The Power of Linear Programming for Valued CSPs

Johan Thapper
Laboratoire d'Informatique (LIX)
École Polytechnique
91128 Palaiseau, France
thapper@lix.polytechnique.fr

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

***Abstract*—A class of *valued constraint satisfaction problems* (VCSPs) is characterised by a *valued constraint language*, a fixed set of cost functions on a finite domain. An instance of the problem is specified by a sum of cost functions from the language with the goal to minimise the sum. This framework includes and generalises well-studied constraint satisfaction problems (CSPs) and maximum constraint satisfaction problems (Max-CSPs).**

Our main result is a precise algebraic characterisation of valued constraint languages whose instances can be solved *exactly* by the *basic linear programming relaxation*. Using this result, we obtain tractability of several novel and previously widely-open classes of VCSPs, including problems over valued constraint languages that are: (1) submodular on *arbitrary lattices*; (2) bisubmodular (also known as k-submodular) on *arbitrary finite domains*; (3) weakly (and hence strongly) tree-submodular on *arbitrary trees*.

***Keywords*-valued constraint satisfaction; fractional polymorphisms; fractional homomorphisms; submodularity; bisubmodularity; linear programming**

I. Introduction

The constraint satisfaction problem (CSP) provides a common framework for many theoretical and practical problems in computer science. An instance can be vaguely described as a set of variables to be assigned values from the domains of the variables so that all constraints are satisfied [45]. The CSP is NP-complete in general and thus we are interested in restrictions which give rise to tractable classes of problems. Following Feder & Vardi [22], we restrict the constraint language; that is, all constraint relations in a given instance must belong to a fixed, finite set of relations on the domain. The most successful approach to classifying language-restricted CSPs is the so-called algebraic approach [7], [30], [31], which has led to several complexity classifications [1], [4], [6], [8] and algorithmic characterisations [2], [28] going beyond the seminal work of Schaefer [47].

Motivated by reasons both theoretical (optimisation problems are different from decision problems) and practical (many problems are over-constrained and hence have no solution, or under-constrained and hence have many solutions), we study valued constraint satisfaction problems (VCSPs) [5], [48]. A valued constraint language is a finite set of cost functions on the domain, and a VCSP instance is given by a weighted sum of cost functions from the language with the goal to *minimise* the sum. (CSPs correspond to the case when the range of all cost functions is $\{0, \infty\}$, and Max-CSPs correspond to the case when the range of all cost functions is $\{0, 1\}$.[1]) The VCSP framework is very robust and has also been studied under different names such as Min-Sum problems, Gibbs energy minimisation, Markov Random Fields, Conditional Random Fields and others in different contexts in computer science [16], [41], [50].

Given the generality of the VCSP, it is not surprising that only few complexity classifications are known. In particular, only Boolean (on a 2-element domain) languages [12], [17] and conservative (containing all $\{0, 1\}$-valued unary cost functions) languages [37] have been completely classified with respect to exact solvability. On the algorithmic side, most known tractable languages are somewhat related to submodular functions on distributive lattices [11], [12], [33], [37]. An alternative approach for solving VCSPs is using linear programming (LP) and semidefinite programming (SDP); these have been used mostly for approximation [3], [19], [40], [46].

Contribution

We study the power of the *basic linear programming relaxation* (BLP). Our main result (Theorem 3.1) is a precise characterisation of valued constraint languages for which BLP is a decision procedure. In more detail, we characterise valued constraint languages over which VCSP instances can be solved exactly by the BLP, i.e., when the BLP has integrality gap 1. The characterisation is algebraic in terms of *fractional polymorphisms* [10].

Our work is the first link between solving VCSPs exactly using LP and the algebraic machinery for VCSPs introduced by Cohen et al. in [9], [10], [13]. Part of the proof is inspired by the characterisation of width-1 CSPs [20], [22]. One of the main technical contributions is a construction of symmetric fractional polymorphisms of all arities (Theorem 3.5).

[1] With respect to exact solvability, Max-CSPs ("maximising the number of satisfied constraints") are polynomial-time equivalent to Min-CSPs ("minimising the number of unsatisfied constraints"). Therefore, with respect to exact solvability, Max-CSPs are polynomial-time equivalent to $\{0, 1\}$-valued VCSPs.

This result allows us to demonstrate that several valued constraint languages are covered by our characterisation and thus are tractable; that is, VCSP instances over these languages can be solved exactly using BLP. New tractable languages include: (1) submodular languages on *arbitrary lattices*; (2) bisubmodular (also known as k-submodular) languages on *arbitrary finite domains*; (3) weakly (and hence strongly) tree-submodular languages on *arbitrary trees*. The complexity of (subclasses of) these languages has been mentioned explicitly as open problems in [21], [27], [36], [38]. More generally, we show that any valued constraint language with a binary multimorphism in which at least one operation is a semi-lattice operation is tractable (cf. Section V). Our results cover *all* known tractable finite-valued constraint languages.

Related work

Apart from identifying tractable classes of CSPs and VCSPs with respect to exact solvability, the approximability of Max-CSPs has attracted a lot of attention [18], [32], [34]. Under the assumption of the *unique games conjecture* [35], Raghavendra showed that the basic SDP relaxation solves all tractable finite-valued VCSPs [46].[2] Very recently, Max-CSPs that are *robustly approximable* have been characterised as those having *bounded width* [3], [19], [40]. Specifically, Kun et al. studies the question of which Weighted Max-CSPs[3] can be robustly approximated using BLP [40]. Their result is related but incomparable to ours as it applies to robust approximability and not to exact solvability, except for the special case of width-1 CSPs. In particular, "solving" ("deciding") for us means finding an optimum solution to a VCSP instance, which is an optimisation problem, whereas "solving" in [40] means (ignoring their results on robust approximability, which do not apply here) the basic LP formulation of a CSP instance finds a solution if one exists.[4]

We remark that our tractability results apply to the minimisation problem of VCSP instances (i.e., the objective function is given by a sum of "local" cost functions) but not to objective functions given by an oracle. In particular, submodular functions given by an oracle can be minimised on distributive lattices [29], [49], diamonds [39], and several constructions on lattices preserving tractability have been identified [38], but it is widely open what happens on non-distributive lattices. Similarly, bisubmodular functions given by an oracle can be minimised in polynomial-time on domains of size 3 [25], but the complexity is open on domains of larger size [27]. It is known that strongly tree-submodular functions given by an oracle can be minimised in polynomial time on binary trees [36], but the complexity is open on general (non-binary) trees. Similarly, it is known that weakly tree-submodular functions given by an oracle can be minimised in polynomial time on chains and forks [36], but the complexity on (even binary) trees is open.

Extending the notion of (generalised) arc consistency for CSPs [24], [42] and several previously studied notions of arc consistencies for VCSPs [15], Cooper et al. introduced *optimal soft arc consistency* (OSAC) [14], which is a linear program relaxation of a given VCSP instance. Since OSAC is is a tighter relaxation than BLP, all tractable classes identified in this paper are solved by OSAC as well. Similarly, since the basic SDP relaxation from [46] is tighter than BLP, all tractable cases identified in this paper are solved by it as well.

