The complexity of maximal constraint languages

Andrei A. Bulatov
Department of Algebra and Discrete Mathematics
Ural State University
620083 Ekaterinburg, Russia
Andrei.Bulatov@usu.ru

Andrei A. Krokhin
Computing Laboratory
Oxford University
Wolfson Building, Parks Road
Oxford OX1 3QD, UK
Andrei.Krokhin@comlab.ox.ac.uk

Peter G. Jeavons
Computing Laboratory
Oxford University
Wolfson Building, Parks Road
Oxford OX1 3QD, UK
Peter.Jeavons@comlab.ox.ac.uk

ABSTRACT

Many combinatorial search problems can be expressed as "constraint satisfaction problems" using an appropriate "constraint language", that is, a set of relations over some fixed finite set of values. It is well-known that there is a trade-off between the expressive power of a constraint language and the complexity of the problems it can express. In the present paper we systematically study the complexity of all maximal constraint languages, that is, languages whose expressive power is just weaker than that of the language of all constraints. Using the algebraic invariance properties of constraints, we exhibit a strong necessary condition for tractability of such a constraint language. Moreover, we show that, at least for small sets of values, this condition is also sufficient.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*computations on discrete structures*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*relation systems*; G.2.1 [**Discrete Mathematics**]: Combinatorics—*combinatorial algorithms*

General Terms

Algorithms, Theory

Keywords

constraint satisfaction problem, constraint language, complexity, tractability, NP-completeness, algebraic invariance property, relational clone

1. INTRODUCTION

A wide range of combinatorial search problems can be naturally expressed as 'constraint satisfaction problems' [18], in which the aim is to find an assignment of values to a given set of variables subject to specified constraints. For example, the standard propositional satisfiability problem [12] may be viewed as a constraint satisfaction problem where the variables must be assigned Boolean values, and the constraints are specified by clauses. Further examples include graph colorability, clique, and Hamiltonian circuit problems, scheduling problems, and many others (see [16]). One advantage of considering a common framework for all of these diverse problems is that it makes it possible to obtain generic structural results concerning the computational complexity of constraint satisfaction problems that can be applied in many different areas such as database theory [28], temporal and spatial reasoning [26], machine vision [18], belief maintenance [6], technical design [20], natural language comprehension [1], and programming language analysis [19].

The general constraint satisfaction problem is NP-complete. There are two main streams in the study of this problem: the search for heuristics (see e.g., [7, 8, 17]) and the investigation of how restricting the allowed constraints affects the complexity of the corresponding problem (see e.g., [3, 4, 10, 11, 14, 16, 25]). In this paper we systematically consider the effect of restricting, in the *weakest possible ways*, the allowed constraints over an arbitrary finite set of possible values. Surprisingly, in some cases even these weakest possible restrictions are sufficicient to ensure tractability. This paper is aimed at separating such restrictions from those which remain NP-complete.

To formalise the idea of restricting the allowed constraints, we make use of the notion of a "constraint language" [15], which is simply a set of possible relations that can be used to specify constraints in a problem. Clearly, the more relations that are included in a constraint language, the more constraints it is able to express. We give a precise definition of this idea of expressive power. Naturally, increasing the expressive power of a constraint language can only make the corresponding problem harder to solve. Imposing a weakest possible proper restriction on the possible constraints gives rise to a "maximal constraint language", which is a language with the property that adding *any* relation which it cannot already express allows it to express every relation. We show that these maximal constraint languages are either NP-complete, or else satisfy a certain algebraic condition. Furthermore, we show that this condition is sufficient to ensure tractability when the set of possible values has two or three elements.

The technique used in this paper relies upon the idea,

STOC'01, July 6-8, 2001, Hersonissos, Crete, Greece.

that was developed in [3, 16] (and also mentioned in [10] as a possible direction of future research), that algebraic invariance properties of constraints can be used for studying the complexity of the corresponding constraint satisfaction problems. The main advantage of this technique is that it allows us to employ structural results from universal algebra, especially from clone theory. We believe that the synthesis between complexity theory and clone theory which we describe here is likely to lead to new results in both fields.

2. PRELIMINARIES

2.1 Constraint languages

Throughout the paper, let D be a finite set. The set of all n-tuples of elements of D is denoted by D^n. Any subset of D^n is called an n-ary *relation* on D. The set of all finitary relations over D is denoted by R_D.

DEFINITION 1. *A constraint language over a finite set, D, is an arbitrary set $\Gamma \subseteq R_D$.*

DEFINITION 2. *The* constraint satisfaction problem *over the constraint language Γ, denoted* CSP(Γ)*, is defined to be the decision problem with instance (V, D, C), where*

- *V is a finite set of variables,*
- *D is a set of values (sometimes called a domain), and*
- *C is a set of constraints $\{C_1, \ldots, C_q\}$,*

  in which each constraint C_i is a pair (s_i, ϱ_i) with s_i a list of variables of length m_i, called the constraint scope*, and ϱ_i an m_i-ary relation over the set D, belonging to Γ, called* the constraint relation.

The question is whether there exists a solution *to (V, D, C), that is, a function from V to D such that, for each constraint in C, the image of the constraint scope is a member of the constraint relation.*

The size of a problem instance is the length of the encoding of all tuples in all constraints. If there exists an algorithm which decides every instance of CSP(Γ) in polynomial time, then we say that Γ is a tractable constraint language. On the other hand, if CSP(Γ) is NP-complete, then we say that Γ is NP-complete.

