $\mathcal{P}^K$ derives $I_{a_1{}^N}(a_1)$ and $I_{a_2{}^N}(a_2)$ on input $\mathbf{A}$, by Lemma 11 (1), $\mathcal{P}^{K+1}$ derives $I_{a_1^N \times a_2^N}(a_1, a_2)$. Consider now the rule t with head $I_{R_2^N}(x_1, x_2)$ and whose body consists of $I_{a_1^N \times a_2^N}(x_1, x_2)$ in addition to all predicates in $\mathbf{A}_{a_1,a_2 \rightarrow x_1,x_2}$. Since rule t allows one to derive $I_{R_2^N}(a_1, a_2)$ it only remains to show that t is a rule of $\mathcal{P}^3$.

We shall start by showing that t is a rule of the canonical (not necessarily symmetric) 3-Datalog program. Let $b_1, b_2 \in B$ be such that the assignment $x_1, x_2 \rightarrow b_1, b_2$ satisfies the body of t. Since the body of t contains all predicates in $\mathbf{A}_{a_1,a_2 \rightarrow x_1,x_2}$ we can conclude that $a_1, a_2 \rightarrow b_1, b_2$ is a partial homomorphism. Since the body also contains the IDB $I_{a_1^N \times a_2^N}(x_1, x_2)$ we conclude that $b_1 \in a_1^N$ and $b_2 \in a_2^N$. Hence b_1, b_2 N-supports a_1, a_2 and hence $(b_1, b_2) \in R_2^N$.

Now let us see that the symmetric complement t_{sym} of t belongs also to the canonical 3-Datalog program. Let $b_1, b_2 \in B$ such that the assignment $x_1, x_2 \rightarrow b_1, b_2$ satisfies the body of $t_{\rm sym}$. Since it contains $I_{R_2^N(x_1,x_2)}$ and, by definition, $R_2^N \subseteq a_1^N \times a_2^N$, we can conclude that $(b_1, b_2) \in a_1^N \times a_2^N$.

Case ($\mathbf{i-1 \Rightarrow i}$)

By (1) and Lemma 11 (1), $\mathcal{P}^{K+1}$ derives $I_{a_{i-1}^N \times a_i^N}(a_{i-1}, a_i)$. Hence, again by Lemma 11 (1), $I_{R_{i-1}^N \times a_{i-1}^N \times a_i^N}(a_1, a_{i-1}, a_{i-1}, a_i)$ is derived by $\mathcal{P}^{K+3}$. Now consider the rule t with head $I_{R_i^N \times a_{i-1}^N \times a_i^N}(x_1, x_i, x_{i-1}, x_i)$ and whose body consists of $I_{R_{i-1}^N \times a_{i-1}^N \times a_i^N}(x_1, x_{i-1}, x_{i-1}, x_i)$ in addition to all predicates in $\mathbf{A}_{a_{i-1}, a_i \rightarrow x_{i-1}, x_i}$.

We shall prove that t is in $\mathcal{P}^{K+3}$. In fact we shall show that t is a rule of the canonical (not necessarily symmetric) 5-Datalog program.

Let b_1, b_{i-1}, b_i be elements of B such that the assignment $x_1, x_{i-1}, x_i \rightarrow b_1, b_{i-1}, b_i$ satisfies the body of t. Hence $(b_1, b_{i-1}, b_{i-1}, b_i)$ is in $R_{i-1}^N \times a_{i-1}^N \times a_i^N$ which implies that $(b_1, b_{i-1}) \in R_{i-1}^N$, $b_{i-1} \in a_{i-1}^N$, and $b_i \in a_i^N$. Since $(b_1, b_{i-1}) \in R_{i-1}^N$, there exists some symmetric path P^* associated to P that ends at a_{i-1} such that $(b_1, b_{i-1}) \in R_{P*}^N$. Hence there exists a path Q^* that N-supports P^* that starts at b_1 and ends at b_{i-1}.

Consider now the paths P' and Q' obtained by adding respectively a_i and b_i at the end of P^* and Q^* respectively. We want to see that Q' N-supports P'. To this end it is only necessary to verify that $a_{i-1}, a_i \rightarrow b_{i-1}, b_i$ is a partial homomorphism, which follows from the fact that the body of t contains all predicates in $\mathbf{A}_{a_{i-1}, a_i \rightarrow x_{i-1}, x_i}$. Consequently, since Q' N-supports P' then (b_1, b_i) belongs to R_i^N. Hence $(b_1, b_i, b_{i-1}, b_i) \in R_i^N \times a_{i-1}^N \times a_i^N$ and we are done.

To complete the proof one may show in a similar vein that the symmetric complement $t_{\rm sym}$ of t is also in the canonical 5-Datalog program.

Hence, $\mathcal{P}^{K+3}$ can derive, using rule t, $I_{R_i^N \times a_{i-1}^N \times a_i^N}(a_1, a_i, a_{i-1}, a_i)$. Finally, by Lemma 11(2) with $r = 2$ and $j = K + 1$, $\mathcal{P}^{K+3}$ derives $I_{R_i^N}(a_1, a_i)$.