II. Preliminaries

The set of non-negative rational numbers is denoted by $\mathbb{Q}_{\geq 0}$. A *signature* τ is a set of *function symbols* f, each with an associated positive *arity*, $ar(f)$. A *valued τ-structure* A (also known as a *valued constraint language*, or just a *language*) consists of a *domain* $D = D(A)$, together with a function $f^A : D^{ar(f)} \to \mathbb{Q}_{\geq 0}$, for each function symbol $f \in \tau$. (To be precise, these are *finite-valued* structures. In Section IV, we will extend $\mathbb{Q}_{\geq 0}$ with infinity.)

Let A be a valued τ-structure. An instance of VCSP(A) is given by a valued τ-structure I. A solution to I is a function $h : D(I) \to D(A)$, its measure given by

$$\sum_{f \in \tau, \bar{x} \in D(I)^{ar(f)}} f^I(\bar{x}) f^A(h(\bar{x})).$$

The goal is to find a solution of minimum measure. This measure will be denoted by $\mathsf{Opt}_A(I)$.[5]

For an m-tuple $\bar{t}$, we denote by $\{\bar{t}\}$ the set of elements in $\bar{t}$. Furthermore, we denote by $[\bar{t}]$ the multiset of elements in $\bar{t}$.

A. Fractional Homomorphisms

Let A and B be valued structures over the same signature τ. Let B^A denote the set of all functions from $D(A)$ to $D(B)$. A *fractional homomorphism* from A to B is a function $\omega : B^A \to \mathbb{Q}_{\geq 0}$, with $\sum_{g \in B^A} \omega(g) = 1$, such that for every function symbol $f \in \tau$ and tuple $\bar{a} \in D(A)^{ar(f)}$, it holds that

$$\sum_{g \in B^A} \omega(g) f^B(g(\bar{a})) \leq f^A(\bar{a}),$$

where the functions g are applied component-wise.

We write $A \to_f B$ to indicate the existence of a fractional homomorphism.

Proposition 2.1: Assume that $A \to_f B$. Then $\mathsf{Opt}_A(I) \geq \mathsf{Opt}_B(I)$, for every instance I.

[2] Max-CSPs (=$\{0,1\}$-valued VCSPs) and finite-valued VCSPs, respectively, are called CSPs and Generalised CSPs (GCSPs), respectively, in [46].

[3] In Weighted Max-CSPs, every constraint f is $\{0, c_f\}$-valued, where c_f is a positive constant. Weighted Max-CPSs are a special case of VCSPs.

[4] Note that CSPs are defined as $\{0,1\}$-valued in [40] and not as $\{0,\infty\}$-valued, as in this paper. This is needed for the LP formulation and the measure of approximability. After all, [40] deals with Max-CSPs.

[5] We note that the range of the functions in the valued structure A is $\mathbb{Q}_{\geq 0}$ for traditional reasons, but all results hold true with $\mathbb{Q}$ as well.

Proof: Let ω be a fractional homomorphism from A to B, let $X = D(I)$ and let $h : X \to A$ be an arbitrary solution. Then,

$$\sum_{f,\bar{x}} f^I(\bar{x}) f^A(h(\bar{x})) \geq \sum_{f,\bar{x}} f^I(\bar{x}) \sum_{g \in B^A} \omega(g) f^B(g(h(\bar{x}))) = \sum_{g \in B^A} \omega(g) \sum_{f,\bar{x}} f^I(\bar{x}) f^B(g(h(\bar{x}))),$$

where the sums are over $f \in \tau$ and $\bar{x} \in X^{ar(f)}$. Hence, there exists a $g \in B^A$ such that the measure of the solution $g \circ h$ to I as an instance of VCSP(B) is no greater than the measure of the solution h to I as an instance of VCSP(A). ■

B. Fractional Polymorphisms

Let A be a valued τ-structure, and let $D = D(A)$. An m-ary *operation* on D is a function $g : D^m \to D$. Let $\mathcal{O}_D^{(m)}$ denote the set of all m-ary operations on D. An m-ary *fractional operation* is a function $\omega : \mathcal{O}_D^{(m)} \to \mathbb{Q}_{\geq 0}$. Define $\|\omega\|_1 := \sum_g \omega(g)$. An m-ary fractional operation ω is called an m-ary *fractional polymorphism* [10] if $\|\omega\|_1 = 1$ and for every function symbol $f \in \tau$ and tuples $\bar{a}_1, \ldots, \bar{a}_m \in D^{ar(f)}$, it holds that

$$\sum_{g \in \mathcal{O}_D^{(m)}} \omega(g) f^A(g(\bar{a}_1, \ldots, \bar{a}_m)) \leq \frac{1}{m} \sum_{i=1}^{m} f^A(\bar{a}_i). \quad (1)$$

The set $\{g \mid \omega(g) > 0\}$ of operations is called the *support* of ω and is denoted by supp(ω). An operation g is *idempotent* if $g(x, \ldots, x) = x$. A fractional operation is idempotent if all operations in its support are idempotent. Let S_m be the symmetric group on $\{1, \ldots, m\}$. An m-ary operation g is *symmetric* if for every permutation $\pi \in S_m$, we have $g(x_1, \ldots, x_m) = g(x_{\pi(1)}, \ldots, x_{\pi(m)})$. A fractional operation is symmetric if all operations in its support are symmetric.

The *superposition*, $h[g_1, \ldots, g_n]$, of an n-ary operation h with n m-ary operations $g_1, \ldots, g_n$ is the m-ary operation defined by $h[g_1, \ldots, g_n](x_1, \ldots, x_m) = h(g_1(x_1, \ldots, x_m), \ldots, g_n(x_1, \ldots, x_m))$. A set of operations is called a *clone* if it contains all projections and is closed under superposition. The smallest clone that contains a set of operations $\mathcal{F}$ is called the clone *generated* by $\mathcal{F}$. We say that an operation f is *generated* by $\mathcal{F}$ if it is contained in the clone generated by $\mathcal{F}$.

Definition 1: The *superposition*, $\omega[g_1, \ldots, g_n]$, of an n-ary fractional operation ω with n m-ary operations $g_1, \ldots, g_n$ is the m-ary fractional operation ω', where

$$\omega'(h') = \sum_{h : h' = h[g_1, \ldots, g_n]} \omega(h).$$

In general ω' is not a fractional polymorphism, even when ω is, but $\|\omega'\|_1 = \|\omega\|_1$ and ω' satisfies the following inequality:

$$\begin{aligned} &\sum_{h' \in \mathcal{O}_D^{(m)}} \omega'(h') f^A(h'(\bar{a}_1, \ldots, \bar{a}_m)) \\ &= \sum_{h \in \mathcal{O}_D^{(n)}} \omega(h) f^A(h[g_1, \ldots, g_n](\bar{a}_1, \ldots, \bar{a}_m)) \\ &\leq \frac{1}{n} \sum_{i=1}^{n} f^A(g_i(\bar{a}_1, \ldots, \bar{a}_m)), \end{aligned} \quad (2)$$

for every $f \in \tau$ and $\bar{a}_1, \ldots \bar{a}_m \in D^{ar(f)}$.