EXAMPLE 1. *The binary disequality relation, denoted by $\neq_D$, is defined as*

$$\neq_D = \{(d_1, d_2) \in D^2 \mid d_1 \neq d_2\}.$$

Note that CSP($\{\neq_D\}$) *precisely corresponds to the* GRAPH $|D|$-COLORABILITY *problem [12]. Thus the constraint language $\{\neq_D\}$ is tractable when $|D| = 2$ and* NP*-complete when $|D| \geq 3$.*

EXAMPLE 2. *If Γ is the set all binary relations over $\{0,1\}$, then* CSP(Γ) *precisely corresponds to the* 2-SATISFIABILITY *problem, which is tractable [12]. Therefore the constraint language Γ is tractable.*

EXAMPLE 3. *Consider the ternary relation N over $\{0,1\}$ defined by*

$$N = \{(0,0,1), (0,1,0), (1,0,0), (0,1,1), (1,0,1), (1,1,0)\}.$$

The problem CSP($\{N\}$) *precisely corresponds to the* NOT-ALL-EQUAL SATISFIABILITY *problem [12], which is* NP*-complete [25]. Hence the constraint language $\{N\}$ is NP-complete.*

The complexity of constraint languages over a 2-element set has been completely classified by T.J.Schaefer [25]. He showed that there are six maximal tractable constraint languages $\Gamma_1, \ldots, \Gamma_6$ over $\{0,1\}$, and every constraint language over this set either is contained in one of those six (and therefore is tractable), or is NP-complete. These six constraint languages are defined as follows:

- Γ_1 consists of all relations which contain the tuple $(0, \ldots, 0)$;
- Γ_2 consists of all relations which contain the tuple $(1, \ldots, 1)$;
- Γ_3 consists of all relations definable by a formula in conjunctive normal form in which each conjunct has at most one negated variable;
- Γ_4 consists of all relations definable by a formula in conjunctive normal form in which each conjunct has at most one unnegated variable;
- Γ_5 consists of all relations definable by a formula in conjunctive normal form in which each conjunct has at most two literals;
- Γ_6 consists of all relations representable as the set of all solutions of a system of linear equations over the field GF(2).

These languages will appear again in Section 3.2. In [25], Schaefer raised the problem of classifying the complexity of constraint languages over larger finite sets. This problem is still open and is believed to be very hard. In this paper we consider this problem for a particular important class of constraint languages, defined below.

2.2 Expressive power and relational clones

To motivate the definition of expressive power given below (Definition 3), we introduce the idea of implicit constraints. For any problem in CSP(Γ), all of the explicit constraint relations are elements of Γ, by definition; however, there may be *implicit* constraints on some subsets of variables for which the corresponding constraint relations are not in Γ. These implicit constraints arise from the interaction of the explicit constraints specified in a problem instance. To describe implicit constraint relations, we shall make use of a standard correspondence between relations and logical predicates: a relation ϱ consists of all tuples on which the corresponding predicate ψ_ϱ is true. Let $P = (V, D, C)$ be a problem instance from CSP(Γ) where $C = \{(s_1, \varrho_1), \ldots, (s_n, \varrho_n)\}$. The question corresponding to P can be viewed as the question of whether or not the formula $\psi_{\varrho_1}(s_1) \wedge \cdots \wedge \psi_{\varrho_n}(s_n)$ is satisfiable over D. Recall that a primitive positive (pp-) formula is a first order formula using existential quantifiers and conjunction only (such as the formula from Example 4).

EXAMPLE 4. *Reconsider the language $\{N\}$ from Example 3. Let the ternary predicate $\psi_\varrho(x_1, x_2, x_3)$ be defined by the following pp-formula*

$$(\exists x_4)\psi_N(x_1, x_1, x_4) \wedge \psi_N(x_4, x_2, x_4) \wedge \psi_N(x_3, x_1, x_2).$$

Then the corresponding relation ϱ consists of the tuples $(0,0,0)$, $(0,0,1)$, $(1,1,0)$, and $(1,1,1)$. This relation ϱ can occur implicitly as a constraint relation on certain variables in some problem instances from $\mathrm{CSP}(\{N\})$. *In fact, it can be shown that any relation that is invariant under interchanging 0 and 1 can be obtained from N by using an appropriate pp-formula.*

Similarly, all the possible implicit constraint relations in problems from $\mathrm{CSP}(\Gamma)$ correspond to general pp-formulas involving the constraint relations from Γ, such as the one shown in Example 4. Adding the equality relation $=_D$ to a constraint language leaves the expressive capability practically unchanged: it can only add trivial relations and cannot destroy tractablity. However, it does allow us to make use of results and techniques from universal algebra. We therefore make the following definition.

DEFINITION 3. *A relation ϱ can be expressed in a constraint language, Γ, if ϱ can be specified by a pp-formula involving only relations from Γ.*

The expressive power of a constraint language Γ is defined to be the set of all relations that can be expressed in $\Gamma \cup \{=_D\}$; it is denoted by $\langle \Gamma \rangle$.

Our definition of expressive power is similar to that given in [15]. This definition considerably simplifies the analysis of constraint satisfaction problems in view of the following result which links the complexity of constraint languages and their expressive powers.

PROPOSITION 1 ([16]). *For any constraint language Γ, and any finite constraint language Γ', if $\langle \Gamma' \rangle \subseteq \langle \Gamma \rangle$ then $\mathrm{CSP}(\Gamma')$ is reducible to $\mathrm{CSP}(\Gamma)$ in polynomial time*[1].

EXAMPLE 5. *Let ϱ be the following binary relation over $D = \{0,1,\ldots,n\}$, $n \geq 2$.*

$$\varrho = \{(i,i+1) \mid 1 \leq i \leq n-1\} \cup \{(0,i),(i,0) \mid 1 \leq i \leq n\}.$$

It was proved in [9] *that $\langle\{\varrho\}\rangle = R_D$. Then NP-completeness of $\{\varrho\}$ follows from Proposition 1 if we set $\Gamma' = \{\neq_D\}$ and $\Gamma = \{\varrho\}$.*

It is known (see e.g., Theorem 2.3.1 in [21]) that sets of the form $\langle \Gamma \rangle$ are exactly *relational clones*[2], a special type of algebras of finitary relations. We may therefore identify the expressive power of any constraint language with the corresponding relational clone.

