Programming Research Group

AN ALGEBRAIC APPROACH TO MULTI-SORTED CONSTRAINTS

Andrei A. Bulatov
Department of Algebra and Discrete Mathematics
Ural State University, Ekaterinburg, Russia
e-mail: Andrei.Bulatov@usu.ru

Peter Jeavons
Computing Laboratory
University of Oxford, UK
e-mail: Peter.Jeavons@comlab.ox.ac.uk

PRG-RR-01-18

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD

Abstract

We describe a common framework for the Constraint Satisfaction Problem and the Conjunctive Query Evaluation Problem, encompassing a generalised form of these problems in which different variables may take values from different sets. The framework we develop allows us to specify natural subclasses of these two problems using algebraic techniques, and to establish when these subclasses are tractable. We show that a range of tractable classes can be obtained by combining recently identified tractable subclasses of the usual constraint satisfaction problem over a single set of values. We also systematically develop an algebraic structural theory for the general problem, which provides the prerequisites for further use of the powerful algebraic machinery.

Classification: computational complexity, algorithms, constraint satisfaction problems, conjunctive query, algebra.

1 Introduction

Two well established problems, the Conjunctive Query Evaluation problem (CQ-Evaluation) in database theory and the Constraint Satisfaction Problem (CSP) are, in fact, equivalent reformulations of the same problem: the problem of finding a homomorphism between two relational structures [23]. In CQ-Evaluation we are asked if a *conjunctive query* to a relational database, that is, a query of the form $\exists x_1 \ldots \exists x_k(\varrho_1 \wedge \ldots \wedge \varrho_n)$ where $\varrho_1, \ldots, \varrho_n$ are atomic formulas, has a solution [35]. In the Constraint Satisfaction Problem we are required to find an assignment of values to a given set of variables, subject to *constraints* on the values which can be assigned simultaneously to certain specified subsets of the variables [29, 25]. The CSP provides a natural framework for many combinatorial problems, including Satisfiability, Graph Colorability, and many problems of planning and scheduling (see [18]).

Both the constraint satisfaction problem and CQ-Evaluation are NP-hard in general ([29, 25, 4, 35]). However, certain restrictions on the form of the constraints or the structure of the CQ-Evaluation instance have been shown to ensure tractability [6, 8, 20, 21, 22, 29, 33, 34, 36, 5, 14, 15]. The fundamental research goal in this direction is to identify all tractable classes, and hence to identify the precise border between tractable and intractable cases.

There are two principal ways of restricting these problems (corresponding to restrictions on the source and the target relational structures in the associated homomorphism problem). In the first approach restrictions are imposed on the possible ways in which the constraints may interact, or, correspondingly, on the form of the conjunctive queries. The tractability of the restricted problem has been proved in cases where the hypergraph corresponding to a CSP instance or a query is acyclic [36], of bounded tree-width [5, 14], or of bounded hypertree-width [15].

In the second approach one restricts the form of the constraints, or correspondingly, the properties of the database tables. This approach has been very successfully applied to the constraint satisfaction problem, but has not been widely applied for conjunctive queries, although restrictions of this form may arise in practice from the nature of a database.

For the CSP a complete classification of complexity for restricted constraint types was obtained for the important special case of Boolean CSPs by Schaefer in 1978 [31]. Schaefer established that for the Boolean CSP (which he called the "Generalised Satisfiability Problem") there are exactly six different families of tractable constraints, and any problem involving constraints not contained in one of these six families is NP-complete. This important result is known as Schaefer's Dichotomy Theorem for Boolean relations.

The question of how this characterisation of tractable constraints could be generalized to domains with more than 2 elements was raised by Schaefer in the same paper [31]. This problem is still unsolved, although some progress has been made recently. In particular, Feder and Vardi [12] used techniques from logic programming and group theory to identify three broad families of tractable constraints which include all of Schaefer's six classes. Another fruitful approach was suggested by Jeavons and co-authors, who have shown that all tractable constraint classes over finite domains can be characterised using algebraic properties of relations [21, 20, 19]

There is a striking difference between theoretical studies of both the CSP and CQ-EVALUATION, and more applied work on these problems: in almost all theoretical studies constraint satisfaction problems are assumed to have the same domain for all variables, and databases are considered as relational structures on a single set, while in practice the different variables of a CSP often have different domains, and the different attributes of a database also generally have different domains. Usually, this gap is bridged by making use of certain tricks which reduce an arbitrary problem to a problem over a single fixed set. However, as we shall see below, these standard tricks may change the complexity of certain subproblems. We therefore need an approach which allows us to study constraint satisfaction problems and conjunctive queries whilst distinguishing types of variables or attributes of a database.

In this paper we present a framework which is based on algebraic invariance properties and further develops the approach introduced and developed in [21, 20, 19, 3]. This framework satisfies all the required needs: it provides a language in which many natural properties of constraints or databases can be expressed, and allows one to work correctly with problems including several different domains. Another advantage is that our approach makes it possible to use the powerful structural results developed for classifying the structure of finite algebras [28, 17]. Some rudimentary versions of this approach have already been successfully applied to obtain new tractable classes [1, 2].

In this paper we mainly use constraint terminology, but all of the results may easily be translated into the terminology of database theory (see Example 2.4).

2 The Constraint Satisfaction Problem

The central notion in the study of constraints and constraint satisfaction problems is the notion of a *relation*. In this paper we allow relations over an arbitrary collection of sets, defined as follows.

Definition 2.1 *For any collection of sets $\mathcal{A} = \{A_i \mid i \in I\}$, and any list of indices $(i_1, i_2, \ldots, i_m) \in I^m$, a subset ϱ of $A_{i_1} \times A_{i_2} \times \cdots \times A_{i_m}$, together with the list $(i_1, i_2, \ldots, i_m)$, will be called an [m-ary]* relation *over $\mathcal{A}$ with* signature $(i_1, i_2, \ldots, i_m)$. *For any such relation ϱ, the signature of ϱ will be denoted $\sigma(\varrho)$.*

The ith component of a tuple $\overline{a}$ will be denoted by $\overline{a}[i]$.

Example 2.2 Let $\mathcal{A} = \{A_1, A_2\}$, where $A_1 = \{0, 1, 2\}$, $A_2 = \{a, b, c\}$. Consider the following relation over $\mathcal{A}$ with arity 5:

$$\varrho = \begin{pmatrix} 0 & 1 & 2 & 0 & 0 & 1 & 1 & 2 & 2 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 2 \\ 1 & 2 & 0 & 2 & 0 & 0 & 1 & 1 & 2 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 & 0 \\ 2 & 0 & 1 & 2 & 2 & 0 & 0 & 1 & 1 & 2 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ c & c & c & c & c & c & c & c & c & c & c & c & b & b & b & b & a & c \\ c & c & c & c & c & c & c & c & c & b & b & b & c & c & c & b & c & a \end{pmatrix}$$

Note that the tuples of ϱ are written vertically, and ϱ has signature $(1, 1, 1, 2, 2)$. □

The 'constraint satisfaction problem' was introduced by Montanari in 1974 [29] and has been widely studied [10, 12, 24, 25, 26, 27, 33, 32]

Definition 2.3 *The* constraint satisfaction problem (CSP) *is the combinatorial decision problem with*

INSTANCE: *a quadruple $(V; \mathcal{A}; \delta; \mathcal{C})$ where*

- *V is a set of* variables*;*
- *$\mathcal{A} = \{A_i \mid i \in I\}$ is a collection of* sets of values [domains]*;*
- *δ is a mapping from V to I, called the* domain function*;*
- *$\mathcal{C}$ is a set of* constraints.
- *Each constraint $C \in \mathcal{C}$ is a pair $\langle s, \varrho \rangle$, where*
  - *$s = (v_1, \ldots, v_{m_C})$ is a tuple of variables of length m_C, called the* constraint scope*;*
  - *ϱ is an m_C-ary relation over $\mathcal{A}$ with signature $(\delta(v_1), \ldots, \delta(v_{m_C}))$, called the* constraint relation.

QUESTION: *does there exist a* solution, *i.e., a function φ, from V to $\bigcup_{A \in \mathcal{A}} A$, such that, for each variable $v \in V$, $\varphi(v) \in A_{\delta(v)}$, and for each constraint $\langle s, \varrho \rangle \in \mathcal{C}$, with $s = (v_1, \ldots, v_m)$, the tuple $(\varphi(v_1), \ldots, \varphi(v_m))$ belongs to ϱ?*

Example 2.4 Constraint satisfaction problems in which each variable has a distinct set of possible values frequently arise in the study of databases, although a rather different vocabulary is normally used, as in the following definition.

Definition 2.5 *A* relational database *is a finite collection of* tables. *A table consists of a* scheme *and an* instance*:*

A scheme is a finite set of attributes, *where each attribute has an associated set of possible values, referred to as a* domain.

An instance is a finite set of rows, *where each row is a mapping that associates with each attribute of the scheme a value in its domain.*

As is easily seen, an instance of a constraint satisfaction problem corresponds to a relational database by a simple translation of terms: 'attributes' have to be replaced with 'variables', 'tables' with 'constraints', 'domains' with 'sets of values', 'scheme' with 'constraint scope', 'instance' with 'constraint relation', and 'rows' with 'tuples'.