C. The Multiset-Structure $P^m(A)$

Let A be a valued τ-structure, $D = D(A)$, and let $m \geq 1$. We define the *multiset-structure*[6] $P^m(A)$ as the valued structure with domain $\binom{D}{m}$, where $\binom{D}{m}$ denotes the multisets of elements from D of size m, and for every k-ary function symbol $f \in \tau$, and $\alpha_1, \ldots, \alpha_k \in \binom{D}{m}$,

$$f^{P^m(A)}(\alpha_1, \ldots, \alpha_k) = \frac{1}{m} \min_{\bar{t}_i \in D^m : [\bar{t}_i] = \alpha_i} \sum_{i=1}^{m} f^A(\bar{t}_1[i], \ldots, \bar{t}_k[i]).$$

The following lemma follows from the definitions.

Lemma 2.2: Let A be a valued structure and $m > 1$. Then $P^m(A) \to_f A$ if and only if A has an m-ary symmetric fractional polymorphism.

D. The Basic Linear Programming Relaxation

Let I and A be valued structures over a common finite signature τ. Let $X = D(I)$ and $D = D(A)$. The *basic LP relaxation* (BLP) (sometimes also called the *standard*, or *canonical LP relaxation*) has variables $\lambda_{f,\bar{x},\sigma}$ for $f \in \tau$, $\bar{x} \in X^{ar(f)}$, $\sigma : \{\bar{x}\} \to D$; and variables $\mu_x(a)$ for $x \in X, a \in D$.

$$\begin{aligned} \min \quad & \sum_{f,\bar{x}} \sum_{\sigma : \{\bar{x}\} \to D} f^I(\bar{x}) f^A(\sigma(\bar{x})) \lambda_{f,\bar{x},\sigma} \\ \text{s.t.} \quad & \sum_{\sigma : \sigma(x) = a} \lambda_{f,\bar{x},\sigma} = \mu_x(a) && \forall f \in \tau, \bar{x} \in X^{ar(f)}, \\ & && x \in \{\bar{x}\}, a \in D \\ & \sum_{a \in D} \mu_x(a) = 1 && \forall x \in X \\ & 0 \leq \lambda, \mu \leq 1 \end{aligned} \quad (3)$$

For any fixed A, BLP is polynomial in the size of a given VCSP(A) instance. Let IP be the program obtained from (3) together with the constraints that all variables take values in the range $\{0, 1\}$ rather than $[0, 1]$. This is an integer programming formulation of the original VCSP instance. The interpretation of the variables in IP is as follows: $\mu_x(a) = 1$ iff variable x is assigned value a; $\lambda_{f,\bar{x},\sigma} = 1$ iff constraint f on scope $\bar{x}$ is assigned tuple $\sigma(\bar{x})$. LP (3) is now a relaxation of IP and the question of whether (3)

[6] A similar structure for $\{0, \infty\}$-valued languages was introduced in [40].

solves a given VCSP instance I is the question of whether IP has integrality gap 1.

III. Characterisation

Definition 2: Let BLP(I, A) denote the optimum of (3). We say that BLP *solves* VCSP(A) if BLP$(I, A) = \mathsf{Opt}_A(I)$ for every instance I of VCSP(A).

Theorem 3.1 (Main): Let A be a valued structure over a finite signature. TFAE:

(i) BLP solves VCSP(A).
(ii) For every $m > 1$, $P^m(A) \to_f A$.
(iii) For every $m > 1$, A has an m-ary symmetric fractional polymorphism.
(iv) For every $n > 1$, A has a fractional polymorphism ω_n such that supp(ω_n) generates an n-ary symmetric operation.

When A has symmetric fractional polymorphisms of all arities, then solving the BLP provides the optimum value of the VCSP instance. To find an optimal assignment, we apply self-reduction: The idea is to successively try each possible value for a variable and solve the new LP. Once the new optimum matches the original one, we proceed with the next variable. To make sure that optimal solutions in the new instance actually obtain the optimum of the new LP, we need the symmetric fractional polymorphisms of A to be idempotent. This can be guaranteed if the support of any unary fractional polymorphism of A contains only injective operations. The latter can be showed to hold in an appropriate substructure of A. Details will be provided in the full version of this paper.

The rest of this section is devoted to proving Theorem 3.1. We start with proving $(ii) \Rightarrow (i)$.

Theorem 3.2: Assume that $P^m(A) \to_f A$ for every $m > 1$. Then BLP solves VCSP(A).

Proof: Let λ^*, μ^* be an optimal solution to (3). Let M be a positive integer such that $M \cdot \lambda^*$ and $M \cdot \mu^*$ are both integral.

Let $\nu : X \to \binom{D}{M}$ be defined by mapping x to the multiset in which the elements are distributed according to μ^*_x, i.e., the number of occurrences of a in $\nu(x)$ is equal to $M \cdot \mu^*_x(a)$ for each $a \in D$.

Let f be a k-ary function symbol in τ, and consider the sum of all terms of the objective function in which f occurs:

$$\sum_{\bar{x}} f^I(\bar{x})\Big(\sum_{\sigma:\{\bar{x}\}\to D} \lambda^*_{f,\bar{x},\sigma} f^A(\sigma(\bar{x}))\Big).$$

Now, write

$$M \cdot \sum_{\sigma:\{\bar{x}\}\to D} \lambda^*_{f,\bar{x},\sigma} f^A(\sigma(\bar{x})) = f^A(\bar{a}_1) + \cdots + f^A(\bar{a}_M),$$

where the $\bar{a}_i \in D^k$ are such that a $\lambda^*_{f,\bar{x},\sigma}$-fraction are equal to $\sigma(\bar{x})$ and let $\bar{a}'_i = (\bar{a}_1[i], \ldots, \bar{a}_M[i])$ for $i = 1, \ldots, k$.

$$\begin{aligned}
\sum_{\sigma:\{\bar{x}\}\to D} \lambda^*_{f,\bar{x},\sigma} f^A(\sigma(\bar{x})) &= \frac{1}{M}\sum_{i=1}^{M} f^A(\bar{a}_i) \\
&= \frac{1}{M}\sum_{i=1}^{M} f^A(\bar{a}'_1[i], \ldots, \bar{a}'_k[i]) \\
&\geq \frac{1}{M} \min_{\bar{t}_i \in D^M : [\bar{t}_i] = [\bar{a}'_i]} \sum_{i=1}^{M} f^A(\bar{t}_1[i], \ldots, \bar{t}_k[i]) \\
&= f^{P^M(A)}(\nu(\bar{x})),
\end{aligned}$$

where the last equality follows as the number of a's in $\bar{a}'_i$ is $M \cdot \sum_{\sigma:\sigma(\bar{x}[i])=a} \lambda^*_{f,\bar{x},\sigma} = M \cdot \mu^*_{\bar{x}[i]}(a)$.

