



**HAL**  
open science

## **Binarisation for valued constraint satisfaction problems**

David A. Cohen, Martin Cooper, Peter G. Jeavons, Andrei Krokhin, Robert Powell, Stanislav Živný

► **To cite this version:**

David A. Cohen, Martin Cooper, Peter G. Jeavons, Andrei Krokhin, Robert Powell, et al.. Binarisation for valued constraint satisfaction problems. *SIAM Journal on Discrete Mathematics*, 2017, vol. 31 (n° 4), pp. 2279-2300. 10.1137/16M1088107 . hal-01646073

**HAL Id: hal-01646073**

**<https://hal.science/hal-01646073>**

Submitted on 23 Nov 2017

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



## Open Archive TOULOUSE Archive Ouverte (OATAO)

OATAO is an open access repository that collects the work of Toulouse researchers and makes it freely available over the web where possible.

This is an author-deposited version published in : <http://oatao.univ-toulouse.fr/>  
Eprints ID : 19187

**To link to this article** : DOI : 10.1137/16M1088107  
URL : <http://dx.doi.org/10.1137/16M1088107>

**To cite this version** : Cohen, David A. and Cooper, Martin C. and Jeavons, Peter G. and Krokhin, Andrei and Powell, Robert and Zivny, Stanislav *Binarisation for Valued Constraint Satisfaction Problems*. (2017) SIAM Journal on Discrete Mathematics, vol. 31 (n° 4). pp. 2279-2300. ISSN 0895-4801

Any correspondence concerning this service should be sent to the repository administrator: [staff-oatao@listes-diff.inp-toulouse.fr](mailto:staff-oatao@listes-diff.inp-toulouse.fr)

# Binarisation for Valued Constraint Satisfaction Problems\*

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

Peter G. Jeavons  
University of Oxford  
peter.jeavons@cs.ox.ac.uk

Robert Powell  
University of Durham  
robert.powell@durham.ac.uk

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

Andrei Krokhin  
University of Durham  
andrei.krokhin@durham.ac.uk

Stanislav Živný  
University of Oxford  
standa.zivny@cs.ox.ac.uk

## Abstract

We study methods for transforming valued constraint satisfaction problems (VCSPs) to *binary* VCSPs. First, we show that the standard *dual* encoding preserves many aspects of the algebraic properties that capture the computational complexity of VCSPs. Second, we extend the reduction of CSPs to binary CSPs described by Bulín et al. [LMCS'15] to VCSPs. This reduction establishes that VCSPs over a fixed valued constraint language are polynomial-time equivalent to Minimum-Cost Homomorphism Problems over a fixed digraph.

## 1 Introduction

The valued constraint satisfaction problem (VCSP) is a general framework for problems that involve finding an assignment of values to a set of variables, where the assignment must satisfy certain feasibility conditions and optimise a certain objective function. The VCSP includes as a special case the (purely decision) constraint satisfaction problem (CSP) [38] as well as the (purely optimisation) minimum constraint satisfaction problem (Min-CSP), see [34] for a recent survey.

Different subproblems of the VCSP can be obtained by restricting, in various ways, the set of cost functions that can be used to express the constraints. Such a set of cost functions is generally called a valued constraint language [16, 34]. For any such valued constraint language  $\Gamma$  there is a corresponding problem  $\text{VCSP}(\Gamma)$ , and it has been shown that the computational

---

\*An extended abstract of part of this work appeared in the *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)* [17]. Part of this work appeared in a technical report [40] and in the fifth author's doctoral thesis [41]. David Cohen, Martin Cooper, Peter Jeavons, and Stanislav Živný were supported by EPSRC grant EP/L021226/1. Stanislav Živný was supported by a Royal Society University Research Fellowship. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532). The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein.

complexity of  $\text{VCSP}(\Gamma)$  is determined by certain algebraic properties of the set  $\Gamma$  known as fractional polymorphisms [16]. The classical constraint satisfaction problem (CSP) [20] is a special case of the VCSP in which all cost functions are relations. If a valued constraint language  $\Gamma$  contains only relations then we call  $\Gamma$  a constraint language.

There has been significant progress on classifying the computational complexity of different constraint languages [43, 23, 8, 12] and valued constraint languages [32, 25, 31, 51, 33, 52, 30]. Most notably, it has been shown that a dichotomy for constraint languages, conjectured by Feder and Vardi [20], implies a dichotomy for valued constraint languages [30]. This result thus resolves the complexity of valued constraint languages modulo the complexity of constraint languages.

In *binary* VCSPs every valued constraint involves at most two variables; in other words, the interaction between variables is only pairwise. In this paper we consider transformations of the general VCSP, with constraints of arbitrary arity, to the binary VCSP. There are several motivations for studying such reductions. Firstly, binary VCSPs have been extensively studied in the context of energy minimisation problems in computer vision and machine learning [7, 39] since pairwise interaction is enough to model interesting problems. Secondly, algorithms for binary VCSPs may be easier to design, as discussed below in the case of submodular VCSPs. Finally, various aspects of binary VCSPs, such as the algebraic properties that capture the complexity of valued constraint languages, may be easier to study on binary instances.

One important class of valued constraint languages are the *submodular* languages [45]. It is known that VCSP instances where all constraints are submodular can be solved in polynomial time, although the algorithms that have been proposed to achieve this in the general case are rather intricate and difficult to implement [27, 44]. In the special case of binary submodular constraints a much simpler algorithm can be used to find a minimising assignment of values, based on a standard max-flow algorithm [15]. Our results in this paper show that this simpler algorithm can be used to obtain exact solutions to arbitrary VCSP instances with submodular constraints (from a finite language) in polynomial time.

The more restricted question of which valued constraint languages can be transformed to binary valued constraint languages *over the same domain* was studied in [18]. It was shown in [55] that there are submodular valued constraint languages which *cannot* be expressed (using min and sum) by binary submodular languages over the same domain.

However, there are two well-known methods for transforming a non-binary CSP into a binary one over a different domain of values; the *dual encoding* [19] and the *hidden variable encoding* [42]. Both encode the non-binary constraints to variables that have as domains of possible labels the valid tuples of the constraints. That is, these 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 [47]. It was observed in [36] that both of these standard encodings can be extended to valued constraints.

It is also known that any CSP with a fixed constraint language is polynomial-time equivalent to one where the constraint language consists of a single binary relation (i.e., a digraph) [20, 1, 14]. Recent work by Bulín et al. shows that this reduction can be done in a way that preserves certain algebraic properties of the constraint language that are known to characterise the complexity of the corresponding CSP [14].

As our first contribution, we extend the idea of the dual encoding to valued constraint satisfaction and show that this standard encoding preserves many aspects of the algebraic properties that capture the complexity of valued constraint languages. In particular, we show

that for any valued constraint language  $\Gamma$  of finite size, there is a one-to-one correspondence between the fractional polymorphisms of  $\Gamma$  and the fractional polymorphisms of the binary language  $\Gamma_d$  obtained by the dual encoding. Moreover, we show that  $\Gamma_d$  preserves all *identities* involving (fractional) polymorphisms of  $\Gamma$ , where an identity is an equality between arbitrary expressions involving only polymorphisms and all variables are universally quantified. A large body of research on the complexity of (valued) constraint languages has shown that it is the identities satisfied by the (fractional) polymorphisms that determine both the complexity and suitable algorithmic solution techniques [34].

Hence, as well as providing a way to convert any given instance of the VCSP to an equivalent binary instance, we show that 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. We remark that a similar transformation from constraint languages of arbitrary arity to sets of unary and binary relations was used in [5] (and also implicitly in [3]), for the special case of the CSP.

While the idea of the dual encoding is very simple, the resulting  $\Gamma_d$  contains a single unary cost function and *more than one* binary relation (in general). However, all the binary relations that are included in  $\Gamma_d$  are of the same type and correspond to enforcing equality on the shared variables between different constraints in instances of  $\text{VCSP}(\Gamma)$ .

As our second contribution, we adapt the proof from [14] to the VCSP framework and show that each VCSP, with a fixed valued constraint language  $\Gamma$  of finite size, is polynomial-time equivalent to a VCSP with valued constraint language  $\Gamma_e$ , where  $\Gamma_e$  consists of a single unary cost function and a *single* binary relation (i.e., a digraph). Problems of this type have been studied as the Minimum-Cost Homomorphism Problem ( $\text{MinCostHom}$ ) [22, 24, 49], which makes this result somewhat surprising as it was believed that  $\text{MinCostHom}$  was essentially a more restricted optimisation problem than the VCSP.

This second reduction, which we call the *extended dual*, again preserves many aspects of the algebraic properties that capture the complexity of valued constraint languages. In fact, we show that it preserves all identities involving (fractional) polymorphisms of  $\Gamma$  which are *linear* and *balanced*. These are the key properties for characterising most known tractable cases.

However, the extended dual encoding does not preserve all identities: in particular, it does not preserve the (unbalanced) identities defining Mal'tsev 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 Mal'tsev polymorphism also has a majority polymorphism [29]; the former is solved by a generalised form of Gaussian elimination whereas the latter is solved by local consistency operations.

In summary, our first reduction, using the dual encoding, transforms any valued constraint problem over an arbitrary valued constraint language  $\Gamma$  of finite size to a binary problem with more than one form of binary constraint, which satisfies all of the identities on fractional polymorphisms satisfied by  $\Gamma$ . Our second reduction, using the extended dual encoding, transforms any valued constraint problem over an arbitrary valued constraint language  $\Gamma$  of finite size to a binary problem with just one form of binary constraint, which satisfies an important subclass of the identities satisfied by  $\Gamma$ .

## 2 Background and Definitions

In this section we will give the necessary background. Section 2.1 defines the VCSP, whereas Sections 2.2 and 2.3 present the basics of the algebraic approach to studying the complexity of the VCSP.

### 2.1 Valued Constraint Satisfaction Problems

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

**Definition 1.** An  $m$ -ary cost function over  $D$  is any mapping  $\phi : D^m \rightarrow \overline{\mathbb{Q}}$ . We denote by  $\Phi_D^{(m)}$  the set of all  $m$ -ary cost functions and let  $\Phi_D = \bigcup_{m \geq 1} \Phi_D^{(m)}$ .

We call  $D$  the *domain*, the elements of  $D$  *labels* (for variables), and we say that the cost functions in  $\Phi_D$  take *values* (which are elements of  $\overline{\mathbb{Q}}$ ).

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

It is convenient to highlight the special case when the values taken by a cost function are restricted to 0 and  $\infty$ .

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

**Definition 3.** Let  $X = \{x_1, \dots, x_n\}$  be a set of variables. A *valued constraint* over  $X$  is an expression of the form  $\phi(\mathbf{x})$  where  $\phi \in \Phi_D^{(m)}$  and  $\mathbf{x} \in X^m$ , 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 cost function  $\phi$  is called the *constraint cost function*.

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

$$\Phi_{\mathcal{I}}(x_1, \dots, x_n) = \sum_{i=1}^q \phi_i(\mathbf{x}_i) \quad (1)$$

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

Any assignment of labels from  $D$  to the variables of  $X$  for which  $\Phi_{\mathcal{I}}$  is finite will be called a *feasible solution* to  $\mathcal{I}$ . The goal is to find a feasible solution that minimises  $\Phi_{\mathcal{I}}$ .

