Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Binarisation via Dualisation for Valued Constraints

David A. Cohen
Royal Holloway
University of London
dave@cs.rhul.ac.uk

Martin C. Cooper
IRIT
University of Toulouse III
cooper@irit.fr

Peter G. Jeavons, Stanislav Živný*
University of Oxford
{peter.jeavons,standa.zivny}
@cs.ox.ac.uk

Abstract

Constraint programming is a natural paradigm for many combinatorial optimisation problems. The complexity of constraint satisfaction for various forms of constraints has been widely-studied, both to inform the choice of appropriate algorithms, and to understand better the boundary between polynomial-time complexity and NP-hardness.

In constraint programming it is well-known that any constraint satisfaction problem can be converted to an equivalent binary problem using the so-called dual encoding. Using this standard approach any fixed collection of constraints, of arbitrary arity, can be converted to an equivalent set of constraints of arity at most two.

Here we show that this transformation, although it changes the domain of the constraints, preserves all the relevant algebraic properties that determine the complexity. Moreover, we show that the dual encoding preserves many of the key algorithmic properties of the original instance. We also show that this remains true for more general valued constraint languages, where constraints may assign different cost values to different assignments. Hence, we obtain a simple proof of the fact that to classify the computational complexity of *all* valued constraint languages it suffices to classify only *binary* valued constraint languages.

Introduction

There are two well-known methods for transforming a non-binary constraint satisfaction problem (CSP) into a binary one; the *dual encoding* (Dechter and Pearl 1989) and the *hidden variable encoding* (Rossi, Dahr, and Petrie 1990). Both encode the non-binary constraints to variables that have as domains of possible values the valid tuples of the constraints. That is, the techniques derive a binary encoding of a non-binary constraint by changing the domain of the variables to an extensional representation of the original constraints. A combination of these two encodings, known as the double encoding, has also been studied (Smith, Stergiou, and Walsh 2000). It was observed in (Larrosa and Dechter 2000) that both of these standard encodings can be extended to soft constraints.

In this paper we focus on the dual encoding for the valued constraint satisfaction problem (VCSP). In particular, we consider the effect of this encoding on the set of weighted relations used to express the valued constraints.

Different subproblems of the VCSP can be obtained by restricting, in various ways, the set of weighted relations that can be used to express the constraints. Such a set of weighted relations is generally called a valued constraint language (Cohen et al. 2013; Jeavons, Krokhin, and Živný 2014). For any such valued constraint language Γ there is a corresponding problem VCSP(Γ), and it has been shown that the computational complexity of VCSP(Γ) is determined by certain algebraic properties of the set Γ known as weighted polymorphisms (Cohen et al. 2013).

Here we show that the dual encoding preserves many aspects of these algebraic properties. We show that there is a one-to-one correspondence between the weighted polymorphisms of the original valued constraint language and the weighted polymorphisms of the binary language obtained by the dual encoding. Hence, as well as providing a way to convert any given instance of the VCSP to an equivalent binary instance, the dual encoding also provides a way to convert any valued constraint language to a binary language with essentially the same algebraic properties, and hence essentially the same complexity and algorithmic properties.

Related work One special case of our results is the case when all the weighted relations in the valued constraint language we are considering are 0-weighted. This special case corresponds to the standard constraint satisfaction problem where constraints are specified by relations.

In this special case, (Feder and Vardi 1998) and (Atserias 2008) showed that for any constraint language Γ there exists a constraint language Γ' containing a *single* binary relation such that CSP(Γ) and CSP(Γ') are polynomial-time equivalent. Recently, (Bulín et al. 2013; 2014) have given a different construction that gives the same result but can be shown to preserve various types of identities involving the polymorphisms of Γ. Building on this approach, (Powell and Krokhin 2014) have shown that essentially the same construction as in (Bulín et al. 2013; 2014) extends to the VCSP; in this case the valued constraint

*Stanislav Živný was supported by a Royal Society University Research Fellowship.

language Γ' contains a single unary weighted relation and a single binary relation.

Our construction based on the standard dual encoding is considerably simpler. On the other hand, in our case the resulting Γ' contains a single unary weighted relation, and *more than one* binary relation (in general). However, all the binary relations that we include in Γ' are of the same type and correspond to enforcing equality on the shared variables between different constraints in instances of VCSP(Γ).

Moreover, by allowing Γ' to contain more than one binary relation we are able to preserve all *identities* involving polymorphisms of Γ, where an identity is an equality between arbitrary expressions involving only polymorphisms and all variables are universally quantified over the domain D. This is in contrast with the reduction (for the CSP) described in (Bulín et al. 2013; 2014) and its extension (to the VCSP) described in (Powell and Krokhin 2014), which does not preserve the identities defining Maltsev polymorphisms. In fact it is impossible for any reduction to a single binary relation to preserve such identities, without changing the algorithmic nature of the problem, because it has been shown that any *single binary relation* that has a Maltsev polymorphism also has a majority polymorphism (Kazda 2011).

A similar transformation from constraint languages of arbitrary arity to sets of unary and binary relations was implicitly used in (Barto 2013), for the special case of the CSP.

In a related but different direction, (Cohen, Jeavons, and Živný 2008) studied which valued constraint languages can be transformed to binary valued constraint languages *over the same domain*. It was shown in (Živný, Cohen, and Jeavons 2009) that there are submodular valued constraint languages which *cannot* be expressed (using min and sum) by binary submodular languages over the same domain.