We now have

$$\begin{aligned}
BLP(I, A) &= \sum_{f,\bar{x}} \sum_{\sigma:\{\bar{x}\}\to D} f^I(\bar{x}) f^A(\sigma(\bar{x})) \lambda^*_{f,\bar{x},\sigma} \\
&= \sum_{f\in\tau,\bar{x}} f^I(\bar{x})\Big(\sum_{\sigma:\{\bar{x}\}\to D} \lambda^*_{f,\bar{x},\sigma} f^A(\sigma(\bar{x}))\Big) \\
&\geq \sum_{f\in\tau,\bar{x}} f^I(\bar{x}) f^{P^M(A)}(\nu(\bar{x})) \\
&\geq \mathsf{Opt}_{P^M(A)}(I).
\end{aligned}$$

It follows that $\mathsf{Opt}_A(I) \geq BLP(I, A) \geq \mathsf{Opt}_{P^M(A)}(I)$. Since $P^M(A) \to_f A$, the result then follows from Proposition 2.1. ∎

To prove $(i) \Rightarrow (ii)$, we need the following variant of Farkas' Lemma, due to Gale [26] (cf. Mangasarian [43]).

Lemma 3.3: Let $A \in \mathbb{R}^{m\times n}$ and $\bar{b} \in \mathbb{R}^m$. Then exactly one of the two holds:

- $A\bar{x} \leq \bar{b}$ for some $\bar{x} \in \mathbb{R}^n$; or
- $A^T\bar{y} = 0$, $\bar{b}^T\bar{y} = -1$ for some $\bar{y} \in \mathbb{R}_{\geq 0}$.

Theorem 3.4: Let A be a valued structure and assume that BLP solves VCSP(A). Then $P^m(A) \to_f A$ for every $m > 1$.

Proof: Let τ be the signature of A and let $D = D(A)$. We prove the contrapositive. Assume that there is an integer $m > 1$ such that $P^m(A)$ does not have a fractional homomorphism to A. Let Ω denote the set of functions from $\binom{D}{m}$ to D. We are assuming that the following system of inequalities does not have a solution $\omega : \Omega \to \mathbb{Q}_{\geq 0}$.

$$\begin{aligned}
\sum_{g\in\Omega} \omega(g) f^A(g(\bar{\alpha})) \leq f^{P^m(A)}(\bar{\alpha}) \quad \forall f \in \tau, \bar{\alpha} \in \tbinom{D}{m}^{ar(f)} \\
\sum_{g\in\Omega} \omega(g) = 1 \\
\omega(g) \geq 0 \quad \forall g \in \Omega.
\end{aligned}$$

In order to apply Lemma 3.3, we rewrite the equality $\sum_g \omega(g) = 1$ into two inequalities $\sum_g \omega(g) \leq 1$ and $-\sum_g \omega(g) \leq -1$. The last set of inequalities are rewritten to the form $-\omega(g) \leq 0$ for each $g \in \Omega$. We have one variable for each inequality, i.e., $y(f, \bar{\alpha})$ for $f \in \tau$, and

$\bar{\alpha} \in \left(\!\binom{D}{m}\!\right)^{ar(f)}$. Additionally, we have two variables z_+, z_- for the two inequalities involving the constant 1 and one variable $w(g)$ for each $g \in \Omega$.

$$\sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^A(g(\bar{\alpha})) + z_+ - z_- - w(g) = 0$$

$$\sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^{P^m(A)}(\bar{\alpha}) + z_+ - z_- = -1$$

$$y, z_+, z_-, w \geq 0$$

We can isolate $z_+ + z_-$ in the last equality,

$$z_+ + z_- = -1 - \sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^{P^m(A)}(\bar{\alpha}),$$

which substituted into the first set of equalities implies that there is a solution $y(f,\bar{\alpha}), w(g)$ such that, for each $g \in \Omega$,

$$\sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^A(g(\bar{\alpha})) = w(g) + 1 + \sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^{P^m(A)}(\bar{\alpha}).$$

We therefore find that there is a solution to the following system:

$$\sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^A(g(\bar{\alpha})) > \sum_{f,\bar{\alpha}} y(f,\bar{\alpha}) f^{P^m(A)}(\bar{\alpha}) \quad \forall g \in \Omega$$

$$y(f,\bar{\alpha}) \geq 0 \quad \forall f, \bar{\alpha}. \tag{4}$$

Let I be the following instance on variables $\left(\!\binom{D}{m}\!\right)$. For each k-ary function symbol $f \in \tau$, and $\bar{\alpha} \in \left(\!\binom{D}{m}\!\right)^{ar(f)}$, define

$$f^I(\bar{\alpha}) = y(f,\bar{\alpha}).$$

We now give a solution λ, μ to the basic LP (3) with an objective value equal to the right-hand side of (4). Each variable $\mu_\alpha(a)$ is assigned the value of the multiplicity of a in α divided by m. Given $f, \bar{\alpha}$, let $\bar{t}_1, \ldots, \bar{t}_k \in D^m$ be such that $f^{P^m(A)}(\bar{\alpha}) = \frac{1}{m}\sum_{i=1}^m f^A(\bar{t}_1[i], \ldots, \bar{t}_k[i])$, and assign values to the λ-variables as follows:

$$\lambda_{f,\bar{\alpha},\sigma} = \frac{1}{m}|\{i \mid \sigma(\bar{\alpha}[j]) = \bar{t}_j[i] \text{ for all } j\}|$$

Note that $\sum_{\sigma:\sigma(\bar{\alpha}[j])=a} \lambda_{f,\bar{\alpha},\sigma} = \mu_{\bar{\alpha}[j]}(a)$ for all $1 \leq j \leq k$ and $a \in D$. Furthermore, λ is defined so that $f^{P^m(A)}(\bar{\alpha}) = \sum_{\sigma:\{\bar{\alpha}\}\to D} f^A(\sigma(\bar{\alpha}))\lambda_{f,\bar{\alpha},\sigma}$. Hence, the variables λ, μ satisfy the basic LP (3), and we have

$$\begin{aligned} BLP(I,A) &\leq \sum_{f,\bar{\alpha}} f^I(\bar{\alpha}) \sum_{\sigma:\{\bar{\alpha}\}\to D} f^A(\sigma(\bar{\alpha}))\lambda_{f,\bar{\alpha},\sigma} \\ &= \sum_{f,\bar{\alpha}} f^I(\bar{\alpha}) f^{P^m(A)}(\bar{\alpha}), \end{aligned} \tag{5}$$

where the sums are over $f \in \tau$ and $\bar{\alpha} \in \left(\!\binom{D}{m}\!\right)^{ar(f)}$.

It now follows from (4) and (5) that the measure of any solution $g : \left(\!\binom{D}{m}\!\right) \to D$ to I is strictly greater than BLP(I,A). Consequently, BLP does not solve VCSP(A). ■

Lemma 2.2 proves $(ii) \Leftrightarrow (iii)$.

Since $(iii) \Rightarrow (iv)$ follows trivially, it remains to show that $(iv) \Rightarrow (iii)$.

Theorem 3.5: Let A be a valued structure and assume that for every $n > 1$, A has a fractional polymorphism ω_n that generates an n-ary symmetric operation. Then, for every $m > 1$, A has an m-ary symmetric fractional polymorphism.