In order to obtain an alternative characterisation of relational clones we now introduce some algebraic terminology. By an n-ary operation f we mean a mapping $f\colon D^n \to D$. The set of all operations over D will be denoted by O_D. An operation $f(x_1,\ldots,x_n)$ is called a *projection* if there is i in the range 1 to n such that $f(x_1,\ldots,x_n) = x_i$. We say that an m-ary relation ϱ is *invariant* under an operation $f(x_1,\ldots,x_n)$, or f is a *polymorphism* of ϱ, if, for every $(a_{11},\ldots,a_{m1}),\ldots,(a_{1n},\ldots,a_{mn}) \in \varrho$ we have

$$f\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} = \begin{pmatrix} f(a_{11},a_{12},\ldots,a_{1n}) \\ f(a_{21},a_{22},\ldots,a_{2n}) \\ \vdots \\ f(a_{m1},a_{m2},\ldots,a_{mn}) \end{pmatrix} \in \varrho.$$

The set of all relations which are invariant under each operation from a set $C \subseteq O_D$ is denoted by $\mathrm{Inv}\, C$.

PROPOSITION 2 ([21]).

1. *Relational clones are exactly the sets of the form* $\mathrm{Inv}\, C$ *where $C \subseteq O_D$;*
2. $\mathrm{Inv}\, C = R_D$ *if and only if C consists entirely of projections.*

It follows from this proposition that the expressive power of a constraint language Γ can be characterised by invariance properties of the relations in Γ.

Before proceeding further, we remark on how we use, in this paper, previously known algebraic results. A *clone of operations* is a set $C \subseteq O_D$ that contains all projections and that is closed under superposition, that is, for any n-ary $f \in C$ and any m-ary $g_1,\ldots,g_n \in C$, the operation $f(g_1(x_1,\ldots,x_m),\ldots,g_n(x_1,\ldots,x_m))$ belongs to C as well. For $Q \subseteq R_D$, let $\mathrm{Pol}\, Q$ denote the set of all operations $f \in O_D$ such that f is a polymorphism of every relation from Q. Similarly to Proposition 2(1), clones of operations are exactly the sets of the form $\mathrm{Pol}\, Q$ with $Q \subseteq R_D$. The operators $\mathrm{Inv}\,(-)$ and $\mathrm{Pol}\,(-)$ induce a Galois connection between R_D and O_D (see [2, 13, 21]). This connection yields a reciprocal translation between results about clones of operations and results about relational clones. Using this translation, we shall formulate all necessary algebraic results as results on relational clones, although they were originally proved as results on clones of operations.

2.3 Maximal constraint languages

It is clear that any language whose expressive power is the set of all relations, R_D, is NP-complete (see Example 5). In this subsection we introduce constraint languages whose expressive power is just weaker than this.

DEFINITION 4. *A maximal constraint language is a constraint language with maximal incomplete expressive power, that is, a language, Γ, such that $\langle \Gamma \rangle \neq R_D$, but $\langle \Gamma \cup \{\varrho\} \rangle = R_D$ for every $\varrho \notin \langle \Gamma \rangle$.*

It follows that a constraint language Γ is maximal if and only if $\langle \Gamma \rangle$ is a maximal relational subclone of R_D (in other words, a maximal relational clone). In order to give a characterisation of maximal relational clones we need to define several types of operations.

DEFINITION 5. *Let f be an n-ary operation over D.*

1. *If $f(a,\ldots,a) = a$, for all $a \in D$, then f is said to be* idempotent.

[1] If we omit the equality relation in Definition 3 then, as proved in [16], this would remain true after changing polynomial time to logarithmic space. It is an interesting open question whether the same holds with our definition of expressive power.

[2] The algebraic notion of clone has nothing to do with the biological one. This is a mere coincidence of terms.

2. *If $n = 1$ and f is bijective, then f is said to be a* permutation.

3. *If $n = 2$ and for all $a_1, a_2, a_3 \in D$ we have*

   - $f(f(a_1, a_2), a_3) = f(a_1, f(a_2, a_3))$ *(*associativity*),*
   - $f(a_1, a_2) = f(a_2, a_1)$ *(*commutativity*),*
   - $f(a_1, a_1) = a_1$ *(*idempotency*),*

   then f is said to be a semilattice[3] *operation.*

4. *If $n = 3$ and for all $a, a' \in D$ we have*

   $$f(a, a, a') = f(a, a', a) = f(a', a, a) = a,$$

   then f is said to be a majority *operation.*

5. *If $n = 3$ and for all $a_1, a_2, a_3 \in D$ we have $f(a_1, a_2, a_3) = a_1 - a_2 + a_3$, where $+$ and $-$ are the operations of an Abelian group, then f is said to be* affine.

6. *If $n \geq 3$ and there exists $i \in \{1, \ldots, n\}$ such that for all $a_1, \ldots, a_n \in D$ we have $f(a_1, \ldots, a_n) = a_i$ whenever $|\{a_1, \ldots, a_n\}| < n$, but f is not a projection, then f is said to be a* semiprojection.

To date, there is no complete description of maximal relational clones (or, equivalently, of minimal clones of operations) over an arbitrary finite set. However, the following strong classification theorem follows from a result of I.G.Rosenberg [24].

THEOREM 1 ([24]). *Every maximal relational clone has the form $\mathrm{Inv}\{f\}$ where the operation f is of one of the following types:*

1. *a unary operation which is either a permutation or else acts identically on its range;*
2. *a binary idempotent operation which is not a projection;*
3. *a majority operation;*
4. *an affine operation;*
5. *a semiprojection.*

2.4 Tractable and NP-complete constraint languages

The complexity of some constraint languages over an arbitrary finite set has already been determined.

THEOREM 2 ([3]). *A constraint language $\mathrm{Inv}\{f\}$ is tractable if f is an operation of one of the following types:*

1. *a constant operation,*
2. *a semilattice operation,*
3. *a majority operation,*
4. *an affine operation.*

Let $f(x_1, \ldots, x_n) \in O_D$ and $B \subseteq D$ have the property that $B \in \mathrm{Inv}\{f\}$. Then we denote the restriction of f to B by $f|_B$.