**Definition 5.** Any set  $\Gamma \subseteq \Phi_D$  of cost functions on some fixed domain  $D$  is called a *valued constraint language*, or simply a *language*.

We will denote by  $\text{VCSP}(\Gamma)$  the class of all VCSP instances in which the constraint cost functions are all contained in  $\Gamma$ .

The classical constraint satisfaction problem (CSP) can be seen as a special case of the VCSP in which all cost functions are crisp (i.e., relations). A language containing only crisp cost functions is called *crisp*.

A language  $\Gamma$  is called *binary* if all cost functions from  $\Gamma$  are of arity at most two.

## 2.2 Fractional Polymorphisms

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 cost functions that define the constraints and valued constraints [28, 20, 11, 16] resulting in strong complexity classifications [32, 25, 51, 30]. We present here some of the tools used in this line of work.

We first need some standard terminology. A function  $f : D^k \rightarrow D$  is called a  $k$ -ary operation on  $D$ . For any tuples  $\mathbf{x}_1, \dots, \mathbf{x}_k \in D^m$ , we denote by  $f(\mathbf{x}_1, \dots, \mathbf{x}_k)$  the tuple in  $D^m$  obtained by applying  $f$  to  $\mathbf{x}_1, \dots, \mathbf{x}_k$  componentwise.

**Definition 6.** Let  $\phi : D^m \rightarrow \overline{\mathbb{Q}}$  be a cost function. An operation  $f : D^k \rightarrow D$  is a *polymorphism* of  $\phi$  if, for any  $\mathbf{x}_1, \dots, \mathbf{x}_k \in \text{Feas}(\phi)$  we have  $f(\mathbf{x}_1, \dots, \mathbf{x}_k) \in \text{Feas}(\phi)$ .

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  $\text{Pol}(\Gamma)$ .

The  $k$ -ary *projections*, defined for all  $1 \leq i \leq k$ , are the operations  $e_i^{(k)}$  such that  $e_i^{(k)}(x_1, \dots, x_k) = x_i$ . It follows directly from Definition 6 that all projections are polymorphisms of all valued constraint languages.

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 [16, 21].

**Definition 7.** Let  $\phi : D^m \rightarrow \overline{\mathbb{Q}}$  be a cost function. A probability distribution  $\omega$  on the set of  $k$ -ary polymorphisms of  $\phi$  (i.e.,  $\omega : \text{Pol}^{(k)}(\phi) \rightarrow \mathbb{Q}_{\geq 0}$  with  $\sum_{f \in \text{Pol}^{(k)}(\phi)} \omega(f) = 1$ ) is called a  $k$ -ary *fractional polymorphism* of  $\phi$  if for any  $\mathbf{x}_1, \dots, \mathbf{x}_k \in \text{Feas}(\phi)$

$$\sum_{f \in \text{Pol}^{(k)}(\phi)} \omega(f) \phi(f(\mathbf{x}_1, \dots, \mathbf{x}_k)) \leq \frac{1}{k} \sum_{i=1}^k \phi(\mathbf{x}_i). \quad (2)$$

We denote by  $\text{fPol}^{(k)}(\Gamma)$  the set of  $k$ -ary fractional polymorphisms of all  $\phi \in \Gamma$  and set  $\text{fPol}(\Gamma) = \bigcup_{k \geq 1} \text{fPol}^{(k)}(\Gamma)$ .

For any  $\omega \in \text{fPol}(\Gamma)$  we denote by  $\text{supp}(\omega)$  the set  $\{f \in \text{Pol}^{(k)}(\phi) \mid \omega(f) > 0\}$  and define  $\text{supp}(\Gamma) = \bigcup_{\omega \in \text{fPol}(\Gamma)} \text{supp}(\omega)$ .

**Example 1.** Let  $D = \{0, 1\}$ . Let  $\Gamma$  be the set of cost functions  $\phi : D^m \rightarrow \overline{\mathbb{Q}}$  that admit  $\omega_{sub}$  as a fractional polymorphism, where  $\omega_{sub}$  is defined by  $\text{supp}(\omega_{sub}) = \{\min, \max\}$  and  $\omega_{sub}(\min) = \omega_{sub}(\max) = \frac{1}{2}$ ; here  $\min$  and  $\max$  are the binary operations returning the smaller and larger of their two arguments, respectively, with respect to the usual order  $0 < 1$ .

In this case  $\Gamma$  is precisely the well-studied class of submodular set functions [45].

## 2.3 Identities and Rigid Cores

Many important properties of polymorphisms can be specified by *identities*, i.e., equalities of terms that hold for all choices of the variables involved in them. More formally, an operational signature is a set of operation symbols with arities assigned to them and an identity is an expression  $t_1 = t_2$  where  $t_1$  and  $t_2$  are terms in this signature.

Here are some examples of important properties of operations that are specified by identities:

- An operation  $f$  is *idempotent* if it satisfies the identity  $f(x, \dots, x) = x$ .
- A  $k$ -ary ( $k \geq 2$ ) operation  $f$  is *weak near unanimity (WNU)* if it is idempotent and satisfies the identities  $f(y, x, \dots, x, x) = f(x, y, \dots, x, x) = \dots = f(x, x, \dots, x, y)$ .
- A  $k$ -ary ( $k \geq 2$ ) operation  $f$  is *cyclic* if  $f(x_1, x_2, \dots, x_k) = f(x_2, \dots, x_k, x_1)$ .
- A  $k$ -ary ( $k \geq 2$ ) operation  $f$  is *symmetric* if  $f(x_1, \dots, x_k) = f(x_{\pi(1)}, \dots, x_{\pi(k)})$  for each permutation  $\pi$  on  $\{1, \dots, k\}$ .
- A  $k$ -ary ( $k \geq 3$ ) operation  $f$  is *edge* if

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

and, for all  $4 \leq i \leq k$ ,

$$f(x, \dots, x, y, x, \dots, x) = x \text{ where } y \text{ is in position } i.$$

An identity  $t_1 = t_2$  is said to be *linear* if both  $t_1$  and  $t_2$  involve at most one occurrence of an operation symbol (e.g.,  $f(x, y) = g(x)$ , or  $h(x, y, x) = x$ ). An identity  $t_1 = t_2$  is said to be *balanced*<sup>1</sup> if the set of variables occurring in  $t_1$  and  $t_2$  are the same. For example, both  $f(x, x, y) = g(y, y, x)$  and  $f(x, x, x) = x$  are balanced identities. A set  $\Sigma$  of identities is linear if it only contains linear identities, idempotent if for each operation symbol,  $f$ , the identity  $f(x, x, \dots, x) = x$  is in  $\Sigma$  and balanced if all of the identities in  $\Sigma$  are balanced.

Note that the identities defining WNU, symmetric and cyclic operations above are linear and balanced. The identities defining edge operations, on the other hand, are linear but *not* balanced.

We now give some examples of results about the VCSP that are described using identities. We start with the notion of rigid cores [33].

**Definition 8.** A valued constraint language  $\Gamma$  is a *rigid core* if the only unary operation in  $\text{supp}(\Gamma)$  is the identity operation.

It is known that with respect to tractability it suffices to consider valued constraint languages that are *rigid cores*. Indeed, for every valued constraint language  $\Gamma$  which is not a rigid core there is another language  $\Gamma'$  which is a rigid core<sup>2</sup> and with the property that  $\text{VCSP}(\Gamma)$  is polynomial-time equivalent to  $\text{VCSP}(\Gamma')$  [33]. It is also known that  $\Gamma$  is a rigid core if and only if all operations from  $\text{supp}(\Gamma)$  are idempotent [33].

The “algebraic dichotomy conjecture” [11], a refinement of the dichotomy conjecture for the CSP [20], can be re-stated as follows [11, 37]: for a rigid core crisp language  $\Gamma$ ,  $\text{CSP}(\Gamma)$  is tractable if  $\Gamma$  admits a WNU polymorphism of some arity, and **NP**-complete otherwise. Equivalently,  $\text{CSP}(\Gamma)$  is tractable if  $\Gamma$  admits a cyclic polymorphism of some arity, and is **NP**-complete otherwise.

The “bounded-width theorem” for the CSP can be restated as follows [35, 4, 9, 13]: for a rigid core crisp language  $\Gamma$ , the problem  $\text{CSP}(\Gamma)$  has bounded width (and thus can be solved using local consistency methods) if and only if  $\Gamma$  has WNU polymorphisms of all arities.

<sup>1</sup> This notion of balanced identity is not related to the balanced digraphs introduced in Section 5.

<sup>2</sup>  $\Gamma'$  is the restriction of  $\Gamma$  to a subset  $D'$  of the domain of  $\Gamma$  together with the unary relations  $u_d$  for every  $d \in D'$ , where  $u_d$  is defined by  $u_d(d) = 0$  and  $u_d(x) = \infty$  if  $x \neq d$ .



There is an algorithmic technique for the CSP that 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 [10, 26]. 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). Crisp languages where this algorithm is guaranteed to find a solution (or show that none exists) were captured in [26]: for a crisp language  $\Gamma$ , the problems  $\text{CSP}(\Gamma)$  are solvable using the few subpowers algorithm if  $\Gamma$  admits an edge polymorphism of some arity. In fact the converse to this theorem is true in the following sense: the absence of edge polymorphisms of  $\Gamma$  implies that the presence of small enough representations is not guaranteed [26].

For finite-valued constraint languages, the following complexity classification has been obtained [51]: for a finite-valued constraint language  $\Gamma$ ,  $\text{VCSP}(\Gamma)$  is tractable if  $\text{supp}(\Gamma)$  contains a binary symmetric operation, and is **NP**-complete otherwise.

The power of the basic linear programming relaxation has been characterised as follows [31]: for a valued constraint language  $\Gamma$ , the problem  $\text{VCSP}(\Gamma)$  is solvable optimally by the basic linear programming relaxation if and only if  $\text{supp}(\Gamma)$  contains symmetric operations of all arities.

The power of constant-level Sherali-Adams linear programming relaxations has been characterised as follows [52]: for a valued constraint language  $\Gamma$ , the problem  $\text{VCSP}(\Gamma)$  is solvable optimally by a constant-level of the Sherali-Adams linear programming relaxation [46] if and only if the problem  $\text{VCSP}(\Gamma)$  is solvable optimally by the *third* level of the Sherali-Adams linear programming relaxation if and only if  $\text{supp}(\Gamma)$  contains WNU operations of all arities.

### 3 Reduction to a Single Combined Cost Function

Throughout this paper we will make use of the following simple but useful observation about arbitrary finite languages.

**Proposition 1.** *For any valued constraint language  $\Gamma$  such that  $|\Gamma|$  is finite, there is a single cost function  $\phi_\Gamma$  over the same domain such that:*

1.  $\text{Pol}(\Gamma) = \text{Pol}(\{\phi_\Gamma\})$ ;
2.  $\text{fPol}(\Gamma) = \text{fPol}(\{\phi_\Gamma\})$ ;
3.  $\text{VCSP}(\Gamma)$  and  $\text{VCSP}(\{\phi_\Gamma\})$  are polynomial-time equivalent.