The VCSP

Throughout the paper, let D be a fixed finite set and let $\overline{\mathbb{Q}} = \mathbb{Q} \cup \{\infty\}$ denote the set of rational numbers with infinity.

Definition 1. *An m-ary* weighted relation *over D is any mapping $\phi : D^m \to \overline{\mathbb{Q}}$. We denote by $\mathbf{\Phi}_D^{(m)}$ the set of all m-ary weighted relations and let $\mathbf{\Phi}_D = \bigcup_{m \geq 1} \mathbf{\Phi}_D^{(m)}$.*

We call D the *domain*, the elements of D *labels* (for variables), and we say that the weighted relations in $\mathbf{\Phi}_D$ take *values* (which are elements of $\overline{\mathbb{Q}}$).

It is convenient to highlight the special case when the values taken by a weighted relation are restricted to 0 and ∞.

Definition 2. *Any mapping $\phi : D^m \to \{0, \infty\}$ will be called a* 0-weighted relation *(or simply a* relation*) and will often be identified with the set $\{\mathbf{x} \in D^m \mid \phi(\mathbf{x}) = 0\}$.*

We denote by $\mathrm{Feas}(\phi) = \{\mathbf{x} \in D^m \mid \phi(\mathbf{x}) < \infty\}$ the underlying *feasibility relation* of a given m-ary weighted relation. A weighted relation $\phi : D^m \to \overline{\mathbb{Q}}$ is called *finite-valued* if $\mathrm{Feas}(\phi) = D^m$.

Definition 3. *Let $V = \{x_1, \ldots, x_n\}$ be a set of variables. A* valued constraint *over V is an expression of the form $\phi(\mathbf{x})$ where $\phi \in \mathbf{\Phi}_D^{(m)}$ and $\mathbf{x} \in V^m$, for some positive integer m. The integer m is called the* arity *of the constraint, the tuple $\mathbf{x}$ is called the* scope *of the constraint, and the weighted relation ϕ is called the* constraint weighted relation*.*

Definition 4. *An instance of the* valued constraint satisfaction problem *(VCSP) is specified by a finite set $V = \{x_1, \ldots, x_n\}$ of variables, a finite set D of labels, and an* objective function Φ *expressed as follows:*

$$\Phi(x_1, \ldots, x_n) = \sum_{i=1}^{q} \phi_i(\mathbf{x}_i) \tag{1}$$

where each $\phi_i(\mathbf{x}_i)$, $1 \leq i \leq q$, is a valued constraint over V. Each constraint can appear multiple times in Φ.

The goal is to find an assignment *of labels (a* labelling*) to the variables that minimises Φ.*

Definition 5. *Any set $\Gamma \subseteq \mathbf{\Phi}_D$ of weighted relations on some fixed domain D is called a* valued constraint language*, or simply a* language*.*

We will denote by VCSP(Γ) the class of all VCSP instances in which the constraint weighted relations are all contained in Γ.

The classical constraint satisfaction problem (CSP) (Dechter 2003) can be seen as a special case of the VCSP in which all weighted relations are in fact simply relations, (i.e., 0-weighted relations). A language containing only 0-weighted relations is called *crisp*.

We will make use of the following simple but useful observation about arbitrary finite languages.

Proposition 1. *For any valued constraint language Γ such that $|\Gamma|$ is finite, there is a valued constraint language Γ' with $|\Gamma'| = 1$, such that VCSP(Γ) and VCSP(Γ') are polynomial-time equivalent.*

Proof. Let Γ consist of q weighted relations, $\phi_1, \ldots, \phi_q$, with arities $m_1, \ldots, m_q$, respectively. Without loss of generality, we assume that none of the ϕ_i are the constant function ∞. Let $m = \sum_{i=1}^{q} m_i$. Define the weighted relation ϕ_Γ, with arity m, by setting $\phi_\Gamma(x_1, \ldots, x_m) = \phi_1(x_1, \ldots, x_{m_1}) + \phi_2(x_{m_1+1}, \ldots, x_{m_1+m_2}) + \ldots + \phi_q(x_{m-m_q+1}, \ldots, x_m)$, and set $\Gamma' = \{\phi_\Gamma\}$.

Any instance $\mathcal{P}'$ of VCSP(Γ') can clearly be expressed as an instance of VCSP(Γ). Conversely, for any instance $\mathcal{P}$ of VCSP(Γ) we can obtain an equivalent instance $\mathcal{P}'$ of VCSP(Γ') by simply adding irrelevant variables to the scope of each constraint $\phi_i(\mathbf{x})$, which are constrained by the elements of $\Gamma \setminus \{\phi_i\}$, and then minimising over these. The assignments that minimise the objective function of $\mathcal{P}$ can then be obtained by taking the assignments that minimise the objective function of $\mathcal{P}'$ and restricting them to the variables of $\mathcal{P}$. Hence we have shown that VCSP(Γ) and VCSP(Γ') are polynomial-time equivalent. □

From a language Γ to a binary language Γ_d

A language Γ is called *binary* if all weighted relations from Γ are of arity at most two. The goal of this paper is to study a certain type of transformation from an arbitrary finite language Γ to a *binary* language Γ_d such that VCSP(Γ) and VCSP(Γ_d) are polynomial-time equivalent.

For any m-tuple $\mathbf{x} \in D^m$ we will write $\mathbf{x}[i]$ for its ith component.