[3]This is the usual algebraic term. However, such an operation has also been called an *ACI* operation (e.g. in [3]).

LEMMA 1. *Let $f(x_1, \ldots, x_n) \in O_D$ and $B \subseteq D$, $|B| > 1$, be such that $B \in \mathrm{Inv}\{f\}$. If there is $1 \leq i \leq n$ and a permutation g on B such that $f|_B(x_1, \ldots, x_n) = g(x_i)$, then any constraint language Γ with $\langle\Gamma\rangle = \mathrm{Inv}\{f\}$ is NP-complete.*

Proof. The relation $\neq_B$ is invariant under all permutations of B, and hence belongs to $\mathrm{Inv}\{f\}$. Then, as follows from Example 1, $\mathrm{Inv}\{f\}$ is NP-complete whenever $|B| > 2$. If $|B| = 2$ then we may suppose without loss of generality that $B = \{0, 1\}$. Then the relation N from Example 3 belongs to $\mathrm{Inv}\{f\}$, and, hence, this set is NP-complete. Now NP-completeness of Γ follows from Proposition 1 if we set Γ' to be $\{\neq_B\}$ or $\{N\}$, respectively. □

COROLLARY 1. *Let Γ be a constraint language such that $\langle\Gamma\rangle = \mathrm{Inv}\{f\}$ where f is either a permutation, or a unary non-constant operation acting identically on its range, or a semiprojection. Then Γ is NP-complete.*

Proof. Follows from Lemma 1 if we set B to be D, or the range of f, or any two-element subset of D, respectively. □

3. THE MAIN RESULT

In this section we prove the main result of this paper.

THEOREM 3. *Let Γ be a maximal constraint language on an arbitrary finite set D.*

(i) Either Γ is NP-complete or $\langle\Gamma\rangle = \mathrm{Inv}\{f\}$ where f is an operation of one of the following types:

   - *a constant operation;*
   - *a binary commutative idempotent operation;*
   - *a majority operation;*
   - *an affine operation.*

(ii) If $\langle\Gamma\rangle = \mathrm{Inv}\{f\}$ for some f of the above form, and $|D| \leq 3$, then Γ is tractable.

Assuming $P \neq NP$, the first part of this theorem provides a necessary condition for tractability of a maximal constraint language, and the second part establishes that this condition is sufficient, at least on small sets of values.

3.1 The necessary condition

It follows from Theorems 1 and 2, and from Corollary 1, that, in order to prove part (i) of Theorem 3 it suffices to consider the case of a binary idempotent operation.

LEMMA 2. *Let $\mathrm{Inv}\{f\}$ be a maximal relational clone on D where f is binary idempotent. Then either $\mathrm{Inv}\{f\} = \mathrm{Inv}\{g\}$ where $g(x, y) = \frac{p+1}{2}(x + y)$ with $(D, +)$ an Abelian group of prime order $p > 2$, or $B \in \mathrm{Inv}\{f\}$ for some two-element $B \subseteq D$.*

Proof. Follows from Theorem 3.2 in [27]. □

LEMMA 3. *Let $\mathrm{Inv}\{f\}$ be a maximal relational clone with f binary idempotent. Then exactly one of the following conditions holds:*

1. $\mathrm{Inv}\{f\} = \mathrm{Inv}\{g\}$ *for some binary commutative idempotent operation* g*, or*

2. *there is a two-element* $B \subseteq D$ *such that* $B \in \mathrm{Inv}\{f\}$ *and* $f|_B$ *is a projection.*

Proof. Suppose, for contradiction, that both conditions hold for some f. Then $\mathrm{Inv}\{f\}$ contains the entire set R_B. If $\mathrm{Inv}\{f\} = \mathrm{Inv}\{g\}$ for some binary idempotent operation g then, as follows from Proposition 2(2), $g|_B$ must be a projection. Hence, g cannot be commutative, a contradiction.

Now we prove that one of the conditions holds. We may assume that $B \in \mathrm{Inv}\{f\}$ for some two-element $B \subseteq D$; otherwise we get the required result by Lemma 2. Then $f|_B$ is an idempotent binary operation on B. There are four idempotent binary operations on a two-element set; two of them are projections and the other two are semilattice operations. If there exists a two-element $B \subseteq D$ such that $f|_B$ is a projection then the result follows.

Suppose now that $f|_B$ is a semilattice operation for every two-element $B \subseteq D$ such that $B \in \mathrm{Inv}\{f\}$. Let us consider the operations $f_n(x,y)$ where $f_1(x,y) = f(x,y)$ and $f_{n+1}(x,y) = f(f_n(x,y), f_n(y,x))$. Using the definition of a semilattice operation it can be verified that $f_n|_B = f|_B$ for every two-element $B \in \mathrm{Inv}\{f\}$. It is easy to prove by induction that $\mathrm{Inv}\{f\} \subseteq \mathrm{Inv}\{f_n\} \subseteq R_D$. By Proposition 2(2), $\mathrm{Inv}\{f_n\} \neq R_D$, since f_n is not a projection. Therefore, for every n, we have $\mathrm{Inv}\{f\} = \mathrm{Inv}\{f_n\}$, because $\mathrm{Inv}\{f\}$ is maximal. It remains to show that, for some n, the operation f_n is commutative.

Consider a digraph $G = (V, A)$ with $V = D^2$ and $A = \{\langle (a,b), (f(a,b), f(b,a))\rangle \mid (a,b) \in D^2\}$. Suppose G has a simple directed cycle $(a_1,b_1), \ldots, (a_l,b_l), (a_1,b_1)$ where $l \geq 2$. Then $a_1 \neq b_1$ because the only arc outcoming from (a_1,a_1) is a loop. We have $f_l(a_1,b_1) = a_1$ and $f_l(b_1,a_1) = b_1$, that is, f_l is a projection on $\{a_1,b_1\}$. Since $\{a_1,b_1\} \in \mathrm{Inv}\{f_l\} = \mathrm{Inv}\{f\}$ and $f_l|_B = f|_B$ for every two-element $B \in \mathrm{Inv}\{f\}$, we get that $f|_{\{a_1,b_1\}}$ is a projection, which contradicts our assumption that $f|_B$ is a semilattice operation on every two-element $B \in \mathrm{Inv}\{f\}$. Hence, G has no directed cycles. If G has a loop on a node (a,b) such that $a \neq b$ then we have $f(a,b) = a$ and $f(b,a) = b$, a contradiction again. Therefore, there exists a number n such that any directed path of length n in G terminates at a node of the form (a,a). Now it follows that $f_n(x,y) = f_n(y,x)$ for every $x,y \in D$. Finally, it is clear that f_n is idempotent, so the result follows. □

Using Lemma 1, that completes the proof of Part (i) of Theorem 3.