Hence a conjunctive query $\exists x_1 \ldots \exists x_k(\varrho_1 \wedge \ldots \wedge \varrho_n)$ is equivalent to the CSP instance whose variables are the variables of the query (both free and bound). For each atomic formula ϱ_i, there is a constraint C such that the scope of C is the list of variables of ϱ_i and the constraint relation of C is the set of models of ϱ_i. $\Box$

We will also use a restricted version of the constraint satisfaction problem in which all the variables have the same domain. In this case a problem instance may be specified by a triple $(V; D; \mathcal{C})$ where D is the common domain. This special case will be referred to below as the *one-sorted* CSP, whereas the more general form will sometimes be referred to as the *multi-sorted* CSP. (We will use these long names only in those cases when it is not clear from the context which version we mean.)

Example 2.6 An instance of GRAPH q-COLORABILITY consists of a graph G. The question is whether the vertices of G can be labelled with q colours so that adjacent vertices are assigned different colours.

This problem corresponds to the CSP in which the variables are the vertices of G, and for each edge $\{v, w\}$, there is a constraint $(\{v, w\}, \neq_A)$ where A is a q-element set (of colours) and

$$\neq_A \; = \{(a, b) \in A^2 \mid a \neq b\}$$

is the binary disequality relation on A. $\Box$

Many other examples of well-known problems expressed as CSPs can be found in [18].

The general constraint satisfaction problem is known to be NP-hard (this follows immediately from Example 2.6). However, certain restrictions may affect the complexity of the CSP. One of the possible natural ways to restrict the CSP is to limit the relations which can appear in constraints.

Definition 2.7 *For a given set of relations Γ over a collection of sets $\mathcal{A}$,* CSP(Γ) *is defined to be the decision problem with*

INSTANCE: *An instance, $\mathcal{P} = (V; \mathcal{B}; \delta; \mathcal{C})$, of the multi-sorted constraint satisfaction problem, in which*

- $\mathcal{B} \subseteq \mathcal{A}$;
- *for each constraint $\langle s, \varrho\rangle \in \mathcal{C}$, the relation ϱ is an element of Γ.*

QUESTION: *Does $\mathcal{P}$ have a solution?*

Example 2.8 The problem GRAPH q-COLORABILITY corresponds to CSP($\{\neq_A\}$). □

For databases the set Γ is the set of possible tables. Hence, in many cases of interest Γ may be very large, or even infinite, although for any fixed scheme over finite domains it will be finite.

We shall be concerned with distinguishing between those sets of relations which give rise to *tractable* problems (i.e., problems for which there exists a polynomial-time solution algorithm) and those which do not. We therefore make the following definition.

Definition 2.9 *A set of relations, Γ is said to be* tractable, *if* CSP(Γ') *is tractable for each finite subset $\Gamma' \subseteq \Gamma$.*

A set of relations, Γ is said to be NP-complete, *if* CSP(Γ') *is NP-complete for some finite subset $\Gamma' \subseteq \Gamma$.*

Example 2.10 Note that for any set A with 3 or more elements, the set Γ containing only the binary disequality relation $\neq_A$ defined in Example 2.6 is NP-complete. This is because in this case CSP(Γ) corresponds to the $|A|$-colorability problem, which is well-known to be NP-complete when $|A| \geq 3$ [13].

On the other hand, when A is a set with 2 elements, then the set Γ containing only the binary disequality relation $\neq_A$ is tractable, because in this case CSP(Γ) corresponds to the 2-colorability problem, which is well-known to be tractable [13] □

In general, the complexity of a set of multi-sorted relations is not determined by the complexity of the restrictions of those relations onto each individual domain, as the next example shows.

Example 2.11 Consider the sets $A_1 = \{0, 1\}$ and $A_2 = \{a, b, c\}$, and the relations

$$\varrho_1 = \begin{pmatrix} 1 & 0 & 0 \\ a & b & c \end{pmatrix}, \qquad \varrho_2 = \begin{pmatrix} 0 & 1 & 0 \\ a & b & c \end{pmatrix}, \qquad \varrho_3 = \begin{pmatrix} 0 & 0 & 1 \\ a & b & c \end{pmatrix}$$

over $\{A_1, A_2\}$, each with signature $(1, 2)$.

The set of relations $\Gamma = \{\varrho_1, \varrho_2, \varrho_3\}$ is NP-complete. To see this, we notice that the problem ONE-IN-THREE, which is known to be NP-complete [13], can be reduced to CSP(Γ). ONE-IN-THREE is equivalent to CSP($\{\varrho\}$) where

$$\varrho = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

and a reduction can be constructed as follows. For a problem instance $\mathcal{P} = (V; \{0, 1\}; \{C_1, \ldots, C_q\}) \in$ CSP($\{\varrho\}$), we choose q auxiliary variables $v_1, \ldots, v_q$ distinct from the variables in V, and set

$$\mathcal{P}' = (V \cup \{v_1, \ldots, v_q\}; \{A_1, A_2\}; \delta; \{C_1^1, C_1^2, C_1^3, C_2^1, \ldots, C_q^3\})$$

where $\delta(v) = 1$ if $v \in V$, $\delta(v) = 2$ otherwise, and for each $C_i = \langle (w_1, w_2, w_3), \varrho \rangle$, the new constraints are $C_i^1 = \langle (w_1, v_i), \varrho_1 \rangle$, $C_i^2 = \langle (w_2, v_i), \varrho_2 \rangle$, $C_i^3 = \langle (w_3, v_i), \varrho_3 \rangle$. It is easy to check that $\mathcal{P}$ and $\mathcal{P}'$ are equivalent.

However, if we restrict each relation in Γ to the individual domains A_1 and A_2, then we obtain just the unary relations $\{0, 1\}$ and $\{a, b, c\}$ which, obviously, are both tractable. □

The next example indicates that tractability may be due simply to the signatures of a set of relations rather than the tuples they contain.

Example 2.12 Let A_1 and A_2 be two distinct supersets of a set A, and let Γ be the set containing a single binary disequality relation $\neq_A$, considered as a relation over $\{A_1, A_2\}$ with signature $(1, 2)$.

Because of the signature, this constraint can only be imposed between two variables when one has domain A_1 and the other has domain A_2. Hence, in this case CSP(Γ) corresponds to the problem of colouring a bipartite graph with $|A|$ colours, which is clearly tractable. □

It is sometimes desirable to convert a multi-sorted constraint satisfaction problem into a one-sorted problem. The most straightforward way to do this for a given multi-sorted problem instance $(V; \mathcal{A}; \delta; \mathcal{C})$, is to take $B = \bigcup_{A \in \mathcal{A}} A$, and replace each constraint relation by a relation over B containing the same tuples.

However, applying this procedure to the disequality relation in Example 2.12 gives the usual disequality relation over B, which for $|B| > 2$ is NP-complete (Example 2.10). Hence, this straightforward conversion method does *not* necessarily preserve the tractability of Γ. To ensure that we do have this property we shall make use of a more sophisticated conversion technique, based on the following definition.

Definition 2.13 *For any m-ary relation ϱ over $\{A_1, \ldots, A_n\}$ with signature $\sigma(\varrho) = (i_1, \ldots, i_m)$, let $A = A_1 \times A_2 \times \cdots \times A_n$ and define the m-ary relation $\chi(\varrho)$ over A as follows:*

$$\chi(\varrho) = \{(\overline{a}_1, \overline{a}_2, \ldots, \overline{a}_m) \in A^m \mid (\overline{a}_1[1], \ldots, \overline{a}_m[m]) \in \varrho\}$$

For any set of relations Γ, the set $\{\chi(\varrho) \mid \varrho \in \Gamma\}$ is denoted $\chi(\Gamma)$.

Note that for any relation ϱ over a single set A, $\chi(\varrho) = \varrho$.

Example 2.14 Let ϱ be the binary disequality relation $\neq_A$ over A_1, A_2 with signature $(1, 2)$, as in Example 2.12. In this case $\chi(\varrho)$ consists of all pairs $((a, a'), (b, b')) \in (A_1 \times A_2) \times (A_1 \times A_2)$ such that a, b' belong to A and $a \neq b'$. □

Example 2.15 Each tuple of the relation ϱ from Example 2.2 corresponds to 72 tuples of $\chi(\varrho)$:

$$\chi(\varrho) = \left(\begin{array}{cccccccccccc} (0,a) & (0,a) & (0,a) & (0,a) & (0,a) & (0,a) & (0,a) & (0,a) & (0,a) & (0,b) & (0,b) & \\ (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & (1,a) & \\ (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & (2,a) & \ldots \\ (0,c) & (1,c) & (2,c) & (0,c) & (1,c) & (2,c) & (0,c) & (1,c) & (2,c) & (0,c) & (1,c) & \\ (0,c) & (0,c) & (0,c) & (1,c) & (1,c) & (1,c) & (2,c) & (2,c) & (2,c) & (0,c) & (0,c) & \end{array} \right).$$

□

Proposition 2.16 *For any set of relations* Γ *over finite sets* $A_1, \ldots, A_n$, $\mathrm{CSP}(\Gamma)$ *is tractable if and only if* $\mathrm{CSP}(\chi(\Gamma))$ *is tractable.*

Proof. Let Γ' be a finite subset of Γ, and $\mathcal{P} = (V; \{A_1, \ldots, A_n\}; \delta; \mathcal{C})$ an instance of $\mathrm{CSP}(\Gamma')$ where $\mathcal{C} = \{C_1, C_2, \ldots, C_q\}$ and $C_i = \langle s_i, \varrho_i \rangle$.