Definition 6. *Let Γ be any valued constraint language over D, and let ϕ_Γ be the corresponding weighted relation, of arity m, as defined in the proof of Proposition 1.*

The dual *of Γ, denoted Γ_d, is the binary valued constraint language with domain $D' = \mathrm{Feas}(\phi_\Gamma) \subseteq D^m$, defined by*

$$\Gamma_d \;=\; \{\phi'_\Gamma\} \;\bigcup_{i,j\in\{1,\ldots,m\}}\; \{match_{i,j}\}\,,$$

where $\phi'_\Gamma : D' \to \overline{\mathbb{Q}}$ is the unary weighted relation on D' defined by $\phi'_\Gamma(\mathbf{x}) = \phi_\Gamma(x_1, x_2, \ldots, x_m)$ for every $\mathbf{x} = (x_1, x_2, \ldots, x_m) \in D'$, and each $match_{i,j} : D' \times D' \to \overline{\mathbb{Q}}$ is the binary 0-weighted relation on D' defined by

$$match_{i,j}(\mathbf{x}, \mathbf{y}) \;=\; \begin{cases} 0 & \text{if } \mathbf{x}[i] = \mathbf{y}[j], \\ \infty & \text{otherwise.} \end{cases}$$

The language Γ_d contains a single unary weighted relation, which returns only finite values, together with k^2 binary 0-weighted relations, and hence is a binary language.

Example 1. *Let $\Gamma = \{\phi_{eq}\}$, where ϕ_{eq} is the equality relation on D, i.e., $\phi_{eq} : D \times D \to \overline{\mathbb{Q}}$ is defined by $\phi_{eq}(x, y) = 0$ if $x = y$ and $\phi_{eq}(x, y) = \infty$ if $x \neq y$.*

Then $D' = \mathrm{Feas}(\phi_{eq}) = \{(a, a) \,|\, a \in D\}$ and Γ_d consists of a single unary finite-valued relation ϕ', together with four binary 0-weighted relations $match_{1,1}, match_{1,2}, match_{2,1}$, and $match_{2,2}$.

Moreover, $\phi'(\mathbf{x}) = 0$ for every $\mathbf{x} \in D'$, and hence is trivial. All four of the other relations are in fact equal to the equality relation on D' defined by $\{((a, a), (a, a)) \mid (a, a) \in D'\}$. Thus, the dual of the equality relation on D consists of a trivial unary relation, together with the equality relation on D', where $|D| = |D'|$.

Example 2. *Let $\Gamma = \{\phi_{sum}\}$, where $\phi_{sum} : \{0, 1\}^3 \to \overline{\mathbb{Q}}$ is defined as follows:*

$$\phi_{sum}(x, y, z) \;=\; \begin{cases} x + 2y + 3z & \text{if } x + y + z = 1, \\ \infty & \text{otherwise.} \end{cases}$$

Then $D' = \mathrm{Feas}(\phi_{sum}) = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$ and Γ_d consists of a single unary finite-valued relation ϕ'_{sum}, together with nine binary 0-weighted relations $match_{1,1}, match_{1,2}, \ldots, match_{3,3}$.

If we set $a = (1, 0, 0), b = (0, 1, 0), c = (0, 0, 1)$, then $\phi'_{sum}(a) = 1; \phi'_{sum}(b) = 2$ and $\phi'_{sum}(c) = 3$. Also

$$match_{1,1}(x, y) \;=\; \begin{cases} 0 & \text{if } (x, y) \in \{(a, a), (b, b), \\ & \quad (b, c), (c, b), (c, c)\} \\ \infty & \text{otherwise} \end{cases}$$

$$match_{1,2}(x, y) = \begin{cases} 0 & \text{if } (x, y) \in \{(a, b), (b, a), \\ & \quad (b, a), (b, c), (c, a), (c, c)\} \\ \infty & \text{otherwise} \end{cases}$$

and so on.

The dual encoding using Γ_d

In this section we will describe the dual encoding described in (Dechter and Pearl 1989) for the CSP and later extended in (Larrosa and Dechter 2000) to soft constraint problems.

We will need the following notation. For any $\mathbf{x}_i \in V^m$ with $\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_m})$, we write $vars(\mathbf{x}_i)$ for the set $\{x_{i_1}, \ldots, x_{i_m}\}$.

By Proposition 1, without loss of generality we shall assume that Γ contains a single weighted relation $\phi_\Gamma : D^m \to \overline{\mathbb{Q}}$. Let $\mathcal{P}$ be an arbitrary instance of VCSP(Γ) with variables $V = \{x_1, \ldots, x_n\}$, domain D, and constraints $\phi_\Gamma(\mathbf{x}_1), \ldots, \phi_\Gamma(\mathbf{x}_q)$, where $\mathbf{x}_i \in V^m$ for all $1 \leq i \leq q$. We now describe the instance $\mathcal{P}_d$ in VCSP(Γ_d) which we call the *dual* of $\mathcal{P}$.

- The domain of $\mathcal{P}_d$ is $D' = \mathrm{Feas}(\phi_\Gamma) \subseteq D^m$.
- The variables $V' = \{x'_1, \ldots, x'_q\}$ of $\mathcal{P}_d$ correspond to the constraints of $\mathcal{P}$.
- For every $1 \leq i \leq q$, there is a unary constraint $\phi'_\Gamma(x'_i)$, where $\phi'_\Gamma : D' \to \overline{\mathbb{Q}}$ is as defined in Definition 6.
- If the scopes of two constraints of $\mathcal{P}$, say $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$, overlap, then there are binary constraints between x'_i and x'_j enforcing equality on the overlapping variables. More specifically, if $\mathbf{x}_i = (x_{i_1}, \ldots, x_{i_m})$, $\mathbf{x}_j = (x_{j_1}, \ldots, x_{j_m})$, and $vars(\mathbf{x}_i) \cap vars(\mathbf{x}_j) \neq \emptyset$ then there is a binary constraint $match_{k,l}(x'_i, x'_j)$ for every $k, l \in \{1, \ldots, m\}$ with $i_k = j_l$.