Proof: For an m-ary operation g, let $\tilde{g}$ denote the equivalence class of g under the relation:

$$g \sim g' \Leftrightarrow \exists \pi \in S_m : g(x_1, \ldots, x_m) = g'(x_{\pi(1)}, \ldots, x_{\pi(m)}).$$

Note that we have $|\tilde{g}| = 1$ if and only if g is symmetric.

We say that a fractional operation ω is *weight-symmetric* if $\omega(g) = \omega(g')$ whenever $g \sim g'$.

We construct an m-ary symmetric fractional polymorphism by building a rooted tree in a number of stages. At each stage of the construction, every node u of the tree contains an m-ary weight-symmetric fractional operation with support on a single equivalence class of $\sim$. For a node u, we will also denote this fractional operation by u. Since u is weight-symmetric, it follows that $u(g) = u(g')$ for all $g, g' \in \text{supp}(u)$. This common weight for the operations in the support of u will be denoted by $w(u)$. A node u with $|\text{supp}(u)| = 1$ will be called *final*.

The following invariants are maintained throughout the construction.

(a) Every non-leaf node has at least one final child.
(b) For every node u, we have $w(u) > 0$.
(c) For every non-leaf node v,

$$\sum_{u \text{ is a child of } v} \|u\|_1 = \|v\|_1.$$

(d) For every non-leaf node v, every $f \in \tau$, and all tuples $\bar{a}_1, \ldots, \bar{a}_m \in D^{ar(f)}$,

$$\begin{aligned} &\sum_g v(g) f^A(g(\bar{a}_1, \ldots, \bar{a}_m)) \\ &\geq \sum_{u \text{ is a child of } v} \sum_g u(g) f^A(g(\bar{a}_1, \ldots, \bar{a}_m)). \end{aligned}$$

We say that a leaf u *is covered (by v)*, and that v is *a covering node (of u)* if $\text{supp}(u) = \text{supp}(v)$ for some (proper) descendant u of v. We say that v is a *minimal* covering node if no descendant of v is a covering node.

At the beginning of the construction, the tree consists of a single root r with $\text{supp}(r)$ being the set of m-ary projections and $w(r) = \frac{1}{m}$. We then apply the following two steps:

- *Expansion:* A leaf u that is not final and not covered is chosen to be expanded. This amounts to adding a finite non-empty set of children to u while maintaining the invariants. The expansion step is repeated until no longer applicable.
- *Pruning:* A leaf that is not final and covered is removed together with a number of internal nodes while maintaining the invariants. The pruning step is repeated until no longer applicable.

Since there is a finite number of m-ary operations, and hence a finite number of equivalence classes of $\sim$, it follows that, eventually, every leaf in the tree that is not final must be covered. Hence, the expansion step is only applicable a finite number of times.

Each round of pruning shrinks the tree by at least one node, but no final leaf is ever removed. Therefore we eventually obtain a tree containing only final leaves, at which time the pruning step is no longer applicable. Let $\mathcal{L}$ be the set of leaves in the final tree. By repeated application of invariant (d), starting from the root, $\sum_{u\in\mathcal{L}} u$ is then an m-ary symmetric fractional polymorphism.

Expansion

We expand a leaf u with $|\text{supp}(u)| = n$ as follows: Let ω be a k-ary fractional polymorphism of A such that $\text{supp}(\omega)$ generates an n-ary symmetric operation t.

We will define a sequence of m-ary weight-symmetric fractional operations ν_i, each with $\|\nu_i\|_1 = \|u\|_1$. Let $\nu_0 = u$. Assume that ν_{i-1} has been defined for some $i \geq 1$. Let $l_{i-1} = \min\{\nu_{i-1}(g) \mid g \in \text{supp}(\nu_{i-1})\}$ be the minimum weight of an operation in the support of ν_{i-1}. The fractional operation ν_i is obtained by subtracting from ν_{i-1} an equal amount of weight from each operation in $\text{supp}(\nu_{i-1})$ and adding this weight as superpositions of ω by all possible choices of operations in ν_{i-1}. The amount subtracted from each operation is $\frac{1}{2}l_{i-1}$ so that every operation in $\text{supp}(\nu_{i-1})$ is also in $\text{supp}(\nu_i)$.

Formally ν_i is defined as follows:

$$\nu_i = \nu_{i-1} - \frac{l_{i-1}}{2}\chi_{i-1} + \frac{l_{i-1}}{2}\eta_{i-1},$$

where χ_{i-1} is the (normalised) indicator function of the set $\text{supp}(\nu_{i-1})$ and, with $K = |\text{supp}(\nu_{i-1})|^k$,

$$\eta_{i-1} = \frac{1}{K}\sum_{g_1,\ldots,g_k\in\text{supp}(\nu_{i-1})} \omega[g_1,\ldots,g_k].$$

By definition $\|\nu_i\|_1 = \|\nu_{i-1}\|_1 = \|u\|_1$. To verify that ν_i is weight-symmetric, it suffices to verify that η_{i-1} is weight-symmetric. Let g by any m-ary operation of the form $g = h[g_1,\ldots,g_k]$ and let $g \sim g'$. Let $\pi \in S_m$ be such that $g(x_1,\ldots,x_m) = g'(x_{\pi(1)},\ldots,x_{\pi(m)})$ and define $g'_j(x_1,\ldots,x_m) = g_j(x_{\pi(1)},\ldots,x_{\pi(m)})$ for $1 \leq j \leq k$. Since ν_{i-1} is weight-symmetric, it follows that $g_i \in \text{supp}(\nu_{i-1})$ if and only if $g'_i \in \text{supp}(\nu_{i-1})$. Therefore the terms $\omega(h)h[g_1,\ldots,g_k]$ in η_{i-1} such that $g = h[g_1,\ldots,g_k]$ are in bijection with the terms $\omega(h)h[g'_1,\ldots,g'_k]$ such that $g' = h[g'_1,\ldots,g'_k]$. So the fractional operation η_{i-1} assigns the same weight to g and g'.

Let e be an expression for t consisting of superpositions of projections and operations from $\text{supp}(\omega)$. We recursively define the *nested depth*, $d = d(e)$, of e as follows: $d(p) = 0$ for every projection p; and $d(h[g_1,\ldots,g_k]) = 1 + \max_{1\leq i\leq k} d(g_i)$.

Let $\text{supp}(u) = \{g_1,\ldots,g_n\}$. Using $\text{supp}(\nu_0) = \text{supp}(u)$ and the fact that $\text{supp}(\nu_i)$ contains all superpositions of operations in $\text{supp}(\nu_{i-1})$, it follows that $t[g_1,\ldots,g_n] \in \text{supp}(\nu_d)$. Now, we add a child v to u for every equivalence class in the set $\{\tilde{g} \mid g \in \text{supp}(\nu_d)\}$. For an added child v with $\text{supp}(v) = \tilde{g}$, we let $w(v) = \nu_d(g)$.