Consider the instance $\mathcal{P}' = (V; A; \mathcal{C}')$ where $A = A_1 \times A_2 \times \cdots \times A_n$, $\mathcal{C}' = \{C'_1, C'_2, \ldots, C'_q\}$, and $C'_i = \langle s_i, \chi(\varrho_i) \rangle$. Note that every solution to $\mathcal{P}$ can be used to construct a solution of $\mathcal{P}'$ by extending the value assigned to each variable (arbitrarily) to a tuple over A. Conversely, every solution to $\mathcal{P}'$ can be used to obtain a solution to $\mathcal{P}$ by projecting the tuple over A assigned to each variable v onto $A_{\delta(v)}$. Hence we have a reduction from $\mathrm{CSP}(\Gamma')$ to $\mathrm{CSP}(\chi(\Gamma'))$.

Furthermore, every tuple in every constraint relation of $\mathcal{P}$ is replaced by a fixed number of tuples (depending only on the cardinalities of the sets $A_1, \ldots, A_n$ and on the arities of relations from Γ'), to obtain the corresponding constraint relation in $\mathcal{P}'$. Hence, the reduction can be carried out in linear time.

The same arguments can be applied in the reverse direction to obtain a polynomial-time reduction from $\mathrm{CSP}(\chi(\Gamma'))$ to $\mathrm{CSP}(\Gamma')$. □

3 Algebraic techniques for the CSP

In earlier papers [20, 19, 3] we have shown that in the one-sorted case the complexity of $\mathrm{CSP}(\Gamma)$ is closely related to certain algebraic invariance properties of the relations in Γ. In this section we extend these results to the multi-sorted case.

We first define what is meant by an *algebra*.

Definition 3.1 *A* universal algebra *(or simply* algebra*) is an ordered pair* $\mathbb{A} = (A; F)$ *where* A *is a nonempty set and* F *is a family of finitary operations on* A*. The set* A *is called the* universe *(or the* base set*), and the operations from* F *are called* basic*. An algebra is said to be* finite *if its universe is finite.*

In the one-sorted case it has been shown [3] that for each problem class of the form $\mathrm{CSP}(\Gamma)$ there is a corresponding finite algebra $\mathbb{A}_\Gamma$ such that the tractability of Γ depends only on the structure $\mathbb{A}_\Gamma$. We now sketch how the algebra $\mathbb{A}_\Gamma$ is constructed.

Definition 3.2 *Let* f *be an* n*-ary operation on a set* A *and let* ϱ *be an* m*-ary relation over* A*. The operation* f preserves ϱ*, if for every* $(a_{11}, \ldots, a_{m1}), \ldots, (a_{1n}, \ldots, a_{mn}) \in \varrho$*, the tuple* $(f(a_{11}, \ldots, a_{1n}), \ldots, f(a_{m1}, \ldots, a_{mn}))$ *is an element of* ϱ*.*

Given a set of relations, Γ*, over* A*, the operations which preserve every relation in* Γ *are called* polymorphisms *of* Γ*, and the set of all such operations is denoted by* $\mathsf{Pol}\,\Gamma$*. The algebra* $(A, \mathsf{Pol}\,\Gamma)$ *is called the algebra associated with* Γ*, and is denoted* $\mathbb{A}_\Gamma$*.*

Theorem 3.3 ([18]) *For any set of relations Γ over a finite set A, and any finite set of relations Γ_0 over A such that $\mathsf{Pol}\,\Gamma \subseteq \mathsf{Pol}\,\Gamma_0$, there is a polynomial time reduction from* $\mathrm{CSP}(\Gamma_0)$ *to* $\mathrm{CSP}(\Gamma)$.

This result shows that the tractability of Γ is determined by the set of operations $\mathsf{Pol}\,\Gamma$, and therefore by the algebra $\mathbb{A}_\Gamma = (A; \mathsf{Pol}\,\Gamma)$.

In the reverse direction, we can start with an algebra and obtain a corresponding set of relations, in the following way. Given any set of operations, C, on A, the set of all relations over A which are preserved by all operations from C is denoted by $\mathsf{Inv}\,C$. Hence, given any algebra $\mathbb{A} = (A; F)$, we can define an associated set of relations $\mathsf{Inv}\,F$.

For any finite algebra $\mathbb{A} = (A; F)$ there are infinitely many sets Γ such that $\mathbb{A}_\Gamma = \mathbb{A}$. All of these sets are tractable (or intractable) simultaneously, and the union of all such sets is $\mathsf{Inv}\,F$. Hence, the algebra $\mathbb{A}$ will be said to be tractable (NP-complete) if and only if $\mathsf{Inv}\,F$ is tractable (NP-complete).

Using the definitions above, and Theorem 3.3, we can translate questions about the tractability of Γ in the one-sorted case into questions about the tractability of the corresponding algebra.

Corollary 3.4 *For any set of relations Γ over a finite set A, Γ is tractable if and only if $\mathbb{A}_\Gamma$ is tractable.*

It follows from Corollary 3.4 that all maximal tractable sets of one-sorted relations can be defined by specifying a suitable algebra. Of course, we may need very many operations to define an arbitrary algebra. However, in all known cases we need very few operations to ensure that an algebra is tractable (usually, a single binary or ternary operation, see Section 4). Hence the algebraic theory we have sketched here allows a very concise description of all known maximal tractable sets of relations, even though each maximal tractable set contains infinitely many relations.

An analogous result for the multi-sorted case can be obtained by combining Corollary 3.4 with Proposition 2.16, as follows.

Corollary 3.5 *For any set of relations Γ over finite sets $\{A_1, \ldots, A_n\}$, Γ is tractable if and only if $\mathbb{A}_{\chi(\Gamma)}$ is tractable.*

Although the algebra $\mathbb{A}_{\chi(\Gamma)}$ is a possible algebraic structure for $\mathrm{CSP}(\Gamma)$, it is often not the most convenient. For example, if we have 10 distinct domains each with 20 elements then $\mathbb{A}_{\chi(\Gamma)}$ is an algebra with 20^{10} elements. In the remainder of this section we are going to show that, for any multi-sorted problem class $\mathrm{CSP}(\Gamma)$, the domains $A_1, \ldots, A_n$ can each individually be endowed with operations such that the obtained collection of similar algebras $\{\mathbb{A}_1, \ldots, \mathbb{A}_n\}$ is tractable if and only if Γ is tractable. Using these results for the case just mentioned, we may deal with 10 20-element algebras, which is much more convenient, and allows more complete analysis of the algebraic structure.

To develop this novel theory concerning the tractability of *collections* of algebras we need to introduce more algebraic terminology.

First we note that algebras can be grouped into families according to the arities of their operations.

Definition 3.6 *Algebras* $\mathbb{A}_1 = (A_1, F_1)$, $\mathbb{A}_2 = (A_2, F_2)$ *are said to be* similar *(or of the* same type*) if there exists a set* I *such that* $F_1 = \{f_i^1 \mid i \in I\}$, $F_2 = \{f_i^2 \mid i \in I\}$ *and, for all* $i \in I$, f_i^1, f_i^2 *are of the same arity.*

Let $\mathcal{A}$ be a class of similar algebras. The index-set for the basic operations of algebras from $\mathcal{A}$ will be called the set of basic operations of $\mathcal{A}$. For each $\mathbb{A} \in \mathcal{A}$, the concrete operation corresponding to an index f will be called the $\mathbb{A}$-*interpretation* of f and denoted $f^{\mathbb{A}}$. We will also use *terms* of $\mathcal{A}$, constructed from basic operations of $\mathcal{A}$ in the standard way (see, for example, [16, 11]). Again, for each $\mathbb{A} \in \mathcal{A}$, the $\mathbb{A}$-interpretation of a term t is a concrete operation derived by the usual rules; it will be called a *term operation* and denoted $t^{\mathbb{A}}$.