*Proof.* Let  $\Gamma$  consist of  $q$  cost functions,  $\phi_1, \dots, \phi_q$ , with arities  $m_1, \dots, m_q$ , respectively. Without loss of generality, we assume that none of the  $\phi_i$  are the constant function  $\infty$ . Let  $m = \sum_{i=1}^q m_i$ . Define the cost function  $\phi_\Gamma$ , with arity  $m$ , by setting  $\phi_\Gamma(x_1, \dots, x_m) = \phi_1(x_1, \dots, x_{m_1}) + \phi_2(x_{m_1+1}, \dots, x_{m_1+m_2}) + \dots + \phi_q(x_{m-m_q+1}, \dots, x_m)$ , and set  $\Gamma_c = \{\phi_\Gamma\}$ .

Since the operations in  $\text{Pol}(\Gamma)$  are applied co-ordinatewise, it follows easily from Definition 6 that  $\text{Pol}(\Gamma) = \text{Pol}(\{\phi_\Gamma\})$ , and since inequalities are preserved by addition, it follows easily from Definition 7 that  $\text{fPol}(\Gamma) = \text{fPol}(\{\phi_\Gamma\})$ .

For any instance  $\mathcal{I}$  of  $\text{VCSP}(\Gamma)$  we can obtain an equivalent instance  $\mathcal{I}'$  of  $\text{VCSP}(\{\phi_\Gamma\})$  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  $\mathcal{I}$  can then be obtained by taking the assignments that minimise the objective function of  $\mathcal{I}'$  and restricting them to the variables of  $\mathcal{I}$ .

Conversely, any instance  $\mathcal{I}'$  of  $\text{VCSP}(\{\phi_\Gamma\})$  can clearly be expressed as an instance of  $\text{VCSP}(\Gamma)$  since each constraint in  $\mathcal{I}'$  can be expressed as a sum of constraints whose constraint cost functions are contained in  $\Gamma$ .  $\square$

## 4 Reduction by the Dual Encoding

In this section we will describe the dual encoding introduced in [19] for the CSP and later extended in [36] to the VCSP.

### 4.1 From a language $\Gamma$ to a binary language $\Gamma_d$

**Definition 9.** Let  $\Gamma$  be any valued constraint language over  $D$ , such that  $|\Gamma|$  is finite, and let  $\phi_\Gamma$  be the corresponding single cost function, of arity  $m$ , as defined in Proposition 1.

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

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

where  $\phi'_\Gamma : D' \rightarrow \mathbb{Q}$  is the unary finite-valued cost function on  $D'$  defined by  $\phi'_\Gamma(\mathbf{x}) = \phi_\Gamma(x_1, \dots, x_m)$  for every  $\mathbf{x} = (x_1, \dots, x_m) \in D'$ , and each  $match_{i,j} : D' \times D' \rightarrow \mathbb{Q}$  is the binary 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  $\Gamma_d$  contains a single unary cost function, which returns only finite values, together with  $m^2$  binary relations and hence is a binary valued constraint language.

**Example 2.** Let  $\Gamma = \{\phi_{eq}\}$ , where  $\phi_{eq}$  is the equality relation on  $D$ , i.e.,  $\phi_{eq} : D \times D \rightarrow \overline{\mathbb{Q}}$  is defined by  $\phi_{eq}(x, y) = 0$  if  $x = y$  and  $\phi_{eq}(x, y) = \infty$  if  $x \neq y$ .

Then  $D' = \text{Feas}(\phi_{eq}) = \{(a, a) \mid a \in D\}$  and  $\Gamma_d$  consists of a single unary finite-valued cost function  $\phi'_{eq}$ , together with four binary relations  $match_{1,1}, match_{1,2}, match_{2,1}$ , and  $match_{2,2}$ .

Moreover,  $\phi'_{eq}(\mathbf{x}) = 0$  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 3.** Let  $\Gamma = \{\phi_{sum}\}$ , where  $\phi_{sum} : \{0, 1\}^3 \rightarrow \overline{\mathbb{Q}}$  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  $D' = \text{Feas}(\phi_{sum}) = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$  and  $\Gamma_d$  consists of a single unary finite-valued cost function  $\phi'_{sum}$ , together with nine binary relations  $match_{1,1}, match_{1,2}, match_{1,3}, \dots, match_{3,3}$ .

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

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

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

and so on.

## 4.2 The dual encoding using $\Gamma_d$

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

**Definition 10.** Let  $\Gamma$  be any valued constraint language over  $D$ , such that  $|\Gamma|$  is finite, and let  $\phi_\Gamma$  be the corresponding single cost function, of arity  $m$ , as defined in Proposition 1. Let  $\mathcal{I}$  be an arbitrary instance of VCSP( $\{\phi_\Gamma\}$ ) with variables  $X = \{x_1, \dots, x_n\}$ , domain  $D$ , and constraints  $\phi_\Gamma(\mathbf{x}_1), \dots, \phi_\Gamma(\mathbf{x}_q)$ , where  $\mathbf{x}_i \in X^m$  for all  $1 \leq i \leq q$ .

The *dual* of  $\mathcal{I}$ , denoted  $\mathcal{I}_d$ , is defined to be the following instance of VCSP( $\Gamma_d$ ):

- The variables  $V' = \{x'_1, \dots, x'_q\}$  of  $\mathcal{I}_d$  are the constraints of  $\mathcal{I}$ .
- The domain of  $\mathcal{I}_d$  is  $D' = \text{Feas}(\phi_\Gamma) \subseteq D^m$ .
- For every  $1 \leq i \leq q$ , there is a unary constraint  $\phi'_\Gamma(x'_i)$ , where  $\phi'_\Gamma : D' \rightarrow \mathbb{Q}$  is as defined in Definition 9.
- If the scopes of two constraints of  $\mathcal{I}$ , say  $\phi_\Gamma(\mathbf{x}_i)$  and  $\phi_\Gamma(\mathbf{x}_j)$ , overlap, then there are binary constraints between  $x'_i$  and  $x'_j$  enforcing equality at the overlapping coordinate positions. More specifically, if  $\mathbf{x}_i = (x_{i_1}, \dots, x_{i_m})$ ,  $\mathbf{x}_j = (x_{j_1}, \dots, x_{j_m})$ , and  $\text{vars}(\mathbf{x}_i) \cap \text{vars}(\mathbf{x}_j) \neq \emptyset$  then there is a binary constraint  $\text{match}_{k,l}(x'_i, x'_j)$  for every  $k, l \in \{1, \dots, m\}$  with  $i_k = j_l$ .

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

**Proposition 2.** *For any valued constraint language  $\Gamma$  such that  $|\Gamma|$  is finite, if  $\Gamma_d$  is the dual of  $\Gamma$ , then VCSP( $\Gamma$ ) and VCSP( $\Gamma_d$ ) are polynomial-time equivalent.*

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

Hence, for any instance  $\mathcal{I}$  of VCSP( $\Gamma$ ) we can construct in polynomial time the dual instance  $\mathcal{I}_d$  in VCSP( $\Gamma_d$ ), as defined above (Definition 10). It is straightforward to show that the assignments that minimise the objective function of  $\mathcal{I}_d$  correspond precisely to the assignments that minimise the objective function of  $\mathcal{I}$ , and hence we have a polynomial-time reduction from VCSP( $\Gamma$ ) to VCSP( $\Gamma_d$ ).

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

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

We have constructed an instance  $\mathcal{I}$  in  $\text{VCSP}(\{\phi_\Gamma, \text{Feas}(\phi_\Gamma)\})$  such that assignments minimising the objective function of  $\mathcal{I}$  correspond precisely to assignments minimising the objective function of  $\mathcal{I}'$ . Hence we have established a polynomial-time reduction from  $\text{VCSP}(\Gamma_d)$  to  $\text{VCSP}(\Gamma \cup \{\text{Feas}(\phi_\Gamma)\})$ .

However, it follows from the proof of [16, Theorem 4.3] that  $\text{VCSP}(\Gamma \cup \{\text{Feas}(\phi_\Gamma)\})$  can be reduced to  $\text{VCSP}(\Gamma)$  in polynomial time.  $\square$

### 4.3 Preservation of algebraic properties

Our next result shows that the polymorphisms of  $\Gamma_d$  are very closely related to the polymorphisms of  $\Gamma$ .

**Theorem 1.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite, and let  $\Gamma_d$  be the dual of  $\Gamma$ . There is a one-to-one correspondence between the polymorphisms of  $\Gamma$  and the polymorphisms of  $\Gamma_d$ , defined as follows. For any  $f \in \text{Pol}^{(k)}(\Gamma)$  the corresponding operation  $f_d \in \text{Pol}^{(k)}(\Gamma_d)$  is defined by  $f_d(\mathbf{x}_1, \dots, \mathbf{x}_k) = f(\mathbf{x}_1, \dots, \mathbf{x}_k)$  for all  $\mathbf{x}_i$  in the domain of  $\Gamma_d$ .*

*Proof.* By Proposition 1 we may assume that  $\Gamma$  consists of a single cost function  $\phi_\Gamma : D^m \rightarrow \overline{\mathbb{Q}}$ , and hence that the domain  $D'$  of  $\Gamma_d$  is a subset of  $D^m$ .

First, consider any  $f : D^k \rightarrow D \in \text{Pol}^{(k)}(\Gamma)$ , and the corresponding  $f_d : (D')^k \rightarrow D'$  given by  $f_d(\mathbf{x}_1, \dots, \mathbf{x}_k) = f(\mathbf{x}_1, \dots, \mathbf{x}_k)$  for all  $\mathbf{x}_i \in D'$ . Since  $f$  is a polymorphism of  $\phi_\Gamma$ , it is also a polymorphism of the unary cost function  $\phi'_\Gamma$  in  $\Gamma_d$ . It is straightforward to check that  $f_d$  is also a polymorphism of all binary  $\text{match}_{i,j}$  relations in  $\Gamma_d$  (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  $f_d : (D')^k \rightarrow D' \in \text{Pol}^{(k)}(\Gamma_d)$ . Since  $f_d$  is a polymorphism of  $\text{match}_{i,i}$  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  $\text{match}_{i,j}$ , the functions  $g_i$  and  $g_j$  must return the same results for all possible arguments from  $D'$ . Hence, there is a single function  $g : D^k \rightarrow D$  such that the result returned by  $f_d(\mathbf{x}_1, \dots, \mathbf{x}_k)$  is equal to  $g(\mathbf{x}_1, \dots, \mathbf{x}_k)$ . Now, since  $f_d$  must return an element of  $D'$ , it follows that  $g$  must be a polymorphism of  $\phi_\Gamma$ , which gives the result.  $\square$

The individual cost functions in  $\Gamma_d$  often have other polymorphisms, that are not of the form indicated in Theorem 1, but the only polymorphisms that are shared by every cost function in  $\Gamma_d$  are those that correspond to polymorphisms of  $\Gamma$  in this way, as the next example illustrates.