∎

Proof: (of Claim 2)

Since the set of symmetric paths associated to P contains P itself we conclude that $R_P^N \subseteq R_n^N$. Every symmetric path $P^* = y_1, \dots, y_m$ associated to P can be regarded as a sequence of *segments* in which the indices are either increasing or decreasing. Formally, a *segment* of P^* is a maximal subpath $a_r = y_i, y_{i+1}, \dots, y_j$ of P^* such that, for every $i \leq l \leq j$, $y_l = a_{l-i+r}$ (an *increasing segment*) or for every $i \leq l \leq j$, $y_l = a_{i-l+r}$ (a *decreasing segment*).

We shall prove that $R_n^N \subseteq R_P^N$ by contradiction. Suppose there is a tuple (b, b') in R_n^N not in R_P^N; let P^* be a symmetric path (associated to P) ending in a_n with minimum number of segments such that $(b, b') \in R_{P^*}^N$. Observe that the number of segments of a symmetric path is necessarily odd, and since $P^* \neq P$ this number is at least 3. For convenience let the number of segments of P^* be $k + 1$ and let the segments of P^* be numbered $0, 1, \dots, k$; in particular increasing segments are even numbered and decreasing segments are odd. Let l be any integer in $0 \leq l \leq k - 2$, let a_j be the last element of the l-th segment and let $a_{j'}$ be the last element of the $(l + 2)$-th segment. We claim that $j' < j$ if l is even and $j' > j$ if l is odd. Notice that the claim yields an immediate contradiction: we can never reach a_n if this condition is satisfied and the number of segments is greater than 1.

Let us prove the claim by contradiction. Let l be the smallest index for which the claim is violated. We shall consider the case where l is even, the proof for l odd is similar.

Let P^* be $y_1, \dots, y_m$. Since $(b, b') \in R_{P^*}^N$ there exists a path $Q^* = b_1, \dots, b_m$ that N-supports P^* with $b = b_1$ and $b' = b_m$.

Let a_i be the first element of the $(l + 2)$-th segment (see figure 1). Since l is even $j > i$ and let U be the path $a_i, a_{i+1}, \dots, a_j$. We shall study R_U^N. First observe that R_U^N is obtained from relations of $\mathbf{B}$ by a sequence of constructions involving only cartesian products, projections and equality selection. Hence if $\mathbf{B}$ is invariant under a Maltsev operation φ then so is R_U^N (easy exercise). Let a_{i^*} be the first element of the l-th segment. By the minimality of l, we have that $i > i^*$. Hence there is y_{r_0} in the l-th segment (and only one) with $y_{r_0} = a_i$. Let $y_{r_1} = a_j$ be the last element of the l-th segment. Hence $(b_{r_0}, b_{r_1}) \in R_U^N$. Let $y_{r_2} = a_i$ be the last element of the $(l+1)$-th segment. Hence $(b_{r_2}, b_{r_1}) \in R_U^N$. Finally, if $a_{j'}$ is the last element of the $(l+2)$-th segment and by hypothesis $j' \geq j$ then there exists some element y_{r_3} in the $(l+2)$-th segment with $y_{r_3} = a_j$. Hence $(b_{r_2}, b_{r_3}) \in R_U^N$. By Lemma 8 the relation R_U^N is rectangular and hence (b_{r_0}, b_{r_3}) is also in R_U^N. Hence U is N-supported by a path $b'_i, \ldots, b'_j$ with $b'_i = b_{r_0}$ and $b'_j = b_{r_3}$. Consider now the symmetric path P' given by $y_1, \ldots, y_{r_0}, a_{i+1}, \ldots, a_{j-1}, y_{r_3}, \ldots, y_m$. Path P' is N-supported by $b_1, \ldots, b_{r_0}, b'_{i+1}, \ldots, b'_{j-1}, b_{r_3}, \ldots, b_m$. Hence $(b, b') \in R_{P'}^N$ and P' has only $k-2$ segments, a contradiction. This concludes the proof of Claim 2 and hence that (1) implies (2).

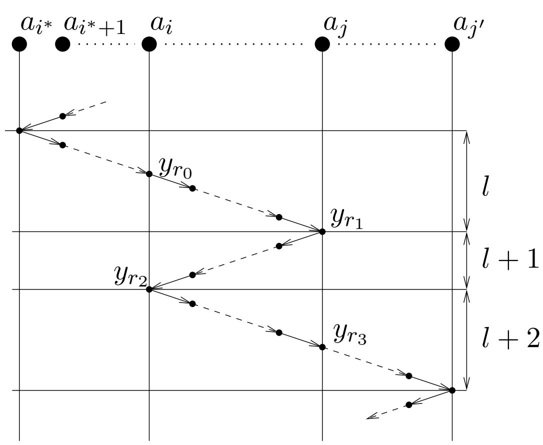

Figure 1.

∎

Finally we shall prove the Lemma by induction. Since, as we have just proved (1) implies (2), we only need to show (1).