We say that an (n-ary) term t preserves an (m-ary) multi-sorted relation ϱ over the collection of the universes of algebras from $\mathcal{A}$ if, for any $(a_{11}, \ldots, a_{m1}), \ldots, (a_{1n}, \ldots, a_{mn}) \in \varrho$ we have

$$t\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} = \begin{pmatrix} t^{\mathbb{A}_{\delta(1)}}(a_{11}, \ldots, a_{1n}) \\ \vdots \\ t^{\mathbb{A}_{\delta(m)}}(a_{m1}, \ldots, a_{mn}) \end{pmatrix} \in \varrho.$$

We shall make use of three standard constructions on algebras which are defined as follows:

Definition 3.7 *Let* $\mathbb{A}_1 = (A_1, F^{\mathbb{A}_1})$, $\mathbb{A}_2 = (A_2, F^{\mathbb{A}_2})$ *be similar algebras. A mapping* $\varphi: A_1 \to A_2$ *is called a* homomorphism *from* $\mathbb{A}_1$ *to* $\mathbb{A}_2$ *if* $\varphi f^{\mathbb{A}_1}(a_1, \ldots, a_k) = f^{\mathbb{A}_2}(\varphi(a_1), \ldots, \varphi(a_k))$ *holds for all* $f \in F$ *and all* $a_1, \ldots, a_k \in A_1$, *where* k *is the arity of* f. *If the map* φ *is surjective then* $\mathbb{A}_2$ *is called a* homomorphic image *of* $\mathbb{A}_1$.

Definition 3.8 *Let* $\mathbb{A} = (A, F)$ *be an algebra and* B *a subset of* A *such that, for any* $f \in F$, *with arity* k, *and for any* $b_1, \ldots, b_k \in B$, *we have* $f(b_1, \ldots, b_k) \in B$. *Then the algebra* $\mathbb{B} = (B, F|_B)$, *where* $F|_B$ *consists of restrictions of the operations in* F *to the set* B, *is called a* subalgebra *of* $\mathbb{A}$.

Definition 3.9 *Let* $\mathbb{A}_1, \ldots, \mathbb{A}_n$ *be similar algebras, and* F *a set of basic operations. The* direct product *of* $\mathbb{A}_1, \ldots, \mathbb{A}_n$ *is the algebra* $\mathbb{B} = (B; F^{\mathbb{B}})$ *where* $B = A_1 \times \ldots \times A_n$ *and the* $\mathbb{B}$*-interpretation of a basic operation* $f \in F$ *with arity* k *is defined as follows*

$$f^{\mathbb{B}}\left(\begin{pmatrix} a_{11} \\ \vdots \\ a_{n1} \end{pmatrix}, \ldots, \begin{pmatrix} a_{1k} \\ \vdots \\ a_{nk} \end{pmatrix}\right) = \begin{pmatrix} f^{\mathbb{A}_1}(a_{11}, \ldots, a_{1k}) \\ \vdots \\ f^{\mathbb{A}_n}(a_{n1}, \ldots, a_{nk}) \end{pmatrix}$$

where $a_{11}, \ldots, a_{1k} \in A_1, \ldots, a_{n1}, \ldots, a_{nk} \in A_n$. *The direct product of* $\mathbb{A}_1, \ldots, \mathbb{A}_n$ *is denoted by* $\mathbb{A}_1 \times \ldots \times \mathbb{A}_n$ *or* $\prod_{i \in \{1,\ldots,n\}} \mathbb{A}_i$.

For a class $\mathcal{A}$ of similar algebras, $\mathsf{SP}_{\text{fin}}(\mathcal{A})$ denotes the class of all subalgebras of direct products of algebras from $\mathcal{A}$. It is well known that, for any single finite algebra $\mathbb{A} = (A; F)$, $\mathsf{SP}_{\text{fin}}(\mathbb{A}) = \mathsf{Inv}\ F$ (see, for example, [30]).

Given an arbitrary collection of similar algebras, we can define an associated multi-sorted CSP in which the domains of the variables are the universes of these algebras and the constraint relations are subalgebras of direct products of these algebras.

Definition 3.10 *For given collection of similar finite algebras, $\mathcal{A}$,* CSP($\mathcal{A}$) *is defined to be the decision problem with*

INSTANCE: *An instance, $\mathcal{P} = (V; \mathcal{B}; \delta; \mathcal{C})$, of the multi-sorted constraint satisfaction problem, in which*

- *for each variable v, the domain $A_{\delta(v)}$ is the universe of an algebra $\mathbb{A}_{\delta(v)} \in \mathcal{A}$;*
- *for each constraint $\langle s, \varrho \rangle$, the relation ϱ is a subuniverse of the direct product $\prod_{w \in s} \mathbb{A}_{\delta(w)}$.*

QUESTION: *Does $\mathcal{P}$ have a solution?*

Note that, if we ignore the operations and consider only the universe, then the set $\mathsf{SP}_{\text{fin}}(\mathcal{A})$, is simply a set of relations over the universes of $\mathcal{A}$. It can be checked that the problem CSP($\mathcal{A}$) defined above corresponds exactly to the problem CSP($\mathsf{SP}_{\text{fin}}(\mathcal{A})$).

Definition 3.11 *A collection of algebras $\mathcal{A}$ is said to be* tractable *if* CSP($\mathcal{A}$) *is tractable. It is said to be NP-complete if* CSP($\mathcal{A}$) *is NP-complete.*

To conclude this section we show that, for a finite collection of sets, $\mathcal{A}$, and a set of relations Γ over $\mathcal{A}$, there exists a finite algebra $\mathbb{A}$, and a collection $\mathcal{A}'$ of similar algebras whose universes are the members of $\mathcal{A}$, such that Γ, $\mathbb{A}$, and $\mathcal{A}'$ are tractable or intractable simultaneously.

Theorem 3.12 *Let Γ be a set of relations over the finite sets $\{A_1, \ldots, A_n\}$. The following conditions are equivalent*
(a) Γ is tractable;
(b) $\widehat{\Gamma} = \Gamma \cup \{=_{A_1}, \ldots, =_{A_n}\}$ is tractable, where each $=_{A_i}$ is the binary equality relation on A_i;
(c) $\mathbb{A}_{\chi(\widehat{\Gamma})}$ is tractable;
(d) $\{\mathbb{A}_1, \ldots, \mathbb{A}_n\}$ is tractable, where each $\mathbb{A}_i$ is the image of $\mathbb{A}_{\chi(\widehat{\Gamma})}$ under the homomorphism φ_i given by $\varphi_i(\overline{a}) = \overline{a}[i]$.

Proof. (a)$\Longleftrightarrow$(b). Clearly $\Gamma \subseteq \widehat{\Gamma}$, so if $\widehat{\Gamma}$ is tractable, then Γ is tractable.

To establish the converse, we will show that CSP($\widehat{\Gamma}$) may be reduced in polynomial-time to CSP(Γ) (this is very similar to the proof of Proposition 3.1 in [18]).

Let $\mathcal{P} = (V; \mathcal{A}; \delta; \mathcal{C})$ be any instance of CSP($\widehat{\Gamma}$). We will modify $\mathcal{P}$ to construct an instance $\mathcal{P}'$ of CSP(Γ), as follows. For each constraint $C = \langle s, \varrho \rangle \in \mathcal{C}$, if ϱ is $=_{A_j}$ then

– if $s = (v_1, v_2)$ then replace all occurrences of v_1 in the constraint scopes of constraints from $\mathcal{C}$ with v_2;

– remove the constraint C.

Clearly, this construction can be carried out in polynomial time, and $\mathcal{P}'$ has a solution if and only if $\mathcal{P}$ has a solution.

(b)$\Longleftrightarrow$(c). Follows from Proposition 2.16.

(d)$\Rightarrow$(a). To establish this implication we will show that $\Gamma \subseteq \mathsf{SP}_{\text{fin}}(\mathbb{A}_1, \dots, \mathbb{A}_n)$, and therefore CSP($\Gamma$) is a subclass of CSP($\mathbb{A}_1, \dots, \mathbb{A}_n$).

To verify that each φ_i is indeed a homomorphism is a routine algebraic calculation, making use of the fact that every operation f of $\mathbb{A}_{\chi(\widehat{\Gamma})}$ preserves the relations $\chi(=_{A_1}), \dots, \chi(=_{A_n})$, and hence the ith component of the result of f depends only on the ith component of its arguments.

By the definition of the mappings φ_i, for any relation ϱ over $\{A_1, \dots, A_n\}$ with signature $(i_1, \dots, i_m)$, we have

$$\varrho = \{(\varphi_{i_1}(\overline{a}_1), \dots, \varphi_{i_m}(\overline{a}_m)) \mid (\overline{a}_1, \dots, \overline{a}_m) \in \chi(\varrho)\}.$$

Moreover, if $\varrho \in \Gamma$, then $\chi(\varrho) \in \mathsf{SP}_{\text{fin}}(\mathbb{A}_{\chi(\widehat{\Gamma})})$, and hence $\chi(\varrho)$ is preserved by every operation of $\mathbb{A}_{\chi(\widehat{\Gamma})}$. Now let f be an arbitrary basic operation of arity k, let $(a_{11}, \dots, a_{m1}), \dots, (a_{1k}, \dots, a_{mk}) \in \varrho$, and let $(\overline{b}_{11}, \dots, \overline{b}_{m1}), \dots, (\overline{b}_{1k}, \dots, \overline{b}_{mk})$ be the corresponding elements of $\chi(\varrho)$. We have $(a_{1i}, \dots, a_{mi}) = (\varphi_{i_1}(\overline{b}_{1i}), \dots, \varphi_{i_m}(\overline{b}_{mi}))$. Hence

$$f\begin{pmatrix} a_{11} & \dots & a_{1k} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mk} \end{pmatrix} = \begin{pmatrix} f^{\mathbb{A}_{i_1}}(\varphi_{i_1}(\overline{b}_{11}), \dots, \varphi_{i_1}(\overline{b}_{1k})) \\ \vdots \\ f^{\mathbb{A}_{i_m}}(\varphi_{i_m}(\overline{b}_{m1}), \dots, \varphi_{i_m}(\overline{b}_{mk})) \end{pmatrix}$$
$$= \begin{pmatrix} \varphi_{i_1} f^{\mathbb{A}_{\chi(\widehat{\Gamma})}}(\overline{b}_{11}, \dots, \overline{b}_{1k}) \\ \vdots \\ \varphi_{i_m} f^{\mathbb{A}_{\chi(\widehat{\Gamma})}}(\overline{b}_{m1}, \dots, \overline{b}_{mk}) \end{pmatrix} \in \varrho.$$

So, f preserves ϱ, and hence $\Gamma \subseteq \mathsf{SP}_{\text{fin}}(\mathbb{A}_1, \dots, \mathbb{A}_n)$.