**Example 4.** Recall the language  $\Gamma = \{\phi_{\text{sum}}\}$ , defined in Example 3.

The cost function  $\phi_{\text{sum}}$  has no polymorphisms, except for the projection operations on  $D = \{0, 1\}$ .

However, the unary finite-valued cost function  $\phi'_{\text{sum}}$ , has *every* operation on  $D' = \{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$  as a polymorphism.

The binary relation  $\text{match}_{1,1}$  has many operations on  $D'$  as polymorphisms, including all of the constant operations.

The binary relation  $\text{match}_{1,2}$  also has many operations on  $D'$  as polymorphisms, including the ternary *majority* operation  $g$  defined by

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

but not including the constant operation returning the label  $\mathbf{a}$ , or the constant operation returning the label  $\mathbf{b}$ .

Continuing in this way it can be shown that the only operations that are polymorphisms of *every* cost function in  $\Gamma_d$  are the projection operations on  $D'$ .

One simple consequence of Theorem 1 is that the polymorphisms of  $\Gamma$  and the polymorphisms of  $\Gamma_d$  satisfy exactly the same identities.

**Corollary 1.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite, and let  $\Gamma_d$  be the dual of  $\Gamma$ . Then the operations in  $\text{Pol}(\Gamma)$  and the operations in  $\text{Pol}(\Gamma_d)$  satisfy exactly the same identities.*

**Corollary 2.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite, and let  $\Gamma_d$  be the dual of  $\Gamma$ . Then  $\Gamma$  is a rigid core if and only if  $\Gamma_d$  is a rigid core.*

*Proof.* Follows immediately from Corollary 1, since the property of being idempotent is specified by an identity, as discussed in Section 2.3.  $\square$

Following our discussion in Section 2.3, Corollary 1 shows that the property of being solvable using local consistency methods or by the few subpowers algorithm is possessed by a language  $\Gamma$  if and only if it is also possessed by the associated binary language  $\Gamma_d$ .

Although the polymorphisms of  $\Gamma$  and  $\Gamma_d$  satisfy the same identities, the polymorphisms of  $\Gamma_d$  do not, in general, have *all* the same properties as the polymorphisms of  $\Gamma$ . For example,  $\text{Pol}(\Gamma)$  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  $\text{Pol}(\Gamma_d)$  is *not* generally conservative, since, for example,  $\min_d((a, b), (b, a)) = (a, a)$  for all  $a < b$ .

Our next result shows that the fractional polymorphisms of  $\Gamma_d$  are closely related to the fractional polymorphisms of  $\Gamma$ .

**Theorem 2.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite, and let  $\Gamma_d$  be the dual of  $\Gamma$ . There is a one-to-one correspondence between the fractional polymorphisms of  $\Gamma$  and the fractional polymorphisms of  $\Gamma_d$ , defined as follows. For any  $\omega : \text{Pol}^{(k)}(\Gamma) \rightarrow \mathbb{Q}_{\geq 0} \in \text{fPol}^{(k)}(\Gamma)$  the corresponding function  $\omega_d : \text{Pol}^{(k)}(\Gamma_d) \rightarrow \mathbb{Q}_{\geq 0} \in \text{fPol}^{(k)}(\Gamma_d)$  is defined by  $\omega_d(f_d) = \omega(f)$  for all  $f \in \text{Pol}^{(k)}(\Gamma)$  and their corresponding operations  $f_d \in \text{Pol}^{(k)}(\Gamma_d)$  (as defined in Theorem 1).*

*Proof.* By Proposition 1 we may assume that  $\Gamma$  consists of a single cost function  $\phi_\Gamma : D^m \rightarrow \overline{\mathbb{Q}}$ , and hence that the domain  $D'$  of  $\Gamma_d$  is a subset of  $D^m$ .

First, consider any  $\omega : \text{Pol}^{(k)}(\Gamma) \rightarrow \mathbb{Q}_{\geq 0} \in \text{fPol}^{(k)}(\Gamma)$ , and the corresponding  $\omega_d : \text{Pol}^{(k)}(\Gamma_d) \rightarrow \mathbb{Q}_{\geq 0}$  given by  $\omega_d(f_d) = \omega(f)$  for all  $f \in \text{Pol}^{(k)}(\Gamma)$ . Since  $\omega$  is a fractional polymorphism of  $\phi_\Gamma$ , it is easy to check that  $\omega_d$  satisfies the conditions in Definition 7, and hence is a fractional polymorphism of the unary cost function  $\phi'_\Gamma$  in  $\Gamma_d$ . Since all other cost functions in  $\Gamma_d$  are the  $\text{match}_{i,j}$  relations, the inequality condition in Definition 7 holds trivially for all these cost functions, and hence  $\omega_d$  is a fractional polymorphism of all cost functions in  $\Gamma_d$ .

Now consider any  $\omega_d : \text{Pol}^{(k)}(\Gamma_d) \rightarrow \mathbb{Q}_{\geq 0} \in \text{fPol}^{(k)}(\Gamma_d)$ . Since  $\omega_d$  is a fractional polymorphism of  $\phi'_\Gamma$ , the function  $\omega : \text{Pol}^{(k)}(\Gamma) \rightarrow \mathbb{Q}_{\geq 0}$  that assigns the same weights to corresponding elements of  $\text{Pol}^{(k)}(\Gamma)$  satisfies the condition of Definition 7, and hence is a fractional polymorphism of  $\phi_\Gamma$ .  $\square$

Following our discussion in Section 2.3, combining Corollary 1 with Theorem 2 shows that the property of being solvable using the basic linear programming relaxation or by constant levels of the Sherali-Adams linear programming relaxations is possessed by a language  $\Gamma$  if and only if it is also possessed by the associated binary language  $\Gamma_d$ .

## 5 Reduction by the Extended Dual Encoding

In this section we will describe our new extension of the reduction from [14] to the VCSP.

### 5.1 From a language $\Gamma$ to a binary language $\Gamma_e$

Throughout this section it will be helpful to view a binary relation on a set as a directed graph (digraph) where the vertices are the elements of the set, and the directed edges are the binary tuples in the relation.

First we introduce some simple definitions relating to digraphs that we will need in our constructions. We define a digraph as a structure  $\mathbb{G} = (V^{\mathbb{G}}, E^{\mathbb{G}})$  with vertices  $v \in V^{\mathbb{G}}$  and directed edges  $e \in E^{\mathbb{G}}$ . We will sometimes write the directed edge  $(a, b) \in E^{\mathbb{G}}$  as  $a \rightarrow b$ .

**Definition 11.** A digraph is an *oriented path* if it consists of a sequence of vertices  $v_0, v_1, \dots, v_k$  such that precisely one of  $(v_{i-1}, v_i), (v_i, v_{i-1})$  is an edge, for each  $i = 1, \dots, k$ .

We now adapt the construction from [14] to valued constraint languages. The construction makes use of *zigzags*, where a zigzag is the oriented path  $\bullet \rightarrow \bullet \leftarrow \bullet \rightarrow \bullet$ . The important property we will use is that there is a surjective homomorphism from a zigzag to a single edge but not from a single edge to a zigzag.

**Definition 12.** Let  $\Gamma$  be any valued constraint language over  $D$ , such that  $|\Gamma|$  is finite, and let  $\phi_{\Gamma}$  be the corresponding single cost function, of arity  $m$ , as defined in Proposition 1. As before, we define  $D' = \text{Feas}(\phi_{\Gamma}) \subseteq D^m$ .

The *extended dual* of  $\Gamma$ , denoted  $\Gamma_e$ , is the binary valued constraint language  $\{\mathbb{D}_{\Gamma}, \mu_{\Gamma}\}$ , where  $\mathbb{D}_{\Gamma}$  is a binary relation, and  $\mu_{\Gamma}$  is a unary cost function, as defined below.

For  $S \subseteq \{1, 2, \dots, m\}$  define  $\mathbb{Q}_{S,i}$  to be a single edge if  $i \in S$ , and a zigzag if  $i \in \{1, 2, \dots, m\} \setminus S$ . Now define the oriented path  $\mathbb{Q}_S$  by

$$\mathbb{Q}_S = \bullet \rightarrow \bullet \dot{+} \mathbb{Q}_{S,1} \dot{+} \mathbb{Q}_{S,2} \dot{+} \dots \dot{+} \mathbb{Q}_{S,m} \dot{+} \bullet \rightarrow \bullet$$

where  $\dot{+}$  denotes the concatenation of paths.

To define the digraph  $\mathbb{D}_{\Gamma}$ , consider the binary relation  $D \times D'$  as a digraph, and replace each edge  $(d, \mathbf{x})$  with the oriented path  $\mathbb{Q}_{\{i \mid \mathbf{x}[i]=d\}}$ . The resulting digraph  $\mathbb{D}_{\Gamma}$  has vertex set  $V^{\mathbb{D}_{\Gamma}} = D \cup D' \cup E$ , where  $E$  consists of all the additional internal vertices from the oriented paths  $\mathbb{Q}_S$ .

Finally, let  $\mu_{\Gamma}$  be the unary cost function on  $V^{\mathbb{D}_{\Gamma}}$  such that

$$\mu_{\Gamma}(v) = \begin{cases} \phi_{\Gamma}(v) & \text{if } v \in D' \\ 0 & \text{otherwise.} \end{cases}$$

The language  $\Gamma_e$  contains a single binary relation  $\mathbb{D}_{\Gamma}$ , together with a unary cost function  $\mu_{\Gamma}$ , which returns only finite values, and hence is a binary valued constraint language with domain  $V^{\mathbb{D}_{\Gamma}}$ .

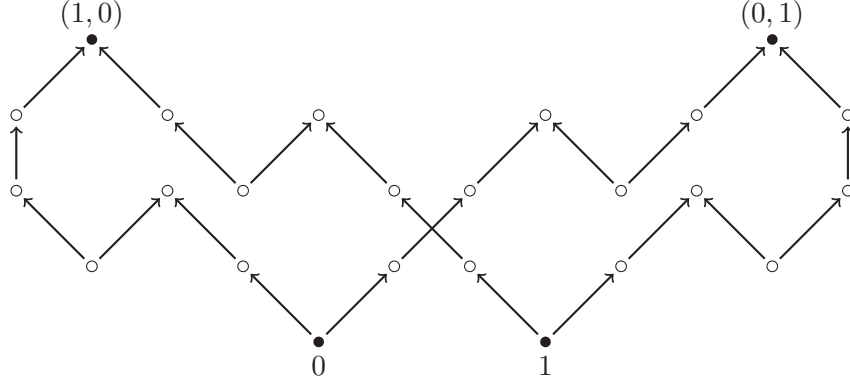


Figure 1: The digraph  $\mathbb{D}_\Gamma$  built from the valued constraint language  $\Gamma$  described in Example 5.