The dual encoding provides a way to reduce instances of VCSP(Γ) to instances of VCSP(Γ_d). Our next result extends this observation to obtain the reverse reduction as well.

Proposition 2. *For any valued constraint language Γ such that $|\Gamma|$ is finite, there is a binary valued constraint language Γ_d, such that VCSP(Γ) and VCSP(Γ_d) are polynomial-time equivalent.*

Proof. By Proposition 1 we may assume that Γ consists of a single weighted relation $\phi_\Gamma : D^m \to \overline{\mathbb{Q}}$. Moreover, since D is finite, and m is fixed, we may assume that this weighted relation is given extensionally as a table of values.

Hence, for any instance $\mathcal{P}$ of VCSP(Γ) we can construct in polynomial-time the dual instance $\mathcal{P}_d$ in VCSP(Γ_d) as defined above. It is straightforward to show that the assignments that minimise the objective function of $\mathcal{P}_d$ correspond precisely to the assignments that minimise the objective function of $\mathcal{P}$, and hence we have a polynomial-time reduction from VCSP(Γ) to VCSP(Γ_d).

For the other direction, given any instance $\mathcal{P}'$ in VCSP(Γ_d) we now indicate how to construct a corresponding instance $\mathcal{P}$ in VCSP(Γ).

For each variable x'_i of $\mathcal{P}'$ we introduce a fresh set of m variables for $\mathcal{P}$. If there is a unary constraint $\phi'_\Gamma(x'_i)$ in $\mathcal{P}$', then we introduce the constraint ϕ_Γ on the corresponding variables of $\mathcal{P}$. If there is no unary constraint on x'_i, then we introduce the constraint $\mathrm{Feas}(\phi_\Gamma)$ on the corresponding variables of $\mathcal{P}$ to code the fact that the domain of x'_i is D'. If there is a binary constraint $match_{k,l}(x'_i, x'_j)$ in $\mathcal{P}'$, then we merge the kth and lth variables in the corresponding sets of variables in $\mathcal{P}$. This construction can be carried out in polynomial time.

We have constructed an instance $\mathcal{P}$ in VCSP($\{\phi_\Gamma, \mathrm{Feas}(\phi_\Gamma)\}$) such that assignments minimising the objective function of $\mathcal{P}$ correspond precisely

to assignments minimising the objective function of $\mathcal{P}'$. Hence we have established a polynomial time reduction from VCSP(Γ_d) to VCSP($\Gamma \cup \{\text{Feas}(\phi_\Gamma)\}$).

However, it follows from the proof of Theorem 4.3 of (Cohen et al. 2013) that VCSP($\Gamma \cup \{\text{Feas}(\phi_\Gamma)\}$) can be reduced to VCSP(Γ) in polynomial-time. □

Algebraic properties of Γ_d

Over the past few years there has been considerable progress in investigating the complexity of different kinds of constraint satisfaction problems and valued constraint satisfaction problems by looking at the algebraic properties of the relations and weighted relations that define the constraints and valued constraints (Jeavons, Cohen, and Gyssens 1997; Jeavons 1998; Feder and Vardi 1998; Bulatov, Krokhin, and Jeavons 2005; Cohen et al. 2013)

Polymorphisms

It was shown in (Jeavons, Cohen, and Gyssens 1997; Jeavons 1998) that the computational complexity of CSP(Γ), the class of CSP instances where the constraint relations all belong to some fixed set Γ, is determined, up to polynomial-time reductions, by a set of operations known as the polymorphisms of Γ, which we will now define.

We first need some standard terminology. A function $f : D^k \to D$ is called a k-ary *operation* on D. The k-ary *projections*, defined for all $1 \leq i \leq k$, are the operations $\text{e}_i^{(k)}$ such that $\text{e}_i^{(k)}(x_1, \ldots, x_k) = x_i$. For any tuples $\mathbf{x_1}, \ldots, \mathbf{x_k} \in D^m$, we denote by $\overline{f}(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ the tuple in D^m obtained by applying f to $\mathbf{x}_1, \ldots, \mathbf{x}_k$ componentwise.

Definition 7. *Let $\phi : D^m \to \overline{\mathbb{Q}}$ be a weighted relation. An operation $f : D^k \to D$ is a* polymorphism *of ϕ if, for any $\mathbf{x}_1, \ldots, \mathbf{x}_k \in \text{Feas}(\phi)$ we have $\overline{f}(\mathbf{x}_1, \ldots, \mathbf{x}_k) \in \text{Feas}(\phi)$.*

We denote by $\text{Pol}(\Gamma)$ the set of all operations on D which are polymorphisms of all $\phi \in \Gamma$. We denote by $\text{Pol}^{(k)}(\Gamma)$ the k-ary operations in $\text{Pol}(\Gamma)$.

It follows directly from the definition that all projections are polymorphisms of all valued constraint languages.

Our next result shows that the polymorphisms of Γ_d are very closely related to the polymorphisms of Γ.

Theorem 1. *Let Γ be a valued constraint language such that $|\Gamma|$ is finite, and let Γ_d be the dual of Γ.*

The operation $f \in \text{Pol}^{(k)}(\Gamma)$ if and only if the operation $f_d \in \text{Pol}^{(k)}(\Gamma_d)$ where $f_d(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \overline{f}(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ for all $\mathbf{x}_i$ in the domain of Γ_d.