(c)$\Rightarrow$(d). Let $\Delta \subseteq \mathsf{SP}_{\text{fin}}(\mathbb{A}_1, \dots, \mathbb{A}_n)$ be finite. By Proposition 2.16, CSP(Δ) is polynomial-time reducible to CSP($\chi(\Delta)$), and since $\chi(\Delta) \subseteq \mathsf{SP}_{\text{fin}}(\mathbb{A}_{\chi(\widehat{\Gamma})})$, we get the required reducibility. $\Box$

If we start with an arbitrary collection of finite algebras, $\mathbb{A}_1, \dots, \mathbb{A}_n$, then by choosing Γ appropriately we can ensure that the algebra $\mathbb{A}_{\chi(\widehat{\Gamma})}$ is the direct product of $\mathbb{A}_1, \dots, \mathbb{A}_n$. Hence, we obtain the following corollary.

Corollary 3.13 *Given finite similar algebras $\mathbb{A}_1, \dots, \mathbb{A}_n$, the collection $\{\mathbb{A}_1, \dots, \mathbb{A}_n\}$ is tractable if and only if $\mathbb{A}_1 \times \dots \times \mathbb{A}_n$ is tractable.*

4 Tractable multi-sorted problems

In this section we exhibit some conditions which allow us to derive the tractability of a collection of algebras from properties of the individual algebras, and hence derive the tractability of certain multi-sorted problem classes from the tractability of one-sorted classes.

Clearly, for a collection of algebras to be tractable it is necessary that each member of the collection is tractable. However, it is not clear that this condition is sufficient. In fact there is currently no complete answer to the question: is a collection of tractable algebras tractable? The answer to this question seems likely to be yes, but a proof of this appears to require some deep results about the structure of algebras. If the answer is yes, then it would mean that, when we are dealing with algebras, the tractability of a multi-sorted problem *can* be decided by considering the problem restricted to each domain separately. (This might appear to contradict Example 2.11, but in fact, the algebraic structure imposes much stronger restrictions on the problem class than just knowing the projections of the relations.) In this section we give some preliminary results in this direction.

In a series of papers [21, 20, 19] certain types of algebras which give rise to tractable problem classes were identified.

Definition 4.1

- *An operation f such that there is $c \in A$ with $f(x_1, \ldots, x_n) = c$ is called* constant *operation.*
- *An operation f on A is called* idempotent *if $f(x, \ldots, x) = x$ for any $x \in A$.*
- *A binary operation $f(x, y)$ is called a* semilattice *operation*[1]*, if it satisfies the following three conditions:*
  - $f(x, f(y, z)) = f(f(x, y), z)$ *(Associativity)*
  - $f(x, y) = f(y, x)$ *(Commutativity)*
  - $f(x, x) = x$ *(Idempotency)*
- *An n-ary operation f is said to be a* near-unanimity *operation if* $f(y, x, \ldots, x) = f(x, y, x, \ldots, x) = \cdots = f(x, \ldots, x, y) = x$ *for any $x, y \in A$.*
- *A ternary operation f is said to be* affine *if $f(x, y, z) = x - y + z$ for any $x, y, z \in A$, where $(A, +, -)$ is an Abelian group.*

Proposition 4.2 ([21, 20, 19]) *If a finite algebra $\mathbb{A}$ has a term operation which is constant, semilattice, affine, or near-unanimity, then $\mathbb{A}$ is tractable.*

As the following theorem shows, all these one-sorted tractable classes can be combined in almost arbitrary ways to obtain a new multi-sorted tractable class. The only restriction is the idempotency of the affine terms (if any).

[1] Note that in some earlier papers [20, 18] the term *ACI operation* is used

Theorem 4.3 *Let $\mathbb{A}_1, \ldots, \mathbb{A}_l, \mathbb{B}_1, \ldots, \mathbb{B}_n$ be similar finite algebras. If each $\mathbb{B}_i$ has either a constant term, or a semilattice or near-unanimity term, and each $\mathbb{A}_i$ has an affine term whose interpretations on other algebras are idempotent, then $\{\mathbb{A}_1, \ldots, \mathbb{A}_l, \mathbb{B}_1, \ldots, \mathbb{B}_n\}$ is tractable.*

Example 4.4 Any single constraint whose signature contains no repeating entries is tractable. To see this, let ϱ be an n-ary relation over $\mathcal{A} = \{A_i \mid i \in I\}$ such that $\sigma(\varrho)[i] \neq \sigma(\varrho)[j]$ for all $1 \leq i, j \leq n$, $i \neq j$. If $\varrho = \varnothing$ then CSP(ϱ) is obviously tractable. Otherwise, take $\overline{a} \in \varrho$, and define a unary term $f(x)$ by the rule $f^{A_{\sigma(\varrho)[i]}}(x) = \overline{a}[i]$, and $f^{A_j}(x) = c_j$ for arbitrary $c_j \in A_j$ if $j \in I$ does not occur in $\sigma(\varrho)$. The term f is constant on each set from $\mathcal{A}$, and preserves ϱ. Therefore, by Theorem 4.3 CSP(ϱ) $\subseteq$ CSP($\mathbb{A}_i \mid i \in I$) where $\mathbb{A}_i = (A_i; f)$ is tractable. □

Example 4.5 Recall the relation ϱ over $A_1 = \{0, 1, 2\}$, $A_2 = \{a, b, c\}$ defined in Example 2.2 We can now prove that $\{\varrho\}$ is tractable. To see this, it is sufficient to check that ϱ is preserved by terms $t(x, y, z)$ and $g(x, y)$ where $t^{A_1}(x, y, z) = x - y + z$ is the affine operation of the group $\mathbb{Z}_3$, $g^{A_1}(x, y) = y$, and t^{A_2}, g^{A_2} are defined via the following Cayley tables

$t^{A_2}(x,y,a)$	a	b	c
a	a	c	c
b	c	c	c
c	c	c	c

$t^{A_2}(x,y,b)$	a	b	c
a	c	c	c
b	c	b	c
c	c	c	c

$t^{A_2}(x,y,c)$	a	b	c
a	c	c	c
b	c	c	c
c	c	c	c

,

$g^{A_2}(x,y)$	a	b	c
a	a	b	c
b	b	b	c
c	c	c	c

.

Moreover, g^{A_2} is a semilattice operation, and t^{A_2} is idempotent, and so we can apply Theorem 4.3. □

5 Conclusion

The ultimate goal of the research direction presented here is to characterize tractable sets of relations. In the Introduction we observed that restricting the set of constraint relations for the CSP corresponds to restricting the possible form of the database tables in CQ-EVALUATION. This seems to be a fruitful research direction because in many cases the form of a database is naturally restricted by the context of an application. We have shown here that using algebraic properties to capture the essential properties of such restrictions provides a useful theoretical tool for the analysis of such situations.

In [3], a possible characterization of tractable relations was conjectured for the one-sorted case.

Conjecture 5.1 *A finite algebra $\mathbb{A}$ is tractable if and only if every homomorphic image of every subalgebra has a non-unary term operation. Otherwise it is NP-complete.*

In can be shown that, if this conjecture is true, then for the multi-sorted case a similar characterization holds.

Conjecture 5.2 *A collection $\mathcal{A}$ of finite algebras is tractable if and only if every homomorphic image of any member from* $\mathsf{SP}_{\text{fin}}(\mathcal{A})$ *has a non-unary term operation. Otherwise it is NP-complete.*

Using the powerful algebraic results obtained in [17], it is possible to derive a simple algorithm which verifies the algebraic conditions in both conjectures. Thus, if these conjectures hold, then the tractability or NP-completeness of a set of relations can be easily recognized by considering the associated algebras, even in the multi-sorted case.

References

[1] A. Bulatov and P. Jeavons. Tractable constraints closed under a binary operation. Technical Report PRG-TR-12-00, Oxford University Computing Laboratory, http://web.comlab.ox.ac.uk/oucl/publications/tr/index.html, 2000.