Invariant (a) holds as $t[g_1,\ldots,g_n] \in \text{supp}(\nu_d)$ is symmetric: for all $\pi \in S_m$ there is a $\pi' \in S_n$ such that

$$\begin{aligned} &t[g_1,\ldots,g_n](x_{\pi(1)},\ldots,x_{\pi(m)}) \\ &= t[g_{\pi'(1)},\ldots,g_{\pi'(n)}](x_1,\ldots,x_m) \\ &= t[g_1,\ldots,g_n](x_1,\ldots,x_m). \end{aligned}$$

Invariants (b) and (c) hold by construction. We now claim that for each $i \geq 1$, we have $\sum_g \nu_i(g)f^A(g(\bar{a}_1,\ldots,\bar{a}_m)) \leq \sum_g \nu_{i-1}(g)f^A(g(\bar{a}_1,\ldots,\bar{a}_m))$, for all $f \in \tau$ and $\bar{a}_1,\ldots\bar{a}_m \in D^{ar(f)}$. From this it follows that invariant (d) also holds after expanding u.

To see why the claim holds, compare the last two terms in the definition of ν_i using inequality (2):

$$\begin{aligned} &\sum_g \chi_{i-1}(g)f^A(g(\bar{a}_1,\ldots,\bar{a}_m)) \\ &= \frac{1}{K}\sum_{g_1,\ldots,g_k\in\text{supp}(\nu_{i-1})} \frac{1}{k}\sum_{i=1}^{k} f^A(g_i(\bar{a}_1,\ldots,\bar{a}_m)) \\ &\geq \sum_h \eta_{i-1}(h)f^A(h(\bar{a}_1,\ldots,\bar{a}_m)). \end{aligned}$$

Pruning

The pruning step maintains an additional invariant, namely that every leaf that is not final is covered. Pruning is accomplished as follows. Pick a minimal covering node v. Let $\nu = \nu_v + \nu_\perp$ be the fractional operation induced by the leaves in the subtree rooted at v, where ν_v is the part of ν with the same support as v and $\nu_\perp$ is the part of ν with support disjoint from v. Inductively, by invariant (d), we have

$$\begin{aligned} &\sum_g v(g)f^A(g(\bar{a}_1,\ldots,\bar{a}_m)) \\ &\geq \sum_g (\nu_v(g) + \nu_\perp(g))f^A(g(\bar{a}_1,\ldots,\bar{a}_m)), \end{aligned}$$

for all $f \in \tau$ and $\bar{a}_1,\ldots\bar{a}_m \in D^{ar(f)}$. Since v and ν_v are weight-symmetric with identical support, it follows that $v - \nu_v$ is also weight-symmetric, hence $v - \nu_v \equiv \kappa \cdot v$ for $\kappa = \|v - \nu_v\|_1/\|v\|_1$. By invariant (a) the node v is guaranteed to have at least one final child (leaf). Hence, by invariant (b), $\nu_\perp$ is not identically 0. By induction on (c), it follows that $\|v\|_1 > \|\nu_v\|_1$, so $\kappa > 0$. We therefore have

$$\sum_g v(g)f^A(g(\bar{a}_1,\ldots,\bar{a}_m)) \geq \sum_g \frac{1}{\kappa}\nu_\perp(g)f^A(g(\bar{a}_1,\ldots,\bar{a}_m)).$$

Remove all nodes below v and add a new child u to v for every equivalence class in the set $\{\tilde{g} \mid g \in \text{supp}(\nu_\perp)\}$. For an added child u with $\text{supp}(u) = \tilde{g}$, we let $w(u) = \frac{1}{\kappa}\nu_\perp(g)$.

Invariant (a) holds since any symmetric operation in the support of ν must lie in the support of $\nu_\perp$. Clearly $w(u) > 0$ for each new node u, so invariant (b) holds. Furthermore,

$$\sum_{u \text{ is a child of } v} \|u\|_1 = \frac{1}{\kappa}\|\nu_\perp\|_1 = \frac{1}{\kappa}\|\nu - \nu_v\|_1 = \|v\|_1,$$

so invariant (c) holds. Invariant (d) holds by construction.

Only nodes in the subtree rooted at v have been removed and every child added to v has the same support as a previous leaf of this subtree. Every such leaf u that was not final was covered by a node above v in the tree. Hence, all leaves in the tree that are not final are still covered. ∎

IV. General-valued Structures

The valued structures we have dealt with so far were in fact *finite-valued*; i.e., for each function symbol $f \in \tau$, the range of f^A was $\mathbb{Q}_{\geq 0}$. We now discuss how our result can be extended to the *general-valued* case, in which for each function symbol $f \in \tau$, the range of f^A is $\overline{\mathbb{Q}}_{\geq 0} = \mathbb{Q}_{\geq 0} \cup \{\infty\}$. (We define $c + \infty = \infty + c = \infty$ for all $c \in \overline{\mathbb{Q}}_{\geq 0}$, and $0\infty = \infty 0 = 0$.)

Inspired by the OSAC algorithm [14], the algorithm for general-valued structures, denoted by BLP_g, works in two stages. Firstly, the instance is made arc consistent using a standard arc consistency algorithm [24] (more details below).[7] Secondly, BLP (i.e., linear program (3) from Section II-D) is solved.

Definition 3: Let A be a general-valued τ-structure, and let $D = D(A)$. An m-ary operation $g : D^m \to D$ is a *polymorphism* of A if for every function symbol $f \in \tau$ and tuples $\bar{a}_1, \dots, \bar{a}_m \in D^{ar(f)}$, it holds that $\mathsf{Feas}(f^A(g(\bar{a}_1, \dots, \bar{a}_m))) \leq \sum_{i=1}^m \mathsf{Feas}(f^A(\bar{a}_i))$, where $\mathsf{Feas}(\infty) = \infty$ and $\mathsf{Feas}(c) = 0$ for any $c \in \mathbb{Q}_{\geq 0}$.

From Definition 3, if ω is a fractional polymorphism of a general-valued structure A, then every $g \in \text{supp}(\omega)$ is a polymorphism of A. In fact, any operation g that is generated by $\text{supp}(\omega)$ is a polymorphism of A. (For finite-valued structures, trivially, every operation g is a polymorphism.)

Arc consistency (also known as $(1,k)$-consistency) [24] is a decision procedure for precisely those $\{0,\infty\}$-valued structures A that are closed under a *set function* $g : 2^{D(A)} \setminus \{\emptyset\} \to D(A)$ [20], [22]. This condition has recently been shown to be equivalent to the requirement that A should have symmetric polymorphisms of all arities [40].

The idea behind arc consistency is to prune the domains of the variables so that domain values without any support are removed. In particular, using the same notation as in Section II-D, we initialise $u(x) = D$ for every $x \in X$.[8] The arc consistency algorithm consists in repeating the following step until convergence: if there is $x \in X$ and $a \in u(x)$ and $f \in \tau$ and $\bar{x} \in X^{ar(f)}$ with $x \in \{\bar{x}\}$ such that there is no $h(\bar{x})$ with $h(x) = a$ satisfying $f^I(\bar{x})f^A(h(\bar{x})) < \infty$ and $h(x_i) \in u(x_i)$ for $x_i \in \{\bar{x}\}$, then remove a from $u(x)$.

After the arc consistency step, BLP_g is identical to BLP, but only remaining domain values are used for respective variables. In particular, variables $u_x(a)$ are defined only for $a \in u(x)$, variables $\lambda_{f,\bar{x},\sigma}$ are defined only for σ satisfying $\sigma(x_i) \in u(x_i)$ for every $x_i \in \{\bar{x}\}$, and similarly for the sums in (3) ranging over σ and a.