Proof. By Proposition 1 we may assume that Γ consists of a single weighted relation $\phi_\Gamma : D^m \to \overline{\mathbb{Q}}$, and hence that the domain D' of Γ_d is a subset of D^m. First, consider any $f : D^k \to D \in \text{Pol}^{(k)}(\Gamma)$, and the corresponding $f_d : (D')^k \to D$ given by $f_d(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \overline{f}(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ for all $\mathbf{x}_i \in D^m$. Since f is a polymorphism of ϕ_Γ, it is also a polymorphism of the unary weighted relation ϕ'_Γ in Γ_d. It is straightforward to check that f_d is also a polymorphism of all binary $match_{i,j}$ relations in Γ_d (since it will return the same label at all positions where its arguments have the same label). Hence $f_d \in \text{Pol}^{(k)}(\Gamma_d)$. Now consider any $f_d : (D')^k \to D' \in \text{Pol}^{(k)}(\Gamma_d)$. Since f_d is a polymorphism of $match_{i,i}$ it must return an element of D' whose label in position i is a function, g_i, of the labels in position i of its arguments. Moreover, since f_d is a polymorphism of $match_{i,j}$, the functions g_i and g_j must return the same results for all possible arguments from D'. Hence, there is a single function $g : D^k \to D$ such that the result returned by $f_d(\mathbf{x}_1, \ldots, \mathbf{x}_k)$ is equal to $\overline{g}(\mathbf{x}_1, \ldots, \mathbf{x}_k)$. Now, since f_d must return an element of D', it follows that g must be a polymorphism of ϕ_Γ, which gives the result. □

The individual weighted relations in Γ_d often have other polymorphisms, that are not of the form indicated in Theorem 1, but the only polymorphisms that are shared by every weighted relation in Γ_d are those that correspond to polymorphisms of Γ in this way.

Example 3. *Recall the language $\Gamma = \{\phi_{sum}\}$, defined in Example 2.*

The weighted relation ϕ_{sum} has no polymorphisms, except for the projection operations on $D = \{0, 1\}$.

However, the unary finite-valued relation ϕ'_{sum}, has every *operation on $D' = \{a, b, c\}$ as a polymorphism.*

The binary 0-weighted relation $match_{1,1}$ has many operations on D' as polymorphisms, including all of the constant operations.

The binary 0-weighted relation $match_{1,2}$ also has many operations on D' as polymorphisms, including the ternary majority *operation g defined by*

$$g(x, y, z) = \begin{cases} x & \text{if } x = y \text{ or } x = z \\ y & \text{if } y = z \\ c & \text{otherwise} \end{cases}$$

but not including the constant operation returning the label a, or the constant operation returning the label b.

The only operations that are polymorphisms of every *weighted relation in Γ_d are the projection operations on D'.*

For certain types of languages Γ there are known to be polynomial-time algorithms to determine whether an instance of CSP(Γ) has an assignment that is allowed by all the constraints. In all known cases, these special-purpose algorithms can be applied precisely when Γ has polymorphisms that satisfy certain *identities*. For example, it is known (Feder and Vardi 1998; Jeavons, Cohen, and Cooper 1998) that enforcing *3-consistency* can decide whether an instance has a solution if and only if Γ has a polymorphism g that satisfies the identities:

$$g(x, x, y) = g(x, y, x) = g(y, x, x) = x \quad \forall x, y \in D.$$

In fact, it is known that any crisp language that is not NP-complete must have a polymorphism that satisfies certain kinds of identities (Bulatov, Krokhin, and Jeavons 2005).

One simple consequence of Theorem 1 is that the polymorphisms of Γ and the polymorphisms of Γ_d satisfy exactly the same identities.

Corollary 1. *Let Γ be a valued constraint language such that $|\Gamma|$ is finite, and let Γ_d be the dual of Γ.*

The operations in $\text{Pol}(\Gamma)$ and the operations in $\text{Pol}(\Gamma_d)$ satisfy exactly the same identities.

We will show below that it follows from Corollary 1 that essentially the same algorithms can be applied to either CSP(Γ) or CSP(Γ_d). Hence, from the point of view of the availability of known efficient algorithms, it does not matter whether a CSP problem is formulated using constraints of arbitrary arity, or using a dual (binary) formulation.

Although they satisfy the same identities, the polymorphisms of Γ_d do not, in general, share *all* the same properties as the polymorphisms of Γ. For example, $\mathrm{Pol}(\Gamma)$ might include the binary operation $\min$ that returns the smaller of its two arguments, according to some fixed ordering of D. This operation has the property of being *conservative*, which means that the result is always equal to one of the arguments. However, the corresponding operation $\min_d$ in $\mathrm{Pol}(\Gamma_d)$ is *not* generally conservative, since, for example, $\min_d((a,b),(b,a)) = (a,a)$ for all $a < b$.

One way to simplify the analysis of valued constraint languages is to restrict our attention to certain special kinds of languages that have desirable features. For example, a valued constraint language Γ is *not* a core if there is a label $a \in D$ such that for any instance $\mathcal{P} \in \mathrm{VCSP}(\Gamma)$ there is an optimal solution to $\mathcal{P}$ that does not use the label a. In this case, label a can be discarded.

Definition 8. *A valued constraint language Γ is a* core *if all unary polymorphisms of Γ are bijections. Moreover, Γ is a* rigid core *if the only unary polymorphism of Γ is the identity function.*

We can restrict our attention to languages that are rigid cores due to the following result.