**Example 5.** Consider the valued constraint language  $\Gamma$  over the domain  $D = \{0, 1\}$  containing the single (binary) cost function

$$\rho(x, y) = \begin{cases} 2 & \text{if } (x, y) = (0, 1) \\ 1 & \text{if } (x, y) = (1, 0) \\ \infty & \text{otherwise.} \end{cases}$$

The digraph  $\mathbb{D}_\Gamma$  constructed from  $\rho$  is shown in Figure 5.1. The unary cost function built from  $\rho$  is

$$\mu_\Gamma(v) = \begin{cases} 2 & \text{if } v = (0, 1) \\ 1 & \text{if } v = (1, 0) \\ 0 & \text{otherwise} \end{cases}$$

for every vertex  $v \in V^{\mathbb{D}_\Gamma}$ .

The binary relation  $\mathbb{D}_\Gamma$  defined in Definition 12 is identical to the digraph defined in [14, Definition 3.2], where it is shown that the number of vertices in  $\mathbb{D}_\Gamma$  is  $(3n + 1)|D'| |D| + (1 - 2n)|D'| + |D|$  and the number of edges is  $(3n + 2)|D'| |D| - 2n|D'|$ . Also, as noted in [14], this construction can be performed in polynomial time.

## 5.2 The extended dual encoding using $\Gamma_e$

We now show how to reduce instances of  $\text{VCSP}(\Gamma)$  to instances of  $\text{VCSP}(\Gamma_e)$  using a construction that we call the extended dual encoding. This construction is similar in overall structure to the hidden variable encoding described in [42], but has only one form of binary constraint.

**Definition 13.** Let  $\Gamma$  be any valued constraint language over  $D$ , such that  $|\Gamma|$  is finite, and let  $\phi_\Gamma$  be the corresponding single cost function, of arity  $m$ , as defined in Proposition 1. Let  $\mathcal{I}$  be an arbitrary instance of  $\text{VCSP}(\{\phi_\Gamma\})$  with variables  $X = \{x_1, \dots, x_n\}$ , domain  $D$ , and constraints  $\phi_\Gamma(\mathbf{x}_1), \dots, \phi_\Gamma(\mathbf{x}_q)$ , where  $\mathbf{x}_i \in X^m$  for all  $1 \leq i \leq q$ .

The *extended dual* of  $\mathcal{I}$ , denoted  $\mathcal{I}_e$ , is defined to be the following instance of  $\text{VCSP}(\Gamma_e)$ :

- The variables of  $\mathcal{I}_e$  are  $X \cup \{x'_1, \dots, x'_q\} \cup Y$  where  $\{x'_1, \dots, x'_q\}$  correspond to the constraints of  $\mathcal{I}$  and  $Y$  contains additional variables as described below.

- The domain of  $\mathcal{I}_e$  is the same as the domain of  $\Gamma_e$ , as defined in Definition 12; that is,  $D \cup D' \cup E$ , where  $D' = \text{Feas}(\phi_\Gamma) \subseteq D^m$  and  $E$  contains the additional vertices of  $\mathbb{D}_\Gamma$ .
- For every  $1 \leq i \leq q$ , there is a unary constraint  $\mu_\Gamma(x'_i)$ , where  $\mu_\Gamma$  is as defined in Definition 12.
- For each constraint  $\phi_\Gamma(\mathbf{x}_i)$  of  $\mathcal{I}$ , where  $\mathbf{x}_i = (x_{i_1}, \dots, x_{i_m})$ , there is an oriented path  $\mathbb{Q}_{\{j\}}$  from each  $x_{i_j}$  to  $x'_i$  (where  $\mathbb{Q}_S$  for any  $S \subseteq \{1, 2, \dots, m\}$  is as defined in Definition 12). Each such path uses disjoint sets of intermediate vertices, and each oriented edge on these paths, say  $(y, y')$ , is the scope of a constraint in  $\mathcal{I}_e$  with relation  $\mathbb{D}_\Gamma$ . The set  $Y$  is the union of all such intermediate vertices over all such paths.

To verify that the extended dual of  $\mathcal{I}$  gives a reduction from  $\text{VCSP}(\Gamma)$  to  $\text{VCSP}(\Gamma_e)$  we introduce the following terminology.

Given any digraph  $\mathbb{G}$ , we can define an associated undirected graph  $\mathbb{G}^*$  where each directed edge of  $\mathbb{G}$  is replaced by an undirected edge on the same pair of vertices. We will say that a digraph  $\mathbb{G}$  is connected if  $\mathbb{G}^*$  is connected, and we will define the connected components of  $\mathbb{G}$  to be the connected components of  $\mathbb{G}^*$ .

For any digraph  $\mathbb{G}$ , if  $\mathbb{G}^*$  contains a cycle, then the corresponding set of directed edges in  $\mathbb{G}$  will be called an oriented cycle. The length of an oriented cycle is defined as being the absolute value of the difference between the number of edges oriented in one direction around the cycle and edges oriented in the opposite direction. A connected digraph  $\mathbb{G}$  is said to be *balanced* if all of its oriented cycles have zero length [20].

Note that the digraph  $\mathbb{D}_\Gamma$  described in Definition 12 is balanced. Moreover, the binary scopes of the extended dual instance  $\mathcal{I}_e$  constructed in Definition 13 also form a balanced digraph which we will call  $\mathbb{G}_{\mathcal{I}_e}$  (if  $\mathbb{G}_{\mathcal{I}_e}$  is not connected then we may consider each connected component separately).

The vertices of any balanced digraph  $\mathbb{G}$  can be organised into *levels*, which are non-negative integers given by a function  $lvl$  such that for every directed edge  $(a, b) \in E^{\mathbb{G}}$ ,  $lvl(b) = lvl(a) + 1$ . The minimum level of  $\mathbb{G}$  is 0, and the top level is called the *height* of  $\mathbb{G}$ .

Any feasible solution to  $\mathcal{I}_e$  must assign to each vertex  $x$  in  $\mathbb{G}_{\mathcal{I}_e}$  a label  $d_x$  chosen from the vertices of  $\mathbb{D}_\Gamma$ , which must be *at the same level* as  $x$ .

Every variable  $x_i \in X$  of  $\mathcal{I}_e$  is at level 0 in  $\mathbb{G}_{\mathcal{I}_e}$ , and so any feasible solution to  $\mathcal{I}_e$  must assign to  $x_i$  a label at level 0 in  $\mathbb{D}_\Gamma$ , that is, an element  $d(x_i)$  of  $D$ . Similarly, every variable  $x'_j$  of  $\mathcal{I}_e$  is at level  $m + 2$  in  $\mathbb{G}_{\mathcal{I}_e}$ , and so any feasible solution to  $\mathcal{I}_e$  must assign to  $x'_j$  a label at level  $m + 2$  in  $\mathbb{D}_\Gamma$ , that is, an element  $d(x'_j)$  of  $D' = \text{Feas}(\phi_\Gamma)$ .

Every other variable  $y$  of  $\mathcal{I}_e$  lies on an oriented path of the form  $\mathbb{Q}_{\{k\}}$  from some  $x_i$  to  $x'_j$ , and so any feasible solution to  $\mathcal{I}_e$  must assign to all variables on this oriented path a label on some fixed oriented path of the form  $\mathbb{Q}_S$  in  $\mathbb{D}_\Gamma$ . By the construction of the oriented paths  $\mathbb{Q}_S$  (see Definition 12), the labels assigned to the variables in a path of the form  $\mathbb{Q}_{\{k\}}$  must lie in an oriented path of the form  $\mathbb{Q}_S$  for some set  $S$  that contains the index  $k$  [14, Observation 3.1]. By Definition 12, such a path exists in  $\mathbb{D}_\Gamma$  if and only if  $d(x_i) = d(x'_j)[k]$ .

Hence there is a one-to-one correspondence between feasible solutions to  $\mathcal{I}_e$  and feasible solutions to  $\mathcal{I}$ . The cost of each feasible solution to  $\mathcal{I}_e$  is determined by the sum of the values given by the cost function  $\mu_\Gamma$  for the labels assigned to the variables  $x'_i$ , and hence is equal to the cost of the corresponding solution to  $\mathcal{I}$ . Hence the extended dual encoding specified in Definition 13 provides a way to reduce instances of  $\text{VCSP}(\Gamma)$  to instances of  $\text{VCSP}(\Gamma_e)$ .

Our next result extends this observation to obtain the reverse reduction as well.



**Theorem 3.** *For any valued constraint language  $\Gamma$  such that  $|\Gamma|$  is finite, if  $\Gamma_e$  is the extended dual of  $\Gamma$ , then  $\text{VCSP}(\Gamma)$  and  $\text{VCSP}(\Gamma_e)$  are polynomial-time equivalent.*

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

Hence, for any instance  $\mathcal{I}$  of  $\text{VCSP}(\Gamma)$  we can construct in polynomial time the extended dual instance  $\mathcal{I}_e$  in  $\text{VCSP}(\Gamma_e)$  as described in Definition 13. As we have just shown, the assignments that minimise the objective function of  $\mathcal{I}_e$  correspond precisely to the assignments that minimise the objective function of  $\mathcal{I}$ , and hence we have a polynomial-time reduction from  $\text{VCSP}(\Gamma)$  to  $\text{VCSP}(\Gamma_e)$ .

For the other direction, given any instance  $\mathcal{I}'$  in  $\text{VCSP}(\Gamma_e)$  we now indicate how to solve it in polynomial time, or else construct in polynomial time a corresponding instance  $\mathcal{I}'_d$  in  $\text{VCSP}(\Gamma_d)$ . We can then appeal to Proposition 2.

Consider the digraph  $\mathbb{G}$  formed by the binary scopes of  $\mathcal{I}'$ . Since each connected component can be considered separately, we may assume that  $\mathbb{G}$  is connected. Moreover, if  $\mathbb{G}$  is not balanced, then  $\mathcal{I}'$  has no feasible solutions, so we may assume that  $\mathbb{G}$  is balanced (which can be checked in polynomial time).

Any feasible solution to  $\mathcal{I}'$  must assign each vertex in  $\mathbb{G}$  a label chosen from the vertices of  $\mathbb{D}_\Gamma$ , in a way which preserves the differences in levels between different vertices. Hence if the height of  $\mathbb{G}$  is greater than the height of  $\mathbb{D}_\Gamma$ , then  $\mathcal{I}'$  has no feasible solutions, so we may assume that the height of  $\mathbb{G}$  is less than or equal to the height of  $\mathbb{D}_\Gamma$ .

Now consider the case when  $\mathbb{G}$  is balanced and of height  $h$  which is strictly less than the height of  $\mathbb{D}_\Gamma$ . In this case every vertex of  $\mathbb{G}$  must be assigned a vertex in some induced sub-graph of  $\mathbb{D}_\Gamma$  which is connected and of height  $h$ . For a fixed  $\mathbb{D}_\Gamma$ , there are a fixed number of such sub-graphs, and they all have one of three forms:

- An oriented path which is a subpath of  $\mathbb{Q}_S$ , for some set  $S$ , as defined in Definition 12;
- A collection of such oriented paths which all share their initial vertex (and no others);
- A collection of such oriented paths which all share their final vertex (and no others).