[2] A.A. Bulatov. Mal'tsev constraints are tractable. in preparation.

[3] A.A. Bulatov, A.A. Krokhin, and P.G. Jeavons. Constraint satisfaction problems and finite algebras. In *Proceedings 27th International Colloquium on Automata, Languages and Programming—ICALP'00*, volume 1853 of *Lecture Notes in Computer Science*, pages 272–282. Springer-Verlag, 2000.

[4] A.K. Chandra and P.M. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *Proceeedings 9th ACM Symp. on Theory of Computing*, pages 77–90, 1977.

[5] C. Chekuri and A. Rajaraman. Conjunctive query containment revisited. In *Proceedings 5th International Conference on Database Theory*, volume 1186 of *Lecture Notes in Computer Science*, pages 56–70. Springer-Verlag, 1997.

[6] D. Cohen, P. Jeavons, P. Jonsson, and M. Koubarakis. Building tractable disjunctive constraints. *Journal of the ACM*, 47:826–853, 2000.

[7] M.C. Cooper. An optimal k-consistency algorithm. *Artificial Intelligence*, 41:89–95, 1989.

[8] M.C. Cooper, D.A. Cohen, and P.G. Jeavons. Characterising tractable constraints. *Artificial Intelligence*, 65:347–361, 1994.

[9] V. Dalmau. *Computational Complexity of Problems over Generalized Formulas.* PhD thesis, UPC, Barcelona, 2000.

[10] R. Dechter and J. Pearl. Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence*, 34(1):1–38, 1988.

[11] H.B. Enderton. *A mathematical introduction to logic.* Harcourt/Academic Press, 2001.

[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 of Computing*, 28:57–104, 1998.

[13] M. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, San Francisco, CA., 1979.

[14] G. Gottlob, N. Leone, and F. Scarcello. The complexity of acyclic conjunctive queries. In *Proceeedings of the 39th Annual Symposium on Foundations of Computer Science*, pages 706–715, 1998.

[15] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. In *Proceeedings of the 18th Symposium on Principles of Database Systems*, pages 21–32, 1999.

[16] A. G. Hamilton. *Logic for mathematicians.* Cambridge University Press, 1988.

[17] D. Hobby and R.N. McKenzie. *The Structure of Finite Algebras*, volume 76 of *Contemporary Mathematics.* American Mathematical Society, Providence, R.I., 1988.

[18] P.G. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

[19] P.G. Jeavons, D.A. Cohen, and M.C. Cooper. Constraints, consistency and closure. *Artificial Intelligence*, 101(1-2):251–265, 1998.

[20] P.G. Jeavons, D.A. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44:527–548, 1997.

[21] P.G. Jeavons and M.C. Cooper. Tractable constraints on ordered domains. *Artificial Intelligence*, 79(2):327–339, 1995.

[22] L. Kirousis. Fast parallel constraint satisfaction. *Artificial Intelligence*, 64:147–160, 1993.

[23] P.G. Kolaitis and M.Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Information and Computation*, 61:302–332, 2000.

[24] P.B. Ladkin and R.D. Maddux. On binary constraint problems. *Journal of the ACM*, 41:435–469, 1994.

[25] A.K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.

[26] A.K. Mackworth. Constraint satisfaction. In S.C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, volume 1, pages 285–293. Wiley Interscience, 1992.

[27] A.K. Mackworth and E.C. Freuder. The complexity of constraint satisfaction revisited. *Artificial Intelligence*, 59:57–62, 1993.

[28] R.N. McKenzie, G.F. McNulty, and W.F. Taylor. *Algebras, Lattices and Varieties*, volume I. Wadsworth and Brooks, California, 1987.

[29] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 1974.

[30] R. Pöschel and L.A. Kalužnin. *Funktionen- und Relationenalgebren.* DVW, Berlin, 1979.

[31] T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th ACM Symposium on Theory of Computing (STOC'78)*, pages 216–226, 1978.

[32] E. Tsang. *Foundations of Constraint Satisfaction.* Academic Press, London, 1993.

[33] P. van Beek and R. Dechter. On the minimality and decomposability of row-convex constraint networks. *Journal of the ACM*, 42:543–561, 1995.

[34] P. van Hentenryck, Y. Deville, and C-M. Teng. A generic arc-consistency algorithm and its specializations. *Artificial Intelligence*, 57:291–321, 1992.

[35] M.Y. Vardi. The complexity of relational query languages. In *Proceeedings of the 14th ACM Symposium on Theory of Computing*, pages 137–146, 1982.

[36] M. Yannakakis. Algorithms for acyclic database schemes. In *Proceeedings of the 7th International Confernece on Very Large Databases*, pages 82–94, 1981.

Appendix: Proof of Theorem 4.3

The proof of Theorem 4.3 is set out in Propositions A.6, A.7, and A.9.

Before giving the proofs, we need to introduce the standard notion of k-consistency which has been widely used in the study of constraint satisfaction problems.

For an n-ary relation ϱ and a set of indices $I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$, the relation $\mathrm{pr}_I \varrho = \{(a_{i_1}, \ldots, a_{i_k}) \mid (a_1, \ldots, a_n) \in \varrho\}$ is called the *projection* of ϱ onto I.

Definition A.3 *Let $\mathcal{P} = (V; \mathcal{A}; \delta; \mathcal{C})$ be a problem instance. For any subset W of V, the* subproblem of $\mathcal{P}$ generated by W, *denoted $\mathcal{P}|_W$, is the problem instance $(W; \mathcal{A}; \delta|_W; \mathcal{C}')$ where the constraints are obtained from the constraints of $\mathcal{P}$ as follows: for each constraint $\langle s, \varrho\rangle \in \mathcal{C}$ of $\mathcal{P}$, choose $s' = s \cap W$ to be a list of the elements of W in s, and make $\langle s', \mathrm{pr}_{s'} \varrho\rangle$ a constraint of $\mathcal{P}|_W$.*

Definition A.4 *A constraint satisfaction problem $\mathcal{P}$ is said to be* k-consistent *if for any subset W containing $k-1$ variables, and any variable v, any solution to $\mathcal{P}|_W$ can be extended to a solution to $\mathcal{P}|_{W \cup \{v\}}$.*

If $\mathcal{P}$ is i-consistent for $1 \leq i \leq k$, then it is said to be strong k-consistent.

Any constraint satisfaction problem instance $\mathcal{P}$ can be modified to obtain a k-consistent problem instance $\mathcal{P}'$ without changing the set of solutions by solving all subproblems involving k variables, and then imposing constraints on all subsets of $k-1$ and k variables that allow only these solutions. This procedure is called 'establishing k-consistency', and $\mathcal{P}'$ is said to be the k-consistent instance associated with $\mathcal{P}$ (see [7] for the one-sorted case).

Definition A.5 *A class* **C** *of constraint satisfaction problems is said to be of* width k *if any problem instance $\mathcal{P}$ from* **C** *has a solution if and only if the k-consistent problem associated with $\mathcal{P}$ contains no empty constraint.*

Feder and Vardi [12] characterised this concepts in terms of Datalog programs. Notice that the notion of width defined above is exactly equivalent to the notion of width, as defined in [12].

Every class of finite width is tractable, because, for any fixed k, establishing k-consistency takes polynomial time, and recognising the presence of empty constraints can be carried out in linear time.

Proposition A.6 *Let $\mathbb{A}_1, \ldots, \mathbb{A}_n$ be similar finite algebras, where each $\mathbb{A}_i$ has either a semilattice or a near-unanimity term operation. Then $\mathrm{CSP}(\mathbb{A}_1, \ldots, \mathbb{A}_n)$ is of finite width, and therefore, tractable.*

Proof. By Theorem 3.5 of [19] and Theorem 24 of [9], for each i the class $\mathrm{CSP}(\mathbb{A}_i)$ is of width k_i if $\mathbb{A}_i$ has a k_i-ary near-unanimity term operation, and of width $k_i = 1$ if $\mathbb{A}_i$ has a semilattice term operation. We prove by induction that $\mathrm{CSP}(\mathbb{A}_1, \ldots, \mathbb{A}_j)$, $j \leq n$ is of width $k = k_1 + \ldots + k_j$. The base case $j = 1$ follows from the results just quoted.

Consider a problem instance $\mathcal{P} = (V; \{\mathbb{A}_1, \dots, \mathbb{A}_j\}; \delta; \mathcal{C})$. Without loss of generality $\mathcal{P}$ may be assumed to be strong k-consistent. Set $V_j = \{v \in V \mid \delta(v) = j\}$ and $\overline{V} = V - V_j$. The instance $\mathcal{P}|_{\overline{V}}$ belongs to the class $\mathrm{CSP}(\mathbb{A}_1, \dots, \mathbb{A}_{j-1})$.

We consider two cases.

CASE 1. $k_j > 1$, and therefore, $\mathbb{A}_j$ has a k_j-ary near-unanimity term t.

In this case we are going to prove by induction that for any $W \subseteq V_j$, and any solution φ to $\mathcal{P}|_W$, φ can be extended to a solution to $\mathcal{P}|_{W \cup \overline{V}}$.