Proposition 3 (Ochremiak 2014). *For every valued constraint language Γ there is a valued constraint language Γ' that is a rigid core and VCSP(Γ) and VCSP(Γ') are polynomial-time equivalent.*

An operation $f : D^k \to D$ is called *idempotent* if $f(x,\dots,x) = x$ for all $x \in D$. It is known that Γ is a rigid core if and only if all polymorphisms of Γ are idempotent (Ochremiak 2014).

Corollary 2. *Let Γ be a valued constraint language such that $|\Gamma|$ is finite, and let Γ_d be the dual of Γ.*

Γ is a rigid core if and only if Γ_d is a rigid core.

Proof. Follows immediately from Corollary 1, since the property of being idempotent is specified by an identity. □

A precise characterisation of rigid core languages Γ that give rise to CSP instances solvable by *any* form of local consistency has recently been established (Barto and Kozik 2014). This characterisation makes use of the following identities for a k-ary ($k \geq 2$) operation f:

$$f(y,x,\dots,x) = f(x,y,x,\dots,x) = f(x,\dots,x,y).$$

Any k-ary operation on D that satisfies these identities for all $x, y \in D$, is called a *weak near-unanimity* operation.

Theorem 2 (Barto and Kozik 2014). *Let Γ be a crisp language that is a rigid core. CSP(Γ) is solvable by local consistency if and only if $\mathrm{Pol}(\Gamma)$ contains weak near-unanimity operations of all but finitely many arities.*

A second polynomial-time algorithmic technique for the CSP generalises the idea of using Gaussian elimination to solve simultaneous linear equations. The most general version of this approach is based on the property of having a polynomial-sized representation for the solution set of any instance (Bulatov and Dalmau 2006; Idziak et al. 2010). Roughly, the algorithm works by starting from the empty set and adding constraints in an instance one by one while maintaining (in polynomial time) a small enough representation of the current solution set (of feasible assignments). At the end (i.e., after all constraints have been added), either this representation is non-empty and contains a solution to the instance or else there is no solution. This algorithm is called the "few subpowers" algorithm (because it is related to a certain algebraic property to do with the number of subalgebras in powers of an algebra). Languages where this algorithm is guaranteed to find a solution (or show that none exists) were characterised by (Idziak et al. 2010). Once again, this characterisation involves a set of identities on the polymorphisms of the language. A k-ary ($k \geq 3$) operation $f : D^k \to D$ is called an *edge* operation if, for all $x, y \in D$,

$$f(y,y,x,x,\dots,x) = f(y,x,y,x,x,\dots,x) = x \quad \text{and}$$

$$\begin{aligned} f(x,x,x,y,x,\dots,x) &= f(x,x,x,x,y,x,\dots,x) = \\ &\dots \qquad = f(x,\dots,x,y) = x. \end{aligned}$$

Theorem 3 (Idziak et al. 2010). *Let Γ be a crisp language. Then CSP(Γ) is solvable by the few subpowers algorithm if $\mathrm{Pol}(\Gamma)$ contains an edge operation.*

The converse to this theorem is true in the following sense: the absence of edge operations from $\mathrm{Pol}(\Gamma)$ implies that the presence of small enough representations is not guaranteed, see (Idziak et al. 2010) for details.

Combining Corollary 1 with Theorems 2 and 3 shows that the property of being solvable using local consistency, or the few subpowers algorithm, is possessed by a language Γ if and only if it is also possessed by the associated binary language Γ_d.

Weighted Polymorphisms

Polymorphisms are sufficient to analyse the complexity of the CSP, but for the VCSP, it has been shown that in general we need a more flexible notion that assigns weights to a collection of polymorphisms (Cohen et al. 2013).

Definition 9. *Let $\phi : D^m \to \overline{\mathbb{Q}}$ be a weighted relation, and let $C \subseteq \mathrm{Pol}^{(k)}(\phi)$ be some collection of polymorphisms of ϕ. A function $\omega : C \to \mathbb{Q}$ is called a k-ary* weighted polymorphism *of ϕ on C if it satisfies the following conditions:*

- $\sum_{f \in C} \omega(f) = 0$*;*
- *if $\omega(f) < 0$, then f is a projection;*
- *for any $\mathbf{x_1}, \dots, \mathbf{x_k} \in \mathrm{Feas}(\phi)$*

$$\sum_{f \in C} \omega(f)\phi(\overline{f}(\mathbf{x_1},\dots,\mathbf{x_k})) \;\leq\; 0\,. \tag{2}$$

We denote by $\mathrm{wPol}^{(k)}(\Gamma)$ the set of all functions $\omega : \mathrm{Pol}^{(k)}(\Gamma) \to \mathbb{Q}$ which are weighted polymorphisms of all $\phi \in \Gamma$.

Example 4. *Let $D = \{0, 1\}$. Let Γ be the set of weighted relations $\phi : D^m \to \overline{\mathbb{Q}}$ that admit ω_{sub} as a weighted polymorphism, where ω_{sub} is defined by $\omega_{sub}(f) = -1$ if $f \in \{\mathrm{e}_1^{(2)}, \mathrm{e}_2^{(2)}\}$, $\omega_{sub}(f) = +1$ if $f \in \{\min, \max\}$, and $\omega_{sub}(f) = 0$ otherwise; here $\min$ and $\max$ are the binary operations returning the smaller and larger of their two arguments, respectively, wrt the usual order $0 < 1$.*

In this case Γ is precisely the well-studied class of submodular set functions (Schrijver 2003).