Our main theorem (Theorem 3.1) also holds for general-valued structures:

Theorem 4.1: Let A be a general-valued structure over a finite signature. TFAE:

(i) BLP_g solves $\text{VCSP}(A)$.
(ii) For every $m > 1$, $P^m(A) \to_f A$.
(iii) For every $m > 1$, A has an m-ary symmetric fractional polymorphism.
(iv) For every $n > 1$, A has a fractional polymorphism ω_n such that $\text{supp}(\omega_n)$ generates an n-ary symmetric operation.

Proof: Note that (ii) implies that A has symmetric polymorphisms of all arities. This follows from Lemma 2.2, which holds for general-valued structures, and the above-mentioned fact that any $g \in \text{supp}(\omega)$, where ω is a fractional polymorphism of A, is a polymorphism of A. The same argument guarantees symmetric polymorphisms of all arities in (iii) and (iv); in (iv), we use that fact than any operation generated by $\text{supp}(\omega_n)$ is a polymorphism of A.

Theorem 3.2 proves $(ii) \Rightarrow (i)$ since the assumption of having symmetric polymorphisms of all arities guarantees a feasible solution to (3). From the discussion above on arc consistency, if BLP_g solves $\text{VCSP}(A)$, then A has symmetric polymorphisms of all arities since arc consistency decides the existence of a finite-valued solution. Furthermore, the proof of Theorem 3.4 shows that having symmetric polymorphisms of all arities but not having a fractional homomorphism $P^m(A) \to_f A$ implies that BLP_g does not solve $\text{VCSP}(A)$. This gives $(i) \Rightarrow (ii)$. $(ii) \Leftrightarrow (iii)$ and $(iii) \Leftrightarrow (iv)$ are proved the same way as in Theorem 3.1, by Lemma 2.2 and Theorem 3.5, respectively. ∎

Remark 1: BLP_g for general-valued structures uses the arc consistency algorithm [24] and BLP. Since Kun et al. [40] have shown that BLP solves CSPs (i.e., $\{0,\infty\}$-valued VCSPs), if represented by $\{0,1\}$-valued structures, of width 1 – thus providing an alternative to the standard arc consistency algorithm [24] – a different approach is to combine Theorem 3.1 from this paper with [40] and solve general-valued structures using only BLP with an amended objective function which takes care of infinite costs using a large (but polynomial), instance-dependent constant.

[7] The algorithm from [24] is sometimes called the generalised arc consistency algorithm to emphasise the fact that it works for CSPs of arbitrary arities, not only for binary CSPs [42].

[8] Note that $u(x)$ is the effective domain of variable x.

V. Tractable Valued Constraint Languages

As before, we denote $D = D(A)$. A special case of fractional polymorphisms are *multimorphisms* [12]. A binary multimorphism $\langle g_1, g_2\rangle$ of a valued structure A is a binary fractional polymorphism ω of A such that $\omega(g_1) = \omega(g_2) = 1/2$, where $g_1, g_2 : D^2 \to D$. For a binary multimorphism $\langle g_1, g_2\rangle$, the fractional polymorphism inequality (1) simplifies to, for every function symbol $f \in \tau$ and tuples $\bar{a}_1, \bar{a}_2 \in D^{ar(f)}$,

$$f^A(g_1(\bar{a}_1, \bar{a}_2)) + f^A(g_2(\bar{a}_1, \bar{a}_2)) \leq f^A(\bar{a}_1) + f^A(\bar{a}_2).$$

A *semi-lattice operation* is an associative, commutative, and idempotent operation. Since any semi-lattice operation generates symmetric operations of all arities, we get:

Corollary 5.1 (of Theorem 4.1): Let A be a valued structure with a binary multimorphism $\langle g_1, g_2\rangle$ where either g_1 or g_2 is a semi-lattice operation. Then A is tractable.

We now give examples of valued structures (i.e., valued constraint languages) defined by such binary multimorphisms.

Example 1: Let $(D; \wedge, \vee)$ be an *arbitrary* lattice on D. Assume that a valued structure A has the multimorphism $\langle \wedge, \vee\rangle$. Then VCSP($A$) is tractable. The tractability of A was previously known only for distributive lattices [29], [49] and diamonds [39], see also [38].

Example 2: A pair of operations $\langle g_1, g_2\rangle$ is called a symmetric tournament pair (STP) if both g_1 and g_2 are commutative, conservative ($g_1(x, y) \in \{x, y\}$ and $g_2(x, y) \in \{x, y\}$ for all $x, y \in D$), and $g_1(x, y) \neq g_2(x, y)$ for all $x, y \in D$. Let A be a finite-valued structure with an STP multimorphism $\langle g_1, g_2\rangle$. It is known that if a finite-valued structure admits an STP multimorphism, it also admits a submodularity multimorphism. This result is implicitly contained in [11].[9] Therefore, BLP solves any instance from VCSP(A).

Example 3: Assume that a valued structure A is bisubmodular [25]. This means that $D = \{0, 1, 2\}$ and A has a multimorphism $\langle \min_0, \max_0\rangle$ [12], where $\min_0(x, x) = x$ for all $x \in D$ and $\min_0(x, y) = 0$ for all $x, y \in D, x \neq y$; $\max_0(x, y) = 0$ if $0 \neq x \neq y \neq 0$ and $\max_0(x, y) = \max(x, y)$ otherwise, where $\max$ returns the larger of its two arguments with respect to the normal order of integers. Since $\min_0$ is a semi-lattice operation, A is tractable. The tractability of (finite-valued) A was previously known only using a general algorithm for bisubmodular functions given by an oracle [25], [44].

Example 4: Assume that a valued structure A is *weakly tree-submodular* on an *arbitrary* tree [36]. The meet (which is defined as the highest common ancestor) is again a semi-lattice operation. The same holds for *strongly tree-submodular* structures since strong tree-submodularity implies weak tree-submodularity [36]. The tractability of weakly tree-submodular valued structures was previously known only for chains and forks [36]. The tractability of strongly tree-submodular valued structures was previously known only for binary trees [36].

Example 5: Note that the previous example applies to *all* trees, not just binary ones. In particular, it applies to the tree consisting of one root with k children. This is equivalent to structures with $D = \{0, 1, \ldots, k\}$ and the multimorphism $\langle \min_0, \max_0\rangle$ from Example 3. This is a natural generalisation of submodular ($k = 1$) and bisubmodular ($k = 2$) functions, known as k-submodular functions [27]. The tractability of k-submodular valued structures for $k > 2$ was previously open.

Example 6: Let b and c be two distinct elements of D and let $(D; <)$ be a partial order which relates all pairs of elements except for b and c. A pair $\langle g_1, g_2\rangle$, where $g_1, g_2 : D^2 \to D$ are two binary operations, is a *1-defect* multimorphism if g_1 and g_2 are both commutative and satisfy the following conditions:

- If $\{x, y\} \neq \{b, c\}$, then $g_1(x, y) = x \wedge y$ and $g_2(x, y) = x \vee y$.
- If $\{x, y\} = \{b, c\}$, then $\{g_1(x, y), g_2(x, y)\} \cap \{x, y\} = \emptyset$, and $g_1(x, y) < g_2(x, y)$.