In all three cases we can order the vertices of the subgraph by increasing level in  $\mathbb{D}_\Gamma$ , and within that by which path they belong to (when there is more than one), and within that by the distance along the path. With the vertices ordered in this way, the subgraph has the property that for all edges  $(a, b), (c, d)$  with  $a < c$  we have  $b \leq d$ , so it admits the binary polymorphisms  $\min$  and  $\max$ . Together with the fact that any unary cost function is submodular for any ordering of the domain, it follows that in all such cases the corresponding valued constraint language is submodular, and an optimal solution can be found in polynomial time [15].

Finally, we consider the case when  $\mathbb{G}$  is balanced and has the same height as  $\mathbb{D}_\Gamma$ . In this case only vertices at the top level in  $\mathbb{G}$  can be assigned labels at the top level in  $\mathbb{D}_\Gamma$ . Let these vertices of  $\mathbb{G}$  be  $x_1, x_2, \dots, x_q$ . We will build an instance  $\mathcal{I}'_d$  of  $\text{VCSP}(\Gamma_d)$  beginning with these vertices as variables.

If there is a unary constraint with cost function  $\mu_\Gamma$  on any of these vertices in  $\mathcal{I}'$ , then we add a unary constraint with cost function  $\phi'_\Gamma$  in  $\mathcal{I}'_d$ , where  $\phi'_\Gamma$  is the unary cost function defined in Definition 9. (Note that any other unary constraints on other variables in  $\mathcal{I}'$  will not affect the cost of a feasible solution, because all other variables must be assigned a label with cost 0.)

To complete the construction of  $\mathcal{I}'_d$  we will add constraints of the form  $match_{k,l}$  between pairs of vertices  $x_i$  and  $x_j$  where it can be shown from the structure of  $\mathbb{G}$  that they must be assigned labels that agree in positions  $k$  and  $l$  respectively.

To examine the structure of  $\mathbb{G}$ , consider the connected components of the induced subgraph of  $\mathbb{G}$  obtained by removing all vertices at the top level and all vertices at level 0. Each such component is a balanced digraph of height at most  $m$  which must be assigned labels from a single oriented path in  $\mathbb{D}_\Gamma$  of the form  $\mathbb{Q}_S$ , for some set  $S \subseteq \{1, \dots, m\}$ . Note that the choice of oriented path in  $\mathbb{D}_\Gamma$  is fixed by the assignment to any vertex in the component.

For any such component  $C$  there will be a unique smallest set  $S_0 \subseteq \{1, \dots, m\}$  such that any feasible solution to  $\mathcal{I}'$  can assign labels to the variables in  $C$  from the oriented path  $\mathbb{Q}_{S_0}$  [14]. Moreover, it is shown in [14] that this set  $S_0$  can be computed in polynomial time (in fact, in logarithmic space). For each component  $C$  this set will be denoted<sup>3</sup> by  $S_0(C)$ .

If there are edges in  $\mathbb{G}$  from one such component  $C$  to two distinct vertices  $x_i$  and  $x_j$  at the top level of  $\mathbb{G}$ , then these vertices must be assigned the same label in any feasible solution to  $\mathcal{I}'$ , due to the structure of the paths in  $\mathbb{D}_\Gamma$ , so we add a constraint  $match_{kk}(x_i, x_j)$  to  $\mathcal{I}'_d$  for  $k = 1, 2, \dots, m$ .

Next, if there is an edge in  $\mathbb{G}$  from some component  $C$  to a vertex  $x_i$  at the top level of  $\mathbb{G}$ , and  $S_0(C)$  contains two distinct indices  $k$  and  $l$ , then the label assigned to  $x_i$  in any feasible solution to  $\mathcal{I}'$  must agree in positions  $k$  and  $l$ . Hence for each such case we add a constraint  $match_{kl}(x_i, x_i)$  to  $\mathcal{I}'_d$ .

Next, if there is an edge in  $\mathbb{G}$  from some vertex  $y_1$  at level 0 to some component  $C$ , and another edge in  $\mathbb{G}$  from some vertex  $y_2$  at level 0 to the same component  $C$ , then we know that any feasible solution to  $\mathcal{I}'$  must assign the same label to  $y_1$  and  $y_2$ , so we say that  $y_1$  and  $y_2$  are linked. Taking the reflexive, transitive closure of this linking relation gives an equivalence relation on the vertices in  $\mathbb{G}$  at level 0.

Finally, if there is an edge in  $\mathbb{G}$  from a vertex  $y_1$  at level 0 in  $\mathbb{G}$  to a component  $C_1$ , and an edge from  $C_1$  to a vertex  $x_i$  at the top level, and there is also a vertex  $y_2$  at level 0 which is equivalent to  $y_1$ , and an edge from  $y_2$  to a component  $C_2$ , and an edge from  $C_2$  to a vertex  $x_j$  at the top level in  $\mathbb{G}$ , then we proceed as follows: choose an index  $k \in S_0(C_1)$  and an index  $l \in S_0(C_2)$  and add the constraint  $match_{kl}(x_i, x_j)$  to  $\mathcal{I}'_d$ . This ensures that the label assigned to  $x_i$  in any feasible solution to  $\mathcal{I}'$  must agree in position  $k$  with the label assigned to  $x_j$  in position  $l$ .

Now we have constructed an instance  $\mathcal{I}'_d$  in  $\text{VCSP}(\Gamma_d)$  whose constraints impose precisely the same restrictions on feasible solutions as the binary constraints in  $\mathcal{I}'$  (whose scopes are specified by the edges of  $\mathbb{G}$ ). We have also imposed unary constraints on the variables of  $\mathcal{I}'_d$  to ensure that the cost of any feasible solution is the same as the cost of the corresponding feasible solution to  $\mathcal{I}'$ . Hence for any feasible solution to  $\mathcal{I}'$  there will be a feasible solution to  $\mathcal{I}'_d$  with the same cost, and vice versa, which gives the result.  $\square$

### 5.3 Preservation of Algebraic Properties

We now investigate how the polymorphisms of a valued constraint language  $\Gamma$  (with finitely many cost functions) are related to the polymorphisms of the extended dual language  $\Gamma_e$ . In the proof of Theorem 4 we will closely follow results from [14].

---

<sup>3</sup>The notation used in [14] is  $\Gamma(C)$ , but we use a different notation here to avoid confusion with the valued constraint language  $\Gamma$ .

**Theorem 4.** *Let  $\Gamma$  be any valued constraint language over  $D$ , such that  $|\Gamma|$  is finite, and let  $\Gamma_e$  be the extended dual of  $\Gamma$ . If  $\Gamma$  is a rigid core, then  $\{f_e|_D : f_e \in \text{Pol}(\Gamma_e)\} = \text{Pol}(\Gamma)$ .*

*Moreover, for each  $f \in \text{Pol}^{(k)}(\Gamma)$  there is at least one operation  $f_e \in \text{Pol}^{(k)}(\Gamma_e)$  such that  $f_e$  satisfies all linear balanced identities satisfied by  $f$  and*

$$f_e(\mathbf{x}_1, \dots, \mathbf{x}_k) = \begin{cases} f(\mathbf{x}_1, \dots, \mathbf{x}_k) & \text{if each } \mathbf{x}_i \in D' \\ \text{some label not in } D' & \text{otherwise.} \end{cases} \quad (3)$$

where  $D'$  denotes the set  $\text{Feas}(\phi_\Gamma) \subseteq D^m$  for the single cost function  $\phi_\Gamma$  defined in Proposition 1.

*Proof.* First, consider any  $f_e \in \text{Pol}(\Gamma_e)$ . If we apply the extended dual construction given in Definition 13, we obtain an instance  $\mathcal{I}_e$  of  $\text{CSP}(\Gamma_e)$  where in any feasible solution the variables at level 0 must take values from  $D$  that together form tuples from  $\text{Feas}(\phi_\Gamma)$ . Hence  $f_e|_D$  must be a polymorphism of  $\Gamma$ .

For the converse, consider any  $f \in \text{Pol}(\Gamma)$ . As noted in Section 2.3, assuming that  $\Gamma$  is a rigid core ensures that every polymorphism of  $\Gamma$  is idempotent. It is shown in [14, Proof of Theorem 5.1] that any idempotent polymorphism  $f$  of the relation  $\text{Feas}(\phi_\Gamma)$  can be extended to a polymorphism  $f_e$  of the associated digraph  $\mathbb{D}_\Gamma$  described in Definition 12 that satisfies Equation 3. Since *any* operation defined on the vertices of  $\mathbb{D}_\Gamma$  is a polymorphism of the unary finite-valued cost function  $\mu_\Gamma$  described in Definition 12, the operation  $f_e$  is a polymorphism of  $\Gamma_e$ .

Moreover, it is also shown in [14, Proof of Theorem 5.1] that  $f_e$  satisfies many of the same identities as  $f$ , including all linear balanced identities that are satisfied by the polymorphisms of the zigzag. By Lemma 5.3 of [14], *all* balanced identities are satisfied by the polymorphisms of the zigzag, so  $f_e$  satisfies all linear balanced identities satisfied by  $f$ .  $\square$

For the special case of unary polymorphisms, we can say more: Lemma 4.1 of [14] states that the unary polymorphisms of a relation and of the corresponding digraph  $\mathbb{D}_\Gamma$  are in one-to-one correspondence. Hence, we immediately get the following.

**Lemma 1.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite, and let  $\Gamma_e$  be the extended dual of  $\Gamma$ .  $\Gamma$  is a rigid core if and only if  $\Gamma_e$  is a rigid core.*

Following our discussion in Section 2.3, Theorem 4 and Lemma 1 show that if a rigid core crisp language  $\Gamma$  has the property of being solvable using local consistency methods then so does the associated binary language  $\Gamma_e$ .

Our next result shows that for finite rigid core valued constraint languages  $\Gamma$ , the fractional polymorphisms of  $\Gamma_e$  are closely related to the fractional polymorphisms of  $\Gamma$ .

**Theorem 5.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite and let  $\Gamma_e$  be the extended dual of  $\Gamma$ .*

*If  $\Gamma$  is a rigid core, then for any fractional polymorphism  $\omega$  of  $\Gamma$  there is a corresponding fractional polymorphism  $\omega_e$  of  $\Gamma_e$  such that for each  $f \in \text{supp}(\omega)$  there is a corresponding  $f_e \in \text{supp}(\omega_e)$  and vice versa. Moreover,  $\omega_e(f_e) = \omega(f)$  for all  $f_e \in \text{supp}(\omega_e)$ .*

*Proof.* By Proposition 1 we may assume that  $\Gamma$  consists of a single cost function  $\phi_\Gamma : D^m \rightarrow \overline{\mathbb{Q}}$ , where  $\text{Feas}(\phi_\Gamma) = D'$ .

Now consider any function  $\omega : \text{Pol}^{(k)}(\Gamma) \rightarrow \mathbb{Q}_{\geq 0} \in \text{fPol}^{(k)}(\Gamma)$ . By Theorem 4, for each  $f \in \text{Pol}^{(k)}(\Gamma)$  we can choose a corresponding  $f_e \in \text{Pol}^{(k)}(\Gamma_e)$  satisfying Equation 3. Hence we can define a function  $\omega_e : \text{Pol}^{(k)}(\Gamma_e) \rightarrow \mathbb{Q}_{\geq 0}$  by setting  $\omega_e(f_e) = \omega(f)$  for all  $f \in \text{Pol}^{(k)}(\Gamma)$  (and setting all other values of  $\omega_e$  to zero).