Our next result shows that the weighted polymorphisms of Γ_d are closely related to the weighted polymorphisms of Γ.

Theorem 4. *Let Γ be a valued constraint language such that $|\Gamma|$ is finite, and let Γ_d be the dual of Γ.*

A function $\omega : \mathrm{Pol}^{(k)}(\Gamma) \to \mathbb{Q} \in \mathrm{wPol}^{(k)}(\Gamma)$ if and only if the function $\omega_d : \mathrm{Pol}^{(k)}(\Gamma_d) \to \mathbb{Q} \in \mathrm{wPol}^{(k)}(\Gamma_d)$, where $\omega_d(f_d) = \omega(f)$ for all $f \in \mathrm{Pol}^{(k)}(\Gamma)$ and their corresponding operations $f_d \in \mathrm{Pol}^{(k)}(\Gamma_d)$ (as defined in Theorem 1).

Proof. By Proposition 1 we may assume that Γ consists of a single weighted relation $\phi_\Gamma : D^m \to \overline{\mathbb{Q}}$, and hence that the domain D' of Γ_d is a subset of D^m.

First, consider any $\omega : \mathrm{Pol}^{(k)}(\Gamma) \to \mathbb{Q} \in \mathrm{wPol}^{(k)}(\Gamma)$, and the corresponding $\omega_d : \mathrm{Pol}^{(k)}(\Gamma_d) \to \mathbb{Q}$ given by $\omega_d(f_d) = \omega(f)$ for all $f \in \mathrm{Pol}^{(k)}(\Gamma)$. Since ω is a weighted polymorphism of ϕ_Γ, it is easy to check that ω_d satisfies all three conditions in Definition 9, and hence is a weighted polymorphism of the unary weighted relation ϕ'_Γ in Γ_d. Since all other weighted relations in Γ_d are the 0-weighted $match_{i,j}$ relations, the third condition in Definition 9 holds trivially for all these weighted relations, and hence ω_d is a weighted polymorphism of all weighted relations in Γ_d.

Now consider any $\omega_d : \mathrm{Pol}^{(k)}(\Gamma_d) \to \mathbb{Q} \in \mathrm{wPol}^{(k)}(\Gamma_d)$. Since ω_d is a weighted polymorphism of ϕ'_Γ, the function $\omega : \mathrm{Pol}^{(k)}(\Gamma) \to \mathbb{Q}$ that assigns the same weights to corresponding elements of $\mathrm{Pol}^{(k)}(\Gamma)$ satisfies all the conditions of Definition 9, and hence is a weighted polymorphism of ϕ_Γ. □

Weighted polymorphisms capture the complexity of any valued constraint language (Cohen et al. 2013). In fact, for any valued constraint language Γ, it was shown that the associated class of problems VCSP(Γ) is NP-hard, unless Γ has certain kinds of weighted polymorphisms.

Moreover, the weighted polymorphisms of Γ can be used to select an appropriate algorithmic technique for VCSP(Γ). The algorithmic technique for the VCSP that has been most thoroughly investigated is based on linear programming: every VCSP instance Φ has a natural linear programming relaxation called the *basic LP relaxation*, and denoted BLP(Φ) - see (Kolmogorov, Thapper, and Živný 2013) and the references therein.

Given a VCSP instance Φ, we say that BLP *solves* Φ if the solution to BLP(Φ) is equal to the minimal value of Φ over all assignments to the variables. Moreover, we say that BLP solves a valued constraint language Γ if BLP solves every instance $\Phi \in$ VCSP(Γ). It is shown in (Kolmogorov, Thapper, and Živný 2013) that in all cases where BLP solves Γ, a standard self-reduction technique can be used to obtain an assignment that minimises any Φ in VCSP(Γ) in polynomial time.

The power of BLP for valued constraint languages was fully characterised in (Thapper and Živný 2012). To state this result, we first introduce some further terminology about operations. A k-ary operation $f : D^k \to D$ is called *symmetric* if for every permutation π on $\{1, \ldots, k\}$, it satisfies the identity

$$f(x_1, \ldots, x_k) = f(x_{\pi(1)}, \ldots, x_{\pi(k)}).$$

A weighted polymorphism ω is called symmetric if it assigns positive weight to one or more symmetric operations, and no others.

Theorem 5 (Thapper and Živný 2012). *Let Γ be a valued constraint language. VCSP(Γ) can be solved using the BLP algorithm if and only if Γ has a k-ary symmetric weighted polymorphism, for every $k \geq 2$.*

Example 5. *A binary operation $f : D^2 \to D$ is called a* semilattice *operation if f is associative, commutative, and idempotent. Example of semilattice operations include the $\min$ and $\max$ operations on ordered sets that return the smaller or larger of their two arguments. Since any semilattice operation generates symmetric operations of all arities, Theorem 5 implies that any valued constraint language with a binary weighted polymorphism that assigns positive weight to some semilattice operation is solvable using the BLP. Such languages include the submodular languages (Example 4) and several others - see (Jeavons, Krokhin, and Živný 2014).*

Combining Corollary 1 with Theorem 4 and Theorem 5 shows that the property of being solvable using BLP is possessed by a language Γ if and only if it is also possessed by the associated binary language Γ_d.

Conclusion

Transforming a constraint satisfaction problem to a binary problem has a number of advantages and disadvantages which have been investigated by many authors (Rossi, Dahr, and Petrie 1990; Bacchus et al. 2002; Stergiou and Samaras 2005). Such a transformation changes many aspects of the problem, such as what inferences can be derived by various kinds of propagation. One might expect that achieving the simplicity of a binary representation would incur a corresponding increase in the sophistication of the required solving algorithms.