The case $N = 0$ is trivial, since $a^0 = B$ and the canonical symmetric 2-Datalog program contains the rule

$$I_B(x) : \texttt{true} \ .$$

Now let us assume that the statement holds for $N-1$ (where $N \geq 1$). For every cycle C, let us define S_C as the subset of B that contains all those b such that $(b, b) \in R_C^{N-1}$. Clearly, a^N is the intersection of all S_C where C is a cycle that starts (and hence ends) at a. Furthermore, since B is finite, a^N can be obtained as a finite intersection of S_C's.

Let C be any cycle in $\mathbf{A}$ an let a its first element. By the inductive hypothesis, $I_{R_C^{N-1}}(a, a)$ is derived by $\mathcal{P}^{1+4N}$. Since $I_{S_C}(x) : I_{R_C}(x, x)$ is a valid symmetric rule, $\mathcal{P}^{1+4N}$ also derives $S_C(a)$.

Let $C_1, \ldots, C_m$ be a sequence of cycles starting at the same node a and let $F_m = \bigcap_{1 \leq i \leq m} S_{C_i}$. We shall show by induction on m that $I_{F_m}(a)$ is derived by $\mathcal{P}^{2+4N}$. The case $m = 1$ follows from the fact that $F_1 = S_{C_1}$. For the inductive case let us assume that $I_{F_{m-1}}(a)$ is derived by $\mathcal{P}^{2+4N}$. Since $I_{S_{C_m}}(a)$ is derived by $\mathcal{P}^{1+4N}$ we can conclude, from Lemma 11 (1), that $I_{F_{m-1} \times S_{C_m}}(a, a)$ is derived by $\mathcal{P}^{2+4N}$. By applying the symmetric complement $I_{F_m}(x) : I_{F_{m-1} \times S_{C_m}}(x, x)$ we derive $I_{F_m}(a)$.

∎

Acknowledgements

We wish to thank Marcos Romero for deep inspirations. We also thank Andrei Krokhin and Matt Valeriote for fruitful discussions.

References

[1] K. Baker, A. Pixley. Polynomial interpolation and the Chinese remainder theorem for algebraic systems. *Math. Z.* **143**, (1975) 165-174.

[2] A. Bulatov, P. Jeavons, and A. Krokhin. Classifying complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.

[3] A. Bulatov. A dichotomy theorem for constraints on a three-element set. *Proc. of 43rd Foundations of Comp. Sci. (FOCS'02)*, (2002), 649–658.

[4] A. Bulatov. Malt'sev constraints are tractable. Research Report RR-02-05, Oxford University Computing Laboratory, April 2002, 36 pages.

[5] A. Bulatov, V. Dalmau. A Simple Algorithm for Mal'tsev Constraints. *SIAM J. Comput.*, **36**, 1, (2006), 16-27.

[6] C. Carvalho, V. Dalmau, P. Marković, and M. Maróti. CD(4) has bounded width. *Algebra Universalis*. to appear.

[7] D. Cohen and P. Jeavons. The complexity of constraint languages. In *Handbook of Constraint Programming*, chapter 8. Elsevier, 2006.

[8] V. Dalmau. Linear Datalog and bounded path duality for relational structures. *Logical Methods in Computer Science*, 1(1), 2005. (electronic).

[9] V. Dalmau, P. G. Kolaitis, M. Y. Vardi. Constraint Satisfaction, Bounded Treewidth, and Finite-Variable Logics. *Principles and Practice of Constraint Programming (CP'02)*, (2002), 310-326.

[10] V. Dalmau and A. Krokhin. Majority constraints have bounded pathwidth duality. *European Journal of Combinatorics*, 2007. to appear.

[11] L. Egri, B. Larose, and P. Tesson. Symmetric Datalog and constraint satisfaction problems in Logspace. In *LICS'07*, pages 193–202, 2007.

[12] T. Feder and M.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, 1998.

[13] D. Hobby and R.N. McKenzie. *The Structure of Finite Algebras*, volume 76 of *Contemporary Mathematics*. AMS, Providence, R.I., 1988.

[14] P. Idziak, P. Markovic, R. McKenzie, M. Valeriote, and R. Willard. Tractability and learnability arising from algebras with few subpowers. In *LICS'07*, pages 213–222, 2007.

[15] B. Larose, P. Tesson. Universal Algebra and Hardness Results for Constraint Satisfaction Problems. (Long version of Icalp '07), 32 pages. Submitted.

[16] B. Larose and L. Zádori. Bounded width problems and algebras. *Algebra Universalis*, 56(3-4):439–466, 2007.

[17] L. Libkin. *Elements of Finite Model Theory*. Springer, 2004.

[18] R. McKenzie, G. McNulty, W. Taylor. *Algebras, Lattices and Varieties.* Wadsworth and Brooks/Cole, 1987.

[19] A.F. Pixley. Distributivity and permutability of congruence relations in equational classes of algebras. *Proc. Amer. Math. Soc.* **14**, 105-109.

[20] T. J. Schaefer. The complexity of satisfiability problems. In *Proc.* 10^{th} *ACM STOC*, pages 216–226, 1978.