3.2 The sufficient condition: easy cases

Let $D = \{0,1\}$. Then, on D, there are 7 operations, f, of the types indicated in Theorem 1: the two constants, 0 and 1, the operations of negation $\neg x$, disjunction $x \vee y$ and conjuction $x \wedge y$, the unique majority operation $(x \vee y) \wedge (y \vee z) \wedge (z \vee x)$, and the unique affine operation $x - y + z$; there exist no semiprojections on a two-element set. It follows from [22] that the seven corresponding relational clones $\mathrm{Inv}\{f\}$ are the maximal relational clones on $\{0,1\}$. Since $\vee$ and $\wedge$ are semilattice operations, Theorem 3(ii) for the case $|D| = 2$ now follows from Theorem 2 and Corollary 1. Interestingly, the six tractable maximal relational clones on $\{0,1\}$ are precisely the six maximal tractable constraint languages $\Gamma_1, \ldots, \Gamma_6$ over $\{0,1\}$ described in [25] (see Section 2.1).

$*$	0	1	2
0	0	0	0
1	0	1	0
2	0	0	2

(0)

$*$	0	1	2
0	0	0	0
1	0	1	0
2	2	2	2

(8)

$*$	0	1	2
0	0	0	0
1	0	1	1
2	0	1	2

(10)

$*$	0	1	2
0	0	0	0
1	0	1	1
2	0	2	2

(11)

$*$	0	1	2
0	0	0	0
1	0	1	1
2	2	1	2

(16)

$*$	0	1	2
0	0	0	0
1	0	1	1
2	2	2	2

(17)

$*$	0	1	2
0	0	0	0
1	0	1	2
2	2	2	2

(26)

$*$	0	1	2
0	0	0	0
1	1	1	0
2	2	0	2

(33)

$*$	0	1	2
0	0	0	0
1	1	1	0
2	2	2	2

(35)

$*$	0	1	2
0	0	0	0
1	2	1	1
2	1	2	2

(68)

$*$	0	1	2
0	0	0	2
1	0	1	1
2	2	1	2

(178)

$*$	0	1	2
0	0	2	1
1	2	1	0
2	1	0	2

(624)

Table 1: Binary operations $*$ on $\{0,1,2\}$ such that $\mathrm{Inv}\{*\}$ is maximal.

From now on we assume that $D = \{0,1,2\}$. It follows from Theorem 2 that we only need to consider the case of a binary idempotent operation. It will be convenient to denote this operation by $*$, and we will write $x * y$ instead of $*(x,y)$. All binary idempotent operations $*$ on $\{0,1,2\}$ such that $\mathrm{Inv}\{*\}$ is maximal have been described by B.Csákány in [5]. Up to permutations of $\{0,1,2\}$ and transposition of the Cayley table, there are 12 such operations given in Table 1 by their Cayley tables. (The number given under each Cayley table is the number given to the operation in Csákány's classification [5].)

The operations (8), (16), (17), (26), (33) and (35), and the operations (11) and (68) are projections when restricted to $B = \{0,2\}$ and $B = \{1,2\}$, respectively. As follows from Lemma 3, the corresponding relational clones cannot be represented as $\mathrm{Inv}\{f\}$ where f is a binary commutative idempotent operation, that is they are not of the form we need to consider here.

It is easy to check that the operations (0) and (10) are semilattice operations. The operation (624) can be represented in the form $2x + 2y$ where $+$ is addition modulo 3, which implies that the relations that are invariant under this operation are also invariant under the affine operation $x - y + z = 2(2x + 2z) + 2y (\mathrm{mod}\ 3)$. Hence, by Theorem 2, the operations (0), (10), and (624) give rise to tractable maximal constraint languages.

So, the only case left to study is the operation (178). Since this operation is commutative, we have to prove that the corresponding relational clone is tractable. It is, in fact, a new type of tractable relational clone, which is not covered by any earlier results.

3.3 The operation (178)

From now on, by $*$ we mean the operation (178) defined above.

In order to construct an algorithm solving CSP(Inv $\{*\}$) we shall first clarify the structure of the relations in Inv $\{*\}$. The operation $*$ will also be considered as acting on D^n component-wise. We shall use the following lemma which can be straightforwardly verified.

LEMMA 4. *For every $n \in \mathbb{N}$, and for every $\overline{a}, \overline{b} \in D^n$ there exist $\overline{x}, \overline{y} \in D^n$ such that $(\overline{a} * \overline{x}) * \overline{y} = \overline{b}$.*

For any $\overline{a} = (a_1, \dots, a_n) \in D^n$, and any subsequence $i_1, \dots, i_k$ of the sequence $1, \dots, n$, we define $\mathrm{pr}_{i_1,\dots,i_k}\overline{a}$ to be the k-tuple $(a_{i_1}, \dots, a_{i_k})$. Moreover, for any n-ary relation ϱ, we define $\mathrm{pr}_{i_1,\dots,i_k}\varrho$ to be the k-ary relation

$$\mathrm{pr}_{i_1,\dots,i_k}\varrho = \{\mathrm{pr}_{i_1,\dots,i_k}\overline{a} \mid \overline{a} = (a_1, \dots, a_n) \in \varrho\}.$$

For $I = i_1, \dots, i_k$, we will sometimes write $\mathrm{pr}_I\varrho$ instead of $\mathrm{pr}_{i_1,\dots,i_k}\varrho$.