However, we have shown here that the well-known dual encoding of the VCSP converts any finite language, Γ, of arbitrary arity to a *binary* language, Γ_d, of a very restricted kind, such that there is a bijection between the polymorphisms of Γ and Γ_d, and the corresponding polymorphisms satisfy exactly the same identities and weightings. Hence we have shown that the algebraic analysis of valued constraint languages can focus on a very restricted class of binary languages. Moreover, many important algorithmic properties, such as the ability to solve problems using a bounded level of consistency, or by a linear programming relaxation, are also preserved by the dual encoding.

References

Atserias, A. 2008. On digraph coloring problems and treewidth duality. *Eur. J. Comb.* 29(4):796–820.

Bacchus, F.; Chen, X.; van Beek, P.; and Walsh, T. 2002. Binary vs. non-binary constraints. *Artificial Intelligence* 140(1/2):1–37.

Barto, L., and Kozik, M. 2014. Constraint Satisfaction Problems Solvable by Local Consistency Methods. *Journal of the ACM* 61(1). Article No. 3.

Barto, L. 2013. Finitely related algebras in congruence distributive varieties have near unanimity terms. *Canadian Journal of Mathematics* 65:3–21.

Bulatov, A., and Dalmau, V. 2006. A simple algorithm for Mal'tsev constraints. *SIAM Journal on Computing* 36(1):16–27.

Bulatov, A.; Krokhin, A.; and Jeavons, P. 2005. Classifying the Complexity of Constraints using Finite Algebras. *SIAM Journal on Computing* 34(3):720–742.

Bulín, J.; Delic, D.; Jackson, M.; and Niven, T. 2013. On the Reduction of the CSP Dichotomy Conjecture to Digraphs. In *Proceedings of the 19th International Conference on Principles and Practice of Constraint Programming (CP'13)*, volume 8124 of *Lecture Notes in Computer Science*, 184–199. Springer.

Bulín, J.; Delic, D.; Jackson, M.; and Niven, T. 2014. A finer reduction of constraint problems to digraphs. Technical report. arXiv:1406.6413.

Cohen, D. A.; Cooper, M. C.; Creed, P.; Jeavons, P.; and Živný, S. 2013. An algebraic theory of complexity for discrete optimisation. *SIAM Journal on Computing* 42(5):915–1939.

Cohen, D. A.; Jeavons, P. G.; and Živný, S. 2008. The expressive power of valued constraints: Hierarchies and collapses. *Theoretical Computer Science* 409(1):137–153.

Dechter, R., and Pearl, J. 1989. Tree Clustering for Constraint Networks. *Artificial Intelligence* 38:353–366.

Dechter, R. 2003. *Constraint Processing*. Morgan Kaufmann.

Feder, T., and Vardi, M. Y. 1998. 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.

Idziak, P. M.; Markovic, P.; McKenzie, R.; Valeriote, M.; and Willard, R. 2010. Tractability and learnability arising from algebras with few subpowers. *SIAM Journal on Computing* 39(7):3023–3037.

Jeavons, P.; Cohen, D.; and Cooper, M. C. 1998. Constraints, Consistency and Closure. *Artificial Intelligence* 101(1–2):251–265.

Jeavons, P. G.; Cohen, D. A.; and Gyssens, M. 1997. Closure Properties of Constraints. *Journal of the ACM* 44(4):527–548.

Jeavons, P.; Krokhin, A.; and Živný, S. 2014. The complexity of valued constraint satisfaction. *Bulletin of the European Association for Theoretical Computer Science (EATCS)* 113:21–55.

Jeavons, P. G. 1998. On the Algebraic Structure of Combinatorial Problems. *Theoretical Computer Science* 200(1-2):185–204.

Kazda, A. 2011. Maltsev digraphs have a majority polymorphism. *Eur. J. Comb.* 32(3):390–397.

Kolmogorov, V.; Thapper, J.; and Živný, S. 2013. The power of linear programming for general-valued CSPs. Technical report. arXiv:1311.4219. Submitted for publication.

Larrosa, J., and Dechter, R. 2000. On the Dual Representation of non-Binary Semiring-based CSPs. In *Workshop on Soft Constraints – CP'00*.

Ochremiak, J. 2014. Algebraic properties of valued constraint satisfaction problem. Technical report. arXiv:1403.0476.

Powell, R., and Krokhin, A. 2014. A reduction of VCSP to digraphs. Unpublished manuscript.

Rossi, F.; Dahr, V.; and Petrie, C. 1990. On the Equivalence of Constraint Satisfaction Problems. In *Proceedings of the 9th European Conference on Artificial Intelligence (ECAI'90)*.

Schrijver, A. 2003. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer.

Smith, B. M.; Stergiou, K.; and Walsh, T. 2000. Using auxiliary variables and implied constraints to model non-binary problems. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI'00)*, 182–187.

Stergiou, K., and Samaras, N. 2005. Binary encodings of non-binary constraint satisfaction problems: Algorithms and experimental results. *Journal of Artificial Intelligence Research (JAIR)* 24:641–684.

Thapper, J., and Živný, S. 2012. The power of linear programming for valued CSPs. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS'12)*, 669–678. IEEE.

Živný, S.; Cohen, D. A.; and Jeavons, P. G. 2009. The Expressive Power of Binary Submodular Functions. *Discrete Applied Mathematics* 157(15):3347–3358.