To check that  $\omega_e$  is a fractional polymorphism of  $\Gamma_e$  we only need to verify that it satisfies Equation 2 in Definition 7 for each cost function in  $\Gamma_e$ .

The language  $\Gamma_e$  contains just the binary relation  $\mathbb{D}_\Gamma$  and the unary cost function  $\mu_\Gamma$ , as specified in Definition 12. As  $\mathbb{D}_\Gamma$  is a relation, and each  $f_e$  is a polymorphism of  $\mathbb{D}_\Gamma$ , the inequality in Definition 7 is trivially satisfied by  $\mathbb{D}_\Gamma$  (both sides are equal to zero).

It remains to show that  $\omega_e$  is a fractional polymorphism of  $\mu_\Gamma$ . When applied to  $\mu_\Gamma$ , this condition says that, for any  $x_1, \dots, x_k$  in the domain of  $\Gamma_e$ , we must have

$$\sum_{f_e \in \text{Pol}^{(k)}(\Gamma_e)} \omega_e(f_e) \mu_\Gamma(f_e(x_1, \dots, x_k)) \leq \frac{1}{k} (\mu_\Gamma(x_1) + \dots + \mu_\Gamma(x_k)).$$

Recall that, by definition,  $\mu_\Gamma(x) = 0$  for all  $x \notin D'$ . By Theorem 4, it follows that if  $f_e(x_1, \dots, x_k) \in D'$  then  $x_1, \dots, x_k \in D'$ . Thus, if not all  $x_1, \dots, x_k$  are in  $D'$ , the only possible non-zero terms appear in the RHS of the inequality, and hence it is trivially true.

On the other hand, if all  $x_1, \dots, x_k$  are in  $D'$  then we have  $\mu_\Gamma(x_i) = \phi_\Gamma(x_i)$  for  $i = 1, 2, \dots, k$ . Moreover, since  $f_e(x_1, \dots, x_k) = f(x_1, \dots, x_k)$ , and  $f$  is a polymorphism of  $\Gamma$ , we have  $f_e(x_1, \dots, x_k) \in D'$  and so  $\mu_\Gamma(f_e(x_1, \dots, x_k)) = \phi_\Gamma(f(x_1, \dots, x_k))$ . In this case, the inequality holds because the inequality

$$\sum_{f \in \text{Pol}^{(k)}(\Gamma)} \omega(f) \phi_\Gamma(f(x_1, \dots, x_k)) \leq \frac{1}{k} (\phi_\Gamma(x_1) + \dots + \phi_\Gamma(x_k))$$

holds for  $\omega$ , because it is a fractional polymorphism of  $\Gamma$ . □

Following our discussion in Section 2.3, combining Theorem 4 with Theorem 5 shows that the property of being solvable using the basic linear programming relaxation is possessed by the binary language  $\Gamma_e$  if it is possessed by  $\Gamma$ . Similarly, the property of being solvable by constant levels of the Sherali-Adams linear programming relaxations is possessed by the binary language  $\Gamma_e$  if it is possessed by  $\Gamma$ .

## 5.4 Reduction to Minimum Cost Homomorphism

We have shown that for any valued constraint language with a finite number of cost functions of arbitrary arity we can construct an equivalent language with a single unary cost function and a single binary crisp cost function.

Valued constraint problems with a single binary crisp cost function, described by a digraph  $\mathbb{H}$ , can also be seen as graph homomorphism problems. In a graph homomorphism problem we are given an instance specified by a digraph  $\mathbb{G}$  and asked whether there is a mapping from the vertices of  $\mathbb{G}$  to the vertices of a fixed digraph  $\mathbb{H}$  such that adjacent vertices in  $\mathbb{G}$  are mapped to adjacent vertices in  $\mathbb{H}$ . Such a mapping is called a homomorphism from  $\mathbb{G}$  to  $\mathbb{H}$ .

If we have a VCSP instance  $\mathcal{I}$  over a language containing only a single binary relation  $\mathbb{H}$ , then it is easy to check that the feasible solutions to  $\mathcal{I}$  are precisely the homomorphisms from  $\mathbb{G}$  to  $\mathbb{H}$ , where  $\mathbb{G}$  is the digraph whose edges are the scopes of the constraints in  $\mathcal{I}$ .

If our instance  $\mathcal{I}$  also has unary finite-valued cost functions, then it is equivalent to the so-called Minimum Cost Homomorphism Problem [22], where the cost of a homomorphism is defined by a unary function on each vertex of the input that assigns a cost to each possible vertex of the target digraph. The Minimum Cost Homomorphism Problem for a fixed digraph  $\mathbb{H}$  is denoted  $\text{MinCostHom}(\mathbb{H})$ . The special case where all the unary finite-valued cost functions are chosen from some fixed set  $\Delta$  is denoted  $\text{MinCostHom}(\mathbb{H}, \Delta)$ .

The problem  $\text{MinCostHom}(\mathbb{H})$  was studied in a series of papers, and complete complexity classifications were given in [22] for undirected graphs, in [24] for digraphs, and in [49] for more general structures. Partial complexity classifications for the problem  $\text{MinCostHom}(\mathbb{H}, \Delta)$  were obtained in [50, 53, 54]. One can see that  $\text{MinCostHom}$  is an intermediate problem between CSP and VCSP, as there is an optimisation aspect, but it is limited in the sense that it is controlled by separate unary cost functions, without explicit interactions of variables.

By Theorem 3 and Lemma 1 we obtain the following corollary, which shows that a very restricted case of binary  $\text{MinCostHom}$  can express all valued constraint problems.

**Corollary 3.** *Let  $\Gamma$  be a valued constraint language such that  $|\Gamma|$  is finite and  $\Gamma$  is a rigid core. There is a balanced digraph  $\mathbb{D}_\Gamma$  which is a rigid core and a finite-valued unary cost function  $\mu_\Gamma$  such that problems  $\text{VCSP}(\Gamma)$  and  $\text{MinCostHom}(\mathbb{D}_\Gamma, \{\mu_\Gamma\})$  are polynomial-time equivalent.*

An interesting problem is to characterise which digraph homomorphism problems can capture **NP**-hard VCSPs. For the restricted case of ordinary CSPs the following result is known.

**Theorem 6** ([20]). *Every CSP is polynomial-time equivalent to a balanced digraph homomorphism problem with only 5 levels.*

Recall that an  $n$ -level digraph has height  $n - 1$ . We remark that [20] also shows that the digraph homomorphism problem for a balanced digraph with 4 levels is solvable in polynomial time.

To illustrate how the digraph homomorphism problem can capture **NP**-hard VCSPs we give an example of a 5-level digraph and unary weighted relation which can capture Max-Cut, a canonical **NP**-hard VCSP.

**Example 6.** Consider the digraph  $\mathbb{H}$  shown Figure 2. Let the unary weighted relation  $\mu(v)$  be

$$\mu(v) = \begin{cases} 1 & \text{if } v = b \text{ or } v = c \\ 0 & \text{otherwise} \end{cases}$$

for every vertex  $v \in V^{\mathbb{H}}$ .

Now consider the instance of  $\text{MinCostHom}(\mathbb{H}, \{\mu\})$  with the source digraph  $\mathbb{G}$  shown in Figure 3 and the unary cost function  $\mu$  applied to all vertices of  $\mathbb{G}$ . It is straightforward to check that the homomorphism that maps  $x \rightarrow 0$  and  $y \rightarrow 1$  has cost 0, as does the homomorphism that maps  $x \rightarrow 1$  and  $y \rightarrow 0$ . However the homomorphism that maps  $x \rightarrow 0$  and  $y \rightarrow 0$  has cost 2, and likewise for the homomorphism that maps  $x \rightarrow 1$  and  $y \rightarrow 1$ . If we consider these homomorphisms as the possible assignments of labels to the variables we have a VCSP instance  $\mathcal{I}$  with  $\Phi_{\mathcal{I}}(0, 0) = \Phi_{\mathcal{I}}(1, 1) > \Phi_{\mathcal{I}}(0, 1) = \Phi_{\mathcal{I}}(1, 0)$ , and thus we capture Max-Cut.

Note that following the construction in Definition 12 for any binary finite-valued cost function  $\phi_\Gamma$  we obtain a digraph  $\mathbb{D}_\Gamma$  which is quite similar to  $\mathbb{H}$ , except for an additional

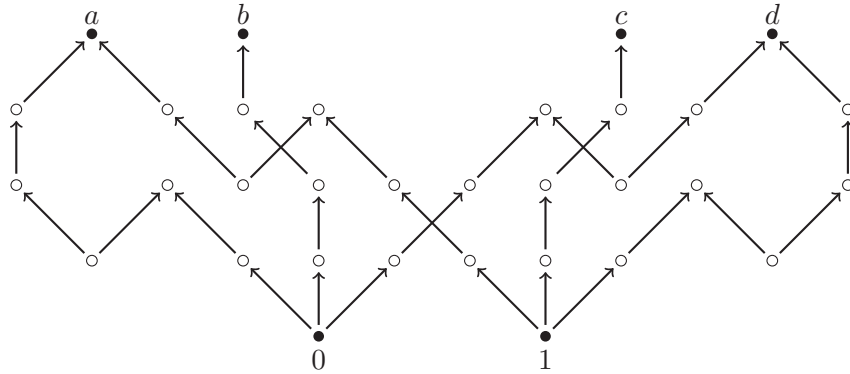


Figure 2: The target digraph  $\mathbb{H}$  of Example 6.

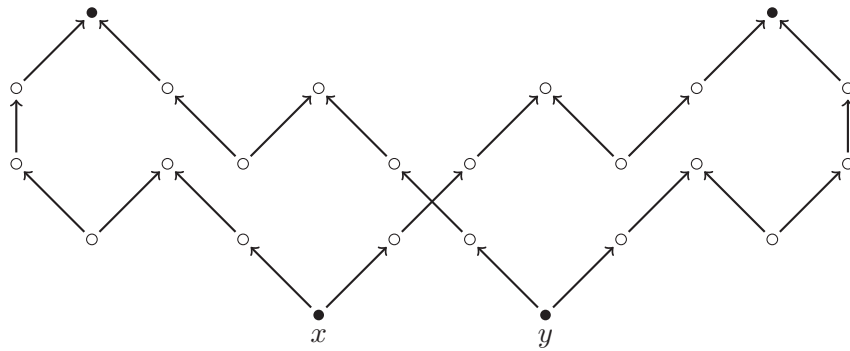


Figure 3: The source digraph  $\mathbb{G}$  of Example 6.

oriented path from 0 to  $c$  and another from 1 to  $b$ , each consisting of a single edge followed by two zigzags and another single edge. However no path in  $\mathbb{G}$  can possibly map onto these oriented paths, so they are omitted from  $\mathbb{H}$  to simplify the diagram.