The tractability of valued structures that have a 1-defect multimorphism has recently been shown in [33]. We now show that valued structures with a 1-defect multimorphism are solvable by BLP$_g$.

Without loss of generality, we assume that $g_1(b, c) < b, c$ and write $g = g_1$. (Otherwise, $g_2(b, c) > b, c$, and g_2 is used instead.) Using g, we construct a symmetric m-ary operation $f(x_1, \ldots, x_m)$.

Let $f_1, \ldots, f_M$ be the $M = \binom{m}{2}$ terms $g(x_i, x_j)$. Let $f = g(f_1, g(f_2, \ldots, g(f_{M-1}, f_M)\ldots))$. There are three possible cases:

- $\{b, c\} \not\subseteq x_1, ..., x_m$. Then g acts as $\wedge$, which is a semi-lattice operation, hence so does f.
- $\{b, c\} \subseteq \{x_1, ..., x_m\}$ and $g(b, c) \leq x_1, \ldots, x_m$. Then $f_i = g(b, c)$ for some $1 \leq i \leq M$, and $g(f_i, f_j) = g(b, c)$ for all $1 \leq j \leq M$, so $f(x_1, \ldots, x_m) = g(b, c)$.
- $\{b, c\} \subseteq \{x_1, \ldots, x_m\}$ and there is a variable x_p for some $1 \leq p \leq m$ such that $x_p \leq g(b, c)$ and $x_p \leq x_1, \ldots, x_m$. Then $g(x_p, x_q) = x_p$ for all $1 \leq q \leq m$ so $f_i = x_p$ for some $1 \leq i \leq M$ and $g(f_i, f_j) = x_p$ for all $1 \leq j \leq M$, so $f(x_1, \ldots, x_m) = x_p$.

VI. Conclusions

We have characterised precisely for which valued structures the basic linear programming relaxation (BLP) is a decision procedure. This implies tractability of several

[9] The STP might contain cycles, but [11, Lemma 7.15] tells us that on cycles we have, in the finite-valued case, only unary functions. Since an acyclic tournament is equivalent to a total order on the domain and unary functions are submodular for all possible domain orderings, it follows that the functions admitting the STP must be submodular with respect to some total order.

previously open classes of VCSPs including several generalisations of submodularity. In fact, BLP solves *all known* tractable finite-valued structures.

The main result does not give a decidability criterion for testing whether a valued structure is solvable by BLP; the so-called *meta problem*. This is left as future work.

An intriguing open question is whether our tractability results hold in the oracle-value model; that is, for objective functions which are not given explicitly as a sum of functions, but only by an oracle. For instance, the *maximisation problem* for submodular functions on distributive lattices, known to be NP-complete, allows for good approximation algorithms in both models [23].

Acknowledgments

We thank Andrei Krokhin for many valuable comments on an earlier draft of this manuscript, Prasad Raghavendra for explaining his results, and Páidí Creed for helpful discussion. Johan Thapper was supported by the *CSP-complexity* research project funded by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013 Grant Agreement no. 257039). Stanislav Živný was supported by a Junior Research Fellowship at University College, Oxford.

References

[1] L. Barto, "The dichotomy for conservative constraint satisfaction problems revisited," in *Proceedings of the 26th IEEE Symposium on Logic in Computer Science (LICS'11)*. IEEE Computer Society, 2011, pp. 301–310.

[2] L. Barto and M. Kozik, "Constraint Satisfaction Problems of Bounded Width," in *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS'09)*. IEEE Computer Society, 2009, pp. 461–471.

[3] ——, "Robust Satisfiability of Constraint Satisfaction Problems," in *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC'12)*. ACM, 2012, pp. 931–940.

[4] L. Barto, M. Kozik, and T. Niven, "The CSP dichotomy holds for digraphs with no sources and no sinks (a positive answer to a conjecture of Bang-Jensen and Hell)," *SIAM Journal on Computing*, vol. 38, no. 5, pp. 1782–1802, 2009.

[5] S. Bistarelli, U. Montanari, and F. Rossi, "Semiring-based Constraint Satisfaction and Optimisation," *Journal of the ACM*, vol. 44, no. 2, pp. 201–236, 1997.

[6] A. Bulatov, "A dichotomy theorem for constraint satisfaction problems on a 3-element set," *Journal of the ACM*, vol. 53, no. 1, pp. 66–120, 2006.

[7] A. Bulatov, A. Krokhin, and P. Jeavons, "Classifying the Complexity of Constraints using Finite Algebras," *SIAM Journal on Computing*, vol. 34, no. 3, pp. 720–742, 2005.

[8] A. A. Bulatov, "Complexity of conservative constraint satisfaction problems," *ACM Transactions on Computational Logic*, vol. 12, no. 4, p. 24, 2011.

[9] D. Cohen, M. Cooper, P. Creed, P. Jeavons, and S. Živný, "An algebraic theory of complexity for discrete optimisation," Tech. Rep., July 2012, arXiv:1207.6692.

[10] D. A. Cohen, M. C. Cooper, and P. G. Jeavons, "An Algebraic Characterisation of Complexity for Valued Constraints," in *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP'06)*, ser. Lecture Notes in Computer Science, vol. 4204. Springer, 2006, pp. 107–121.

[11] ——, "Generalising submodularity and Horn clauses: Tractable optimization problems defined by tournament pair multimorphisms," *Theoretical Computer Science*, vol. 401, no. 1-3, pp. 36–51, 2008.

[12] D. A. Cohen, M. C. Cooper, P. G. Jeavons, and A. A. Krokhin, "The Complexity of Soft Constraint Satisfaction," *Artificial Intelligence*, vol. 170, no. 11, pp. 983–1016, 2006.

[13] D. A. Cohen, P. Creed, P. G. Jeavons, and S. Živný, "An algebraic theory of complexity for valued constraints: Establishing a Galois connection," in *Proceedings of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS'11)*, ser. Lecture Notes in Computer Science, vol. 6907. Springer, 2011, pp. 231–242.

[14] M. C. Cooper, S. de Givry, M. Sánchez, T. Schiex, M. Zytnicki, and T. Werner, "Soft arc consistency revisited," *Artificial Intelligence*, vol. 174, no. 7–8, pp. 449–478, 2010.

[15] M. C. Cooper and T. Schiex, "Arc consistency for soft constraints," *Artificial Intelligence*, vol. 154, no. 1-2, pp. 199–227, 2004.

[16] Y. Crama and P. L. Hammer, *Boolean Functions - Theory, Algorithms, and Applications*. Cambridge University Press, 2011.

[17] P. Creed and S. Živný, "On minimal weighted clones," in *Proceedings of the 17th International Conference on Principles and Practice of Constraint Programming (CP'11)*, ser. Lecture Notes in Computer Science, vol. 6876. Springer, 2011, pp. 210–224.

[18] N. Creignou, S. Khanna, and M. Sudan, *