Let $\underline{n}$ denote the set $\{1, \dots, n\}$. Suppose $I_1, \dots, I_k$ is a partition of $\underline{n}$ and let $\varrho_j = \mathrm{pr}_{I_j}\varrho$ for $j = 1, \dots, k$. Then we write $\varrho = \varrho_1 \times \dots \times \varrho_k$ if ϱ can be represented as

$$\varrho = \{\overline{a} \mid \mathrm{pr}_{I_j}\overline{a} \in \varrho_j \text{ for every } j = 1, \dots, k\}.$$

Finally, the graph of a permutation π is the binary relation $\{(x, y) \mid y = \pi(x)\}$.

LEMMA 5. *Let $\varrho \in \mathrm{Inv}\{*\}$ be an n-ary relation such that $\mathrm{pr}_i\varrho = D$ for every $i \in \underline{n}$. Then ϱ is of the form*

$$\varrho = \varrho_1 \times \dots \times \varrho_k$$

where $\varrho_j = \{(a, \pi_{2j}(a), \dots, \pi_{l_j j}(a)) \mid a \in D\}$, for some permutations $\pi_{2j}, \dots, \pi_{l_j j}$ of D.

Proof. We prove the lemma by induction on n. The induction base, the case $n = 2$, can be proved by straightforward calculation. Let us prove the induction step. By the induction base, for any pair $i, j \in \underline{n}$ the projection $\mathrm{pr}_{i,j}\varrho$ is either D^2, or the graph of a permutation. Assume that there exist i, j such that $\mathrm{pr}_{i,j}\varrho$ is the graph of a permutation π. By the inductive hypothesis $\mathrm{pr}_{\underline{n}\setminus\{j\}}\varrho$ can be represented in the form

$$\mathrm{pr}_{\underline{n}\setminus\{j\}}\varrho = \varrho_1 \times \dots \times \varrho_k,$$

and the i-th coordinate position occurs in one of $\varrho_1, \dots, \varrho_k$. Suppose i is a coordinate position in ϱ_1, that is

$$\begin{aligned}\varrho_1 \quad = \quad & \{(a_{i_1}, \dots, a_{i_{l_1-1}}, a_i) \mid a_{i_1} \in D,\ a_{i_s} = \pi_{s1}(a_{i_1}) \\ & \text{for } s \in \{2, \dots, l_1 - 1\},\ a_i = \pi_i(a_{i_1})\}.\end{aligned}$$

Then letting

$$\begin{aligned}\varrho_1' \quad = \quad & \{(a_{i_1}, \dots, a_{i_{l_1-1}}, a_i, a_j) \mid a_{i_1} \in D,\ a_{i_s} = \pi_{s1}(a_{i_1}) \\ & \text{for } s \in \{2, \dots, l_1 - 1\},\ a_i = \pi_i(a_{i_1}), a_j = \pi\pi_i(a_{i_1})\}\end{aligned}$$

we have $\varrho = \varrho_1' \times \varrho_2 \times \dots \times \varrho_k$, as required.

It remains to prove that if $\mathrm{pr}_{i,j}\varrho = D^2$ for every $i, j \in \underline{n}$, then $\varrho = D^n$. Suppose this holds for each $k < n$. For $a \in D$, set $\varrho_a = \{(a_1, \dots, a_{n-1}) \mid (a_1, \dots, a_{n-1}, a) \in \varrho\}$. Since the operation $*$ is idempotent, ϱ_a also belongs to Inv $\{*\}$. From now on it will be convenient to treat elements of D^n as columns. We write $\begin{pmatrix} \overline{a} \\ \overline{b} \end{pmatrix}$ instead of $(a_1, \dots, a_n, b_1, \dots, b_m)^T$ (transposed vector) if $\overline{a} = (a_1, \dots, a_n)^T$, $\overline{b} = (b_1, \dots, b_m)^T$ are columns.

Consider the case $n = 3$. Suppose that, for some a, say $a = 0$, the relation ϱ_0 is not the graph of a permutation. Then $\varrho_0 = D^2$ by the inductive hypothesis. Since $1, 2 \in \mathrm{pr}_3\varrho$, there are $\overline{a}_1, \overline{a}_2 \in D^2$ such that $\begin{pmatrix} \overline{a}_1 \\ 1 \end{pmatrix} \in \varrho$ and $\begin{pmatrix} \overline{a}_2 \\ 2 \end{pmatrix} \in \varrho$. By Lemma 4, for any $\overline{b} \in D^2$, there are $\overline{x}_1, \overline{y}_1, \overline{x}_2, \overline{y}_2 \in D^2 = \varrho_0$ such that

$$\left(\begin{pmatrix} \overline{a}_2 \\ 2 \end{pmatrix} * \begin{pmatrix} \overline{x}_2 \\ 0 \end{pmatrix}\right) * \begin{pmatrix} \overline{y}_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \overline{b} \\ 2 \end{pmatrix},$$

$$\left(\begin{pmatrix} \overline{a}_1 \\ 1 \end{pmatrix} * \begin{pmatrix} \overline{x}_1 \\ 2 \end{pmatrix}\right) * \begin{pmatrix} \overline{y}_1 \\ 2 \end{pmatrix} = \begin{pmatrix} \overline{b} \\ 1 \end{pmatrix};$$

therefore $\varrho = D^3$.