The base case is easy. Let $|W| = k_j$, φ be a solution to $\mathcal{P}|_W$, and $\mathcal{P}_\varphi$ denote the problem instance $(\overline{V}; \{\mathbb{A}_1, \dots, \mathbb{A}_{j-1}\}; \delta'; \mathcal{C}')$ where δ' is the restriction of δ onto $\overline{V}$, and for each $C = \langle s, \varrho\rangle \in \mathcal{C}$ the constraint $C' = \langle s', \varrho'\rangle$ with $s' = s \cap \overline{V}$, $\varrho' = \{\mathrm{pr}_{s'}\overline{a} \mid \overline{a} \in \varrho$ and $\overline{a}[v] = \varphi(v)$ for each $v \in s \cap W\}$. Since $\mathcal{P}$ is strong k-consistent, for any l-element subset $U \subseteq \overline{V}$, $l \le k - k_j - 1$ and any $w \in \overline{V} - U$, there is a solution to $\mathcal{P}_\varphi|_U$, and this solution can be extended to a solution to $\mathcal{P}_\varphi|_{U \cup \{w\}}$. This means that $\mathcal{P}_\varphi$ is strong $(k - k_j - 1)$-consistent, and hence, has a solution.

For the inductive step, let $W \subseteq V_j$, $|W| = m > k_j$, $v_1, \dots, v_{k_j} \in W$, and $W_i = W - \{v_i\}$. Take a solution φ to $\mathcal{P}|_W$. By the inductive hypothesis, for each $1 \le i \le k_j$, the solution $\varphi_i = \varphi|_{W_i}$ can be extended to a solution ψ_i to $\mathcal{P}|_{W_i \cup \overline{V}}$ such that $\psi_i|_{W_i} = \varphi_i$. We prove that the mapping ψ with

$$\psi(v) = \begin{cases} \varphi(v), & \text{if } v \in W, \\ t^{\mathbb{A}_{\delta(v)}}(\psi_1(v), \dots, \psi_{k_j}(v)), & \text{if } v \in \overline{V} \end{cases}$$

is a solution to $\mathcal{P}|_{W \cup \overline{V}}$.

To see this, take a constraint $\langle s, \varrho\rangle$ of the instance $\mathcal{P}|_{W \cup \overline{V}}$. Since $\psi_1, \dots, \psi_{k_j}$ are solutions of the corresponding problem instances, there are $\overline{a}_1, \dots, \overline{a}_{k_j} \in \varrho$ such that $\overline{a}_i[v] = \psi_i(v)$, for any $v \in s \cap (W_i \cup \overline{V})$. Then, for the tuple $\overline{b} = t(\overline{a}_1, \dots, \overline{a}_{k_j}) \in \varrho$, we have

$$\begin{aligned} \overline{b}[v] &= \begin{cases} t^{\mathbb{A}_i}(\psi_1(v), \dots, \psi_{i-1}(v), \overline{a}_i[v], \psi_{i+1}(v), \dots, \psi_{k_j}(v)), & \text{if } v = v_i, \\ t^{\mathbb{A}_{\delta(v)}}(\psi_1(v), \dots, \psi_{k_j}(v)), & \text{if } v \in \overline{V} \end{cases} \\ &= \begin{cases} t^{\mathbb{A}_j}(\varphi(v), \dots, \varphi(v), \overline{a}_i[v], \varphi(v), \dots, \varphi(v)), & \text{if } v = v_i, \\ \psi(v), & \text{if } v \in \overline{V} \end{cases} = \psi(v). \end{aligned}$$

This means that ψ satisfies every constraint, and hence is a solution of $\mathcal{P}|_{W \cup \overline{V}}$.

CASE 2. $k_j = 1$, and therefore, $\mathbb{A}_j$ has a semilattice term t.

In this case, for each $v \in V_j$, there is a solution $\max(v)$ to $\mathcal{P}|_{\{v\}}$ such that for any other solution a, $t^{\mathbb{A}_{\delta(v)}}(\max(v), a) = t^{\mathbb{A}_{\delta(v)}}(a, \max(v)) = \max(v)$. We prove that for any $W \subseteq V_j$, the solution $\max|_W$ to $\mathcal{P}|_W$ can be extended to a solution to $\mathcal{P}|_{W \cup \overline{V}}$.

The base case $|W| = 1$ can be proved by the same arguments as in Case 1.

Let $W \subseteq V_j$, $|W| = m > 1$, $w \in W$, and $W' = W - \{w\}$. By the inductive hypothesis, $\max|_{W'}$ and $\max|_{\{w\}}$ can be extended to a solution φ to $\mathcal{P}|_{W' \cup \overline{V}}$, and φ' to $\mathcal{P}|_{\{w\} \cup \overline{V}}$

respectively such that $\varphi\big|_{W'} = \max\big|_{W'}$, $\varphi'(w) = \max(w)$. We prove that the mapping ψ with

$$\psi(v) = \begin{cases} \max(v), & \text{if } v \in W, \\ t^{\mathbb{A}_{\delta(v)}}(\varphi(v), \varphi'(v)), & \text{if } v \in \overline{V} \end{cases}$$

is a solution to $\mathcal{P}\big|_{W \cup \overline{V}}$.

As in Case 1, for any constraint $\langle s, \varrho \rangle$ of $\mathcal{P}\big|_{W \cup \overline{V}}$, there are $\overline{a}, \overline{b} \in \varrho$ such that $\overline{a}[v] = \varphi(v)$ whenever $v \neq w$, and $\overline{b}[v] = \varphi'(v)$ whenever $v \notin W'$. Then for the tuple $\overline{c} = t(\overline{a}, \overline{b}) \in \varrho$ we have

$$\begin{aligned} \overline{c}[v] &= \begin{cases} t^{\mathbb{A}_j}(\max(v), \overline{b}[v]), & \text{if } v \in W', \\ t^{\mathbb{A}_j}(\overline{a}[v], \max(v)), & \text{if } v = w, \\ t^{\mathbb{A}_{\delta(v)}}(\varphi(v), \varphi'(v)), & \text{if } v \in \overline{V} \end{cases} \\ &= \begin{cases} \max(v), & \text{if } v \in W, \\ t^{\mathbb{A}_{\delta(v)}}(\varphi(v), \varphi'(v)), & \text{if } v \in \overline{V} \end{cases} = \psi(v). \end{aligned}$$

This means that ψ satisfies every constraint, and hence is a solution to $\mathcal{P}\big|_{W \cup \overline{V}}$. $\square$

Proposition A.7 *Let* $\mathbb{A}_1, \ldots, \mathbb{A}_n$ *be similar finite algebras, where each* $\mathbb{A}_1, \ldots, \mathbb{A}_l$ *has an affine term whose interpretations on other algebras are idempotent, and* $\mathrm{CSP}(\mathbb{A}_{l+1}, \ldots, \mathbb{A}_n)$ *is of finite width. Then* $\mathrm{CSP}(\mathbb{A}_1, \ldots, \mathbb{A}_n)$ *is tractable.*

Remark A.8 The conditions of Proposition A.7 mean, in particular, that there is a term $d(x, y, z)$ such that $d^{\mathbb{A}_i}(x, y, z) = x -_i y +_i z$ for $i \leq l$ where $+_i, -_i$ are operations of an Abelian group which has the same universe as $\mathbb{A}_i$. So, $\mathbb{A}_1 \times \ldots \times \mathbb{A}_l$ has an affine operation, and therefore, any problem instance from $\mathrm{CSP}(\mathbb{A}_1 \times \ldots \times \mathbb{A}_l)$ is solvable in polynomial time by an algorithm similar to Gaussian elimination. The same is true for the class $\mathrm{CSP}(\mathbb{A}_1, \ldots, \mathbb{A}_l)$. Moreover, this solution algorithm can be modified to find a *basis* of the solution space, and hence a representation for the complete set of solutions which is polynomial in the size of the problem.

Proof. Consider a problem instance $\mathcal{P} = (V; \{\mathbb{A}_1, \ldots, \mathbb{A}_n\}; \delta; \mathcal{C})$. Without loss of generality $\mathcal{P}$ may be assumed to be strong k-consistent, where k is the width of $\mathrm{CSP}(\mathbb{A}_{l+1}, \ldots, \mathbb{A}_n)$. Also, let $V_A = \{v \in V \mid \delta(v) \leq l\}$, $V_B = V - V_A$, $W \subseteq V_B$, and $\overline{a} \in \prod_{v \in W} \mathbb{A}_{\delta(v)}$. By $\mathcal{P}_{\overline{a}}$ we denote the problem instance $(V_A; \{\mathbb{A}_1, \ldots, \mathbb{A}_l\}; \delta\big|_{V_A}; \mathcal{C}')$ where for each $C = \langle s, \varrho \rangle \in \mathcal{C}$ there is $C' = \langle s', \varrho' \rangle \in \mathcal{C}'$ such that

- $s' = s \cap V_A$;
- $\varrho' = \{\overline{b} \in \mathrm{pr}_{s'} \varrho \mid$ there is $\overline{c} \in \mathrm{pr}_{s \cap (W \cup V_A)} \varrho$ such that $\overline{c}[v] = \overline{b}[v]$ for $v \in s'$, and $\overline{c}[v] = \overline{a}[v]$ for $v \in W \cap s\}$.