## 6 Conclusion

Transforming a constraint satisfaction problem to a binary problem has a number of advantages and disadvantages which have been investigated by many authors [42, 20, 2, 48, 1, 14]. 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,  $\Gamma$ , of arbitrary arity to a *binary* language,  $\Gamma_d$ , of a very restricted kind, such that there is a bijection between the polymorphisms of  $\Gamma$  and the polymorphisms of  $\Gamma_d$ , and the corresponding polymorphisms satisfy exactly the same identities. Hence we have shown that the algebraic analysis of valued constraint languages can focus on a very restricted class of binary languages (at least in the case of finite 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.

Furthermore, we have adapted the recently obtained reduction for CSPs [14] to VCSPs and thus obtained a polynomial-time equivalence between VCSPs and MinCostHom problems. In order to study families of valued constraint languages with finitely many cost functions defined by fractional polymorphisms satisfying linear balanced identities, we now know that we need only study MinCostHom problems. This is important since, for example, to prove the algebraic dichotomy conjecture for core crisp languages we only need to study polymorphisms satisfying linear balanced identities [6].

We remark that the CSP reduction from [14] is shown to preserve a slightly larger class of identities than that of linear balanced identities, and works not only in polynomial time but actually in *logarithmic space*. We believe that our extension of this reduction can also be adapted to derive similar conclusions, but we leave this as an open problem. Our contribution is to show that the CSP reduction from [14] can be extended to the more general setting of the VCSP, and that the extended reduction preserves all linear balanced identities. Finally we remark that, even in the more general setting of the VCSP, using the dual construction as a stepping stone considerably simplifies the proof.

## References

- [1] Albert Atserias. On digraph coloring problems and treewidth duality. *Eur. J. Comb.*, 29(4):796–820, 2008.
- [2] Fahiem Bacchus, Xinguang Chen, Peter van Beek, and Toby Walsh. Binary vs. non-binary constraints. *Artificial Intelligence*, 140(1/2):1–37, 2002.
- [3] Libor Barto. Finitely related algebras in congruence distributive varieties have near unanimity terms. *Canadian Journal of Mathematics*, 65:3–21, 2013.
- [4] Libor Barto and Marcin Kozik. Constraint Satisfaction Problems Solvable by Local Consistency Methods. *Journal of the ACM*, 61(1), 2014. Article No. 3.
- [5] Libor Barto and Marcin Kozik. Robustly solvable constraint satisfaction problems. *SIAM Journal on Computing*, 45(4):1646–1669, 2016.
- [6] Libor Barto, Jakub Opršal, and Michael Pinsker. The wonderland of reflections. To appear in *Israel Journal of Mathematics*, [arXiv:1510.04521](https://arxiv.org/abs/1510.04521).
- [7] Andrew Blake, Pushmeet Kohli, and Carsten Rother. *Advances in Markov random fields for vision and image processing*. MIT Press, 2011.
- [8] A. Bulatov. A Dichotomy Theorem for Constraint Satisfaction Problems on a 3-Element Set. *Journal of the ACM*, 53(1):66–120, 2006.
- [9] A. Bulatov. Bounded relational width. Unpublished manuscript, 2009.
- [10] Andrei Bulatov and Víctor Dalmau. A simple algorithm for Mal'tsev constraints. *SIAM Journal on Computing*, 36(1):16–27, 2006.

- [11] Andrei Bulatov, Peter Jeavons, and Andrei Krokhin. Classifying the Complexity of Constraints using Finite Algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.
- [12] Andrei A. Bulatov. Complexity of conservative constraint satisfaction problems. *ACM Transactions on Computational Logic*, 12(4), 2011. Article 24.
- [13] Andrei A. Bulatov. Graphs of relational structures: Restricted types. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'16)*, pages 642–651. ACM, 2016.
- [14] Jakub Bulín, Dejan Delic, Marcel Jackson, and Todd Niven. A finer reduction of constraint problems to digraphs. *Logical Methods in Computer Science*, 11(4), 2015.
- [15] D. Cohen, M. Cooper, P. Jeavons, and A. Krokhin. A Maximal Tractable Class of Soft Constraints. *Journal of Artificial Intelligence Research*, 22:1–22, 2004.
- [16] David A. Cohen, Martin C. Cooper, Páidí Creed, Peter Jeavons, and Stanislav Živný. An algebraic theory of complexity for discrete optimisation. *SIAM Journal on Computing*, 42(5):915–1939, 2013.
- [17] David A. Cohen, Martin C. Cooper, Peter Jeavons, and Stanislav Živný. Binarisation via dualisation for valued constraints. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*, pages 3731–3737. AAAI Press, 2015.
- [18] David A. Cohen, Peter G. Jeavons, and Stanislav Živný. The expressive power of valued constraints: Hierarchies and collapses. *Theoretical Computer Science*, 409(1):137–153, 2008.
- [19] Rina Dechter and Judea Pearl. Tree Clustering for Constraint Networks. *Artificial Intelligence*, 38:353–366, 1989.
- [20] Tomás Feder and Moshe Y. Vardi. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.
- [21] Peter Fulla and Stanislav Živný. A Galois Connection for Valued Constraint Languages of Infinite Size. *ACM Transactions on Computation Theory*, 8(3), 2016. Article No. 9.
- [22] G. Gutin, P. Hell, A. Rafiey, and A. Yeo. A dichotomy for minimum cost graph homomorphisms. *European Journal of Combinatorics*, 29(4):900–911, 2008.
- [23] Pavol Hell and Jaroslav Nešetřil. On the Complexity of  $H$ -coloring. *Journal of Combinatorial Theory, Series B*, 48(1):92–110, 1990.
- [24] Pavol Hell and Arash Rafiey. The Dichotomy of Minimum Cost Homomorphism Problems for Digraphs. *SIAM Journal on Discrete Mathematics*, 26(4):1597–1608, 2012.
- [25] Anna Huber, Andrei Krokhin, and Robert Powell. Skew bisubmodularity and valued CSPs. *SIAM Journal on Computing*, 43(3):1064–1084, 2014.
- [26] Pawel M. Idziak, Petar Markovic, Ralph McKenzie, Matthew Valeriote, and Ross Willard. Tractability and learnability arising from algebras with few subpowers. *SIAM Journal on Computing*, 39(7):3023–3037, 2010.



- [27] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM*, 48(4):761–777, 2001.
- [28] Peter G. Jeavons, David A. Cohen, and Marc Gyssens. Closure Properties of Constraints. *Journal of the ACM*, 44(4):527–548, 1997.
- [29] Alexandr Kazda. Maltsev digraphs have a majority polymorphism. *Eur. J. Comb.*, 32(3):390–397, 2011.
- [30] Vladimir Kolmogorov, Andrei A. Krokhin, and Michal Rolínek. The complexity of general-valued CSPs. *SIAM Journal on Computing*, 46(3):1087–1110, 2017.
- [31] Vladimir Kolmogorov, Johan Thapper, and Stanislav Živný. The power of linear programming for general-valued CSPs. *SIAM Journal on Computing*, 44(1):1–36, 2015.
- [32] Vladimir Kolmogorov and Stanislav Živný. The complexity of conservative valued CSPs. *Journal of the ACM*, 60(2), 2013. Article No. 10.
- [33] Marcin Kozik and Joanna Ochremiak. Algebraic properties of valued constraint satisfaction problem. In *Proceedings of the 42nd International Colloquium on Automata, Languages and Programming (ICALP’15)*, volume 9134 of *Lecture Notes in Computer Science*, pages 846–858. Springer, 2015.
- [34] Andrei Krokhin and Stanislav Živný. The complexity of valued CSPs. In Andrei Krokhin and Stanislav Živný, editors, *Complexity and approximability of Constraint Satisfaction Problems*, volume 7 of *Dagstuhl Follow-Ups*, pages 233–266. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.
- [35] Benoit Larose and László Zádori. Bounded width problems and algebras. *Algebra Universalis*, 56(3-4):439–466, 2007.
- [36] J. Larrosa and R. Dechter. On the Dual Representation of Non-binary Semiring-based CSPs. In *Workshop on Soft Constraints – CP’00*, 2000.
- [37] M. Maróti and R. McKenzie. Existence theorems for weakly symmetric operations. *Algebra Universalis*, 59(3-4):463–489, 2008.
- [38] Ugo Montanari. Networks of Constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 1974.
- [39] Sebastian Nowozin, Peter V Gehler, Jeremy Jancsary, and Christoph H Lampert. *Advanced Structured Prediction*. MIT Press, 2014.
- [40] Robert Powell and Andrei A. Krokhin. A reduction from valued CSP to min cost homomorphism problem for digraphs. Technical report, 2015. [arXiv:1507.01776](https://arxiv.org/abs/1507.01776).
- [41] Robert D. Powell. *Complexity Classifications for the Valued Constraint Satisfaction Problem*. PhD thesis, University of Durham, 2015.
- [42] F. Rossi, V. Dahr, and C. Petrie. On the Equivalence of Constraint Satisfaction Problems. In *Proceedings of the 9th European Conference on Artificial Intelligence (ECAI’90)*, Stockholm, August 1990.

- [43] Thomas J. Schaefer. The Complexity of Satisfiability Problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC'78)*, pages 216–226. ACM, 1978.
- [44] Alexander Schrijver. A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.
- [45] Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*, volume 24 of *Algorithms and Combinatorics*. Springer, 2003.
- [46] H. D. Sherali and W. P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal of Discrete Mathematics*, 3(3):411–430, 1990.
- [47] Barbara M. Smith, Kostas Stergiou, and Toby Walsh. Using auxiliary variables and implied constraints to model non-binary problems. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI'00)*, pages 182–187, 2000.
- [48] Kostas Stergiou and Nikos Samaras. Binary encodings of non-binary constraint satisfaction problems: Algorithms and experimental results. *Journal of Artificial Intelligence Research (JAIR)*, 24:641–684, 2005.
- [49] R. Takhanov. A dichotomy theorem for the general minimum cost homomorphism problem. In *In Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS10)*, pages 657–668, 2010.
- [50] R. Takhanov. Extensions of the minimum cost homomorphism problem. In *Proceedings of the 16th Annual International Conference on Computing and Combinatorics (COCON'10)*, pages 328–337, 2010.
- [51] Johan Thapper and Stanislav Živný. The complexity of finite-valued CSPs. *Journal of the ACM*, 63(4), 2016. Article No. 37.
- [52] Johan Thapper and Stanislav Živný. The power of Sherali-Adams relaxations for general-valued CSPs. *SIAM Journal on Computing*, 46(4):1241–1279, 2017.
- [53] H. Uppman. The complexity of three-element min-sol and conservative min-cost-hom. In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP'13)*, pages 804–815, 2013.
- [54] H. Uppman. Computational complexity of the extended minimum cost homomorphism problem on three-element domains. In *Proceedings of the 31st International Symposium on Theoretical Aspects of Computer Science (STACS'14)*, pages 651–662, 2014.
- [55] Stanislav Živný, David A. Cohen, and Peter G. Jeavons. The Expressive Power of Binary Submodular Functions. *Discrete Applied Mathematics*, 157(15):3347–3358, 2009.