Let ϱ_a be a permutation for each $a \in \{0, 1, 2\}$. Since $|\varrho_a| = 3$ for any a, and since $|\mathrm{pr}_{1,2}\varrho| = |\varrho_0 \cup \varrho_1 \cup \varrho_2| = |D^2| = 9$, we have $\varrho_a \cap \varrho_b = \varnothing$ for all $a, b \in \{0, 1, 2\}$ such that $a \neq b$. Taking $a, b \in D$ such that $\begin{pmatrix} a \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} b \\ 0 \\ 1 \end{pmatrix} \in \varrho$ we have $\begin{pmatrix} a \\ 1 \\ 0 \end{pmatrix} * \begin{pmatrix} b \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} c \\ 0 \\ 0 \end{pmatrix}$ where $c \in \{a, b\}$. On the one hand $c \neq b$, since $\begin{pmatrix} b \\ 0 \end{pmatrix} \in \varrho_1$, $\begin{pmatrix} c \\ 0 \end{pmatrix} \in \varrho_0$ and $\varrho_0 \cap \varrho_1 = \varnothing$. On the other hand $c \neq a$, since ϱ_0 is the graph of a permutation. A contradiction.

If $n > 3$ then, by the inductive hypothesis, we have $\mathrm{pr}_{i,j,n}\varrho = D^3$ and, consequently, $\mathrm{pr}_{i,j}\varrho_a = D^2$ holds for all $1 \leq i, j \leq n - 1$. Applying the inductive hypothesis to ϱ_a we obtain $\varrho_a = D^{n-1}$ for each $a \in \{0, 1, 2\}$, which means that $\varrho = D^n$. $\square$

Note that the restriction of $*$ to every two-element subset of $\{0, 1, 2\}$ is a semilattice operation. Suppose $\varrho \in \mathrm{Inv}\{*\}$ is an n-ary relation such that $|\mathrm{pr}_i\varrho| \leq 2$ for all $i \in \underline{n}$. One can define a partial order on ϱ by the following rule: $\overline{a} \leq \overline{b}$ if and only if $\overline{a} * \overline{b} = \overline{b}$. Then it is not hard to see that the 'product' of all elements from ϱ will be the greatest element of ϱ under this order. This element will be denoted by $\max\varrho$.

LEMMA 6. *Let $\varrho \in \mathrm{Inv}\{*\}$ be an n-ary relation and $I = \{i \in \underline{n} \mid |\mathrm{pr}_i\varrho| \leq 2\}$. Then*

$$\{\max \mathrm{pr}_I\varrho\} \times \mathrm{pr}_{\underline{n}\setminus I}\varrho \subseteq \varrho$$

Proof. Without loss of generality, let $I = \{1, \dots, l\}$ and let $\overline{m} = \max \mathrm{pr}_I\varrho$. By Lemma 5 $\mathrm{pr}_{\underline{n}\setminus I}\varrho = \varrho_1 \times \dots \times \varrho_k$ where $\varrho_j = \{(a, \pi_{2j}(a), \dots, \pi_{l_j j}(a)) \mid a \in D\}$ and $\pi_{2j}, \dots, \pi_{l_j j}$ are permutations of D. Let us denote the set of coordinate indices of ϱ_j by J_j, and let J_0 be a system of representatives of $J_1, \dots, J_k$. Then $\mathrm{pr}_{J_0}\varrho = D^{|J_0|}$. Choose $\overline{a} \in \mathrm{pr}_{J_0}\varrho$ so that $\begin{pmatrix} \overline{m} \\ \overline{a} \end{pmatrix} \in \mathrm{pr}_{I\cup J_0}\varrho$. By Lemma 4, for every $\overline{b} \in \mathrm{pr}_{J_0}\varrho$, there are $\overline{x}, \overline{y} \in \mathrm{pr}_{J_0}\varrho$ such that $(\overline{a} * \overline{x}) * \overline{y} = \overline{b}$. Suppose

$\begin{pmatrix} \overline{c}_1 \\ \overline{x} \end{pmatrix}, \begin{pmatrix} \overline{c}_2 \\ \overline{y} \end{pmatrix} \in \mathrm{pr}_{I\cup J_0}\varrho$. Then

$$\left(\begin{pmatrix} \overline{m} \\ \overline{a} \end{pmatrix} * \begin{pmatrix} \overline{c}_1 \\ \overline{x} \end{pmatrix}\right) * \begin{pmatrix} \overline{c}_2 \\ \overline{y} \end{pmatrix} = \begin{pmatrix} \overline{m} \\ \overline{b} \end{pmatrix} \in \mathrm{pr}_{I\cup J_0}\varrho.$$

By the choice of J_0, any element from $\mathrm{pr}_{I\cup J_0}\varrho$ has a unique extension to an element of ϱ. □

Now we are in a position to describe a polynomial-time algorithm for solving CSP(Inv $\{*\}$).

Consider an instance $P = (V, D, \{(s_1, \varrho_1), \dots, (s_n, \varrho_n)\})$ where $\varrho_1, \dots, \varrho_n$ are in Inv $\{*\}$. Suppose that some variable l occurs in s_1 and in s_2. If φ is a solution to P then, by definition $\varphi(l)$ must belong to both $\mathrm{pr}_l\varrho_1$ and $\mathrm{pr}_l\varrho_2$. We can therefore remove from both ϱ_1 and ϱ_2 all the tuples whose l-th coordinate is not in $\mathrm{pr}_l\varrho_1 \cap \mathrm{pr}_l\varrho_2$, and this procedure does not change the set of solutions. Similarly, if l_1 and l_2 belong to both s_1 and s_2 then we can remove the tuples $\overline{a}$ such that $\mathrm{pr}_{l_1,l_2}\overline{a} \notin \mathrm{pr}_{l_1,l_2}\varrho_1 \cap \mathrm{pr}_{l_1,l_2}\varrho_2$. Suppose that ϱ'_1 and ϱ'_2 are the relations obtained from ϱ_1 and ϱ_2 by performing this procedure. Then it is easy to see that there are two pp-formulas that deliver ϱ'_i from ϱ_1 and ϱ_2, that is we have $\varrho'_1, \varrho'_2 \in$ Inv $\{*\}$.

Let $P' = (V, D, \{(s_1, \varrho'_1), \dots, (s_n, \varrho'_n)\})$ be a problem instance obtained from P by performing the above procedure for all coordinates, all pairs of coordinates, and all constraints until the process is stable. This can be done in $O(|V|^2 \cdot k^2)$ time where $k = |\varrho_1| + \dots + |\varrho_n|$. If some constraint relation ϱ'_i is empty then P has no solution. Recall that we have $\varrho'_i \in$ Inv $\{*\}$ for all $i = 1, \dots, n$.