Notice that $\mathcal{P}_{\overline{a}}$ is a problem instance from $\mathrm{CSP}(\mathbb{A}_1, \ldots, \mathbb{A}_l)$, and hence, it can be solved in polynomial time. Moreover, the solving algorithm can be modified to find a basis of the solution space.

Then, repeat the following procedure while it changes the problem instance.

- `For each` $(k-1)$`-element subset` W `of` V_B
  - `For each solution` $\overline{a}$ `to` $\mathcal{P}|_W$
    - `For each` $w \in V_B - W$
      - `For each solution` $\overline{b}$ `to` $\mathcal{P}|_{W\cup\{w\}}$ `such that` $\overline{b}[v] = \overline{a}[v]$ `for any` $v \in W$
        - `Find a basis of the solution space` $S_{\overline{b}}$ `of` $\mathcal{P}_{\overline{b}}$
        - `If` $S_{\overline{b}} = \varnothing$ `remove` $\overline{b}$ `from the set of solutions to` $\mathcal{P}|_{W\cup\{w\}}$
      - `Find a basis of the set` $S_{\overline{a},w}$ `spanned by` $\bigcup S_{\overline{b}}$ `where` $\overline{b}$ `runs over the set of solutions to` $\mathcal{P}|_{W\cup\{w\}}$ `such that` $\overline{b}[v] = \overline{a}[v]$ `for any` $v \in W$
    - `Find a basis of the intersection` $S_{\overline{a}} = \bigcap_{w\in V_B - W} S_{\overline{a},w}$
    - `If` $S_{\overline{a}} = \varnothing$ `remove` $\overline{a}$ `from the set of solutions of` $\mathcal{P}|_W$
  - `Find a basis of the set` S_W `spanned by` $\bigcup S_{\overline{a}}$ `where` $\overline{a}$ `runs over the set of solutions of` $\mathcal{P}|_W$
- `Find a basis of` $S = \bigcap S_W$ `where` W `runs over all` $(k-1)$`-element subsets of` V_B
- `If` $S = \varnothing$ `then` $\mathcal{P}$ `has no solution`
- `For each constraint` $C = \langle s, \varrho\rangle \in \mathcal{C}$ `remove all tuples` $\overline{c} \in \varrho$ `such that` $(\overline{c}[v_1], \dots, \overline{c}[v_t]) \notin S|_{s\cap V_A}$ `where` $s \cap V_A = \{v_1, \dots, v_t\}$
- `Establish strong` k`-consistency of the obtained problem instance`

Let us denote the problem instance obtained after all required repetitions of this procedure by $\mathcal{P}'$.

For disjoint subsets $U, W \subseteq V$ and tuples $\overline{c}_1 \in \prod_{v\in U} \mathbb{A}_{\delta(v)}$, $\overline{c}_2 \in \prod_{v\in W} \mathbb{A}_{\delta(v)}$, the tuple $\overline{c} \in \prod_{v\in U\cup W} \mathbb{A}_{\delta(v)}$ such that $\overline{c}[v] = \overline{c}_1[v]$ for $v \in U$ and $\overline{c}[v] = \overline{c}_2[v]$ for $v \in W$ will be denoted by $(\overline{c}_1, \overline{c}_2)$.

Claim 1. For any $\overline{c} \in S_{\overline{a},w}$, there is a solution $\overline{b}$ to $\mathcal{P}|_{W\cup\{w\}}$ such that $\overline{b}[v] = \overline{a}[v]$ for $v \in W$, and $(\overline{c}, \overline{b})$ is a solution to $\mathcal{P}|_{V_A\cup W\cup\{w\}}$.

By the definition of $S_{\overline{a},w}$, $\overline{c} = t(\overline{c}_{\overline{b}_1}, \dots, \overline{c}_{\overline{b}_m})$ where $\overline{c}_{\overline{b}_i} \in S_{\overline{b}_i}$, $\overline{b}_i[v] = \overline{a}[v]$ for $v \in W$, and t is a certain term built using the term d. Since $(\overline{c}_{\overline{b}_i}, \overline{b}_i)$ is a solution to $\mathcal{P}|_{V_A\cup W\cup\{w\}}$,

$$\begin{pmatrix} \overline{c} \\ \overline{b} \end{pmatrix} = t\left(\begin{pmatrix} \overline{c}_{\overline{b}_1} \\ \overline{b}_1 \end{pmatrix}, \dots, \begin{pmatrix} \overline{c}_{\overline{b}_m} \\ \overline{b}_m \end{pmatrix}\right)$$

where $\overline{b} = t(\overline{b}_1, \dots, \overline{b}_m)$, and $\overline{b}[v] = t^{\mathbb{A}_{\delta(v)}}(\overline{b}_1[v], \dots, \overline{b}_m[v]) = t^{\mathbb{A}_{\delta(v)}}(\overline{a}[v], \dots, \overline{a}[v]) = \overline{a}[v]$ for $v \in W$ (by the idempotency of $t^{\mathbb{A}_{\delta(v)}}$), is a solution to $\mathcal{P}|_{V_A\cup W\cup\{w\}}$.

The following statement can be verified in a very similar way.

Claim 2. For any $\overline{c} \in S_W$ there is a solution $\overline{a}$ to $\mathcal{P}|_W$ such that $(\overline{c}, \overline{a})$ is a solution to $\mathcal{P}|_{V_A \cup W}$. Moreover, for any $w \in V_B - W$, there is a solution $\overline{b}$ to $\mathcal{P}|_{W \cup \{w\}}$ such that $\overline{b}[v] = \overline{a}[v]$ for $v \in W$, and $(\overline{c}, \overline{b})$ is a solution to $\mathcal{P}|_{V_A \cup W \cup \{w\}}$.

Claim 3. Any $\overline{c} \in S$ can be extended to a solution to $\mathcal{P}'$.

Consider the problem instance $\mathcal{P}'_{\overline{c}} = (V_B; \{\mathbb{A}_{l+1}, \ldots, \mathbb{A}_n\}; \delta|_{V_B}; \mathcal{C}')$ defined in an analogous way to $\mathcal{P}_{\overline{a}}$ at the beginning of the proof. It belongs to the class $\mathrm{CSP}(\mathbb{A}_{l+1}, \ldots, \mathbb{A}_n)$, and by Claim 2, is k-consistent. Hence, it has a solution. □

Proposition A.9 *Let $\mathbb{A}_1, \ldots, \mathbb{A}_n$ be similar finite algebras, where each $\mathbb{A}_i$, $1 \le i \le l$, has a constant term operation, and $\{\mathbb{A}_{l+1}, \ldots, \mathbb{A}_n\}$ is tractable. Then $\{\mathbb{A}_1, \ldots, \mathbb{A}_n\}$ is also tractable.*

Proof. Let $t_1(x_1, \ldots, x_{m_1}), \ldots, t_l(x_1, \ldots, x_{m_l})$ be terms such that $t_i^{\mathbb{A}_i}$ is a constant operation. Then $t'_i(x) = t_i(x, \ldots, x)$ also satisfies this condition. Finally, the term $t(x) = t'_1(\ldots t'_l(x) \ldots)$ is constant for all $\mathbb{A}_1, \ldots, \mathbb{A}_l$. Denote the constant $t^{\mathbb{A}_i}$ by c_i.

Take a problem instance $\mathcal{P} = (V; \{\mathbb{A}_1, \ldots, \mathbb{A}_n\}; \delta; \mathcal{C})$ from $\mathrm{CSP}(\mathbb{A}_1, \ldots, \mathbb{A}_n)$, and set $W = \{v \in V \mid \delta(v) > l\}$. Then the instance $\mathcal{P}|_W$ belongs to $\mathrm{CSP}(\mathbb{A}_{l+1}, \ldots, \mathbb{A}_n)$, and so is tractable, by assumption. Solve $\mathcal{P}|_W$. If it has no solution then neither does $\mathcal{P}$. Otherwise, let φ be a solution. We claim that the mapping $\psi\colon V \to \mathbb{A}_1 \cup \ldots \cup \mathbb{A}_n$ defined as follows

$$\psi(v) = \begin{cases} c_{\delta(v)}, & \text{if } \delta(v) \le l, \\ t^{\delta(v)}(\varphi(v)), & \text{otherwise} \end{cases}$$

is a solution to $\mathcal{P}$.

For any constraint $C = \langle s, \varrho \rangle \in \mathcal{C}$ there is $\overline{a} \in \varrho$ such that $\overline{a}[v] = \varphi(v)$ for $v \in s \cap W$. Setting $\overline{b} = t(\overline{a}) \in \varrho$ we get $\overline{b}[v] = c_{\delta(v)} = \psi(v)$ if $\delta(v) \le l$, and $\overline{b}[v] = t^{\delta(v)}(\varphi(v)) = \psi(v)$ otherwise. Thus ψ satisfies every constraint, and hence is a solution to $\mathcal{P}$. □