Let $A_l = \mathrm{pr}_l\varrho'_j$, $A_{l_1,l_2} = \mathrm{pr}_{l_1,l_2}\varrho'_j$ whenever $l, l_1, l_2 \in s_j$. By construction, A_l's and A_{l_1,l_2}'s do not depend on j. Consider the graph $G = (J, E)$ where $J = \{l \in V \mid A_l = D\}$ and $E = \{(l_1, l_2) \in J^2 \mid A_{l_1,l_2}$ is the graph of a permutation$\}$. By Lemma 5, the constraints in P' imply that, in every connected component K of G, any value of any variable uniquely determines the values for all other variables in K, while the variables from different connected components are unrelated. It can therefore be checked in $O(|J|^2)$ time whether there exists a function $\phi : J \to D$ such that $(\phi(l_1), \phi(l_2)) \in A_{l_1,l_2}$ for every $(l_1, l_2) \in E$. If such ϕ does not exist then, evidently, P' (as well as P) has no solution. Otherwise, ϕ is a 'partial' solution to P', which, as follows from Lemma 6, can be extended to a full solution to P' by setting, for every $l \in V \setminus J$, $\phi(l)$ to be a if $A_l = \{a\}$ and $a * b$ if $A_l = \{a, b\}$.

This completes the proof of Theorem 3.

4. CONCLUSION

Throughout this paper we have made use of the exact correspondence between the expressive power of a constraint language and the algebraic notion of a relational clone. This has allowed us to make considerable progress in the study of maximal constraint languages, and in developing general methods to determine the complexity of the corresponding problems.

Finding a complete description of maximal relational clones (or, equivalently, of minimal clones of operations) on an arbitrary finite set is a long-standing open problem in universal algebra, and is known to be very difficult [23]. However, the results of this paper lead us to conjecture that the conditions specified in Theorem 3 provide a complete description of tractability in the class of all maximal constraint languages. Thus, there appears to be an interesting interplay between ideas of computational complexity and structural questions in universal algebra, which deserves to be further explored.

5. ACKNOWLEDGEMENTS

This research is partially supported by the UK EPSRC grants GR/R22704 and GR/M12926.

6. REFERENCES

[1] J. F. Allen. *Natural Language Understanding.* Benjamin Cummings, 2 edition, 1994.

[2] V. G. Bodnarchuk, L. A. Kalužnin, V. N. Kotov, and B. A. Romov. Galois theory for Post algebras. I, II. *Kibernetika (Kiev)*, 3:1–10, 1969. ibid. 5:1–9, 1969.

[3] D. Cohen, M. Gyssens, and P. Jeavons. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

[4] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications.* SIAM, 2001.

[5] B. Csákány. Three-element groupoids with minimal clones. *Acta Sci. Math. (Szeged)*, 45(1-4):111–117, 1983.

[6] R. Dechter and A. Dechter. Structure-driven algorithms for truth maintenance. *Artificial Intelligence*, 82(1-2):1–20, 1996.

[7] R. Dechter and I. Meiri. Experimental evaluation of preprocessing algorithms for constraint satisfaction problems. *Artificial Intelligence*, 68(2):211–241, 1994.

[8] R. Dechter and J. Pearl. Network-based heuristics for constraint satisfaction problems. *Artificial Intelligence*, 34(1):111–117, 1987.

[9] A. Fearnley. A strongly rigid binary relation. *Acta Sci. Math. (Szeged)*, 61(1-4):35–41, 1995.

[10] 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(1):57–104, 1999.

[11] E. C. Freuder. Complexity of k-tree structured constraint satisfaction problems. In *Proceedings of the 8th National Conference on Artificial Intelligence (AAAI'90)*, pages 4–9, 1990.

[12] M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* Freeman, New York, 1979.

[13] D. Geiger. Closed systems of functions and predicates. *Pacific Journal of Mathematics*, 27:95–100, 1968.

[14] G. Gottlob, L. Leone, and F. Scarcello. A comparison of structural CSP decomposition methods. *Artificial Intelligence*, 124(2):243–282, 2000.

[15] P. Jeavons. Constructing constraints. In *Proceedings of the 4th Conference on Constraint Programming (CP'98)*, volume 1520 of *Lecture Notes in Computer Science*, pages 2–17. Springer-Verlag, 1998.

[16] P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1-2):185–204, 1998.

[17] V. Kumar. Algorithms for constraint satisfaction problems: a survey. *AI Magazine*, 13(1):32–44, 1992.

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

[19] B. A. Nadel. Constraint satisfaction in Prolog: Complexity and theory-based heuristics. *Information Sciences*, 83(3-4):113–131, 1995.

[20] B. A. Nadel and J. Lin. Automobile transmission design as a constraint satisfaction problem: Modeling the kinematic level. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM)*, 5(3):137–171, 1991.

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

[22] E. Post. *Two-valued iterative systems of mathematical logic*, volume 5 of *Ann. Math. Studies*. Princeton University Press, 1941.

[23] R. W. Quackenbush. A survey of minimal clones. *Aequationes Math.*, 50(1-2):3–16, 1995.

[24] I. G. Rosenberg. Minimal clones I: the five types. In L. Szabó and A. Szendrei, editors, *Lectures in Universal Algebra*, volume 43 of *Colloq. Math. Soc. Janos Bolyai*, pages 405–427. North-Holland, 1986.

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

[26] E. Schwalb and L. Vila. Temporal constraints: a survey. *Constraints*, 3(2-3):129–149, 1998.

[27] A. Szendrei. Idempotent algebras with restrictions on subalgebras. *Acta Sci. Math. (Szeged)*, 51(1-2):251–268, 1987.

[28] M. Y. Vardi. Constraint satisfaction and database theory: a tutorial. In *Proceedings of 19th ACM Symposium on Principles of Database Systems (PODS'00)*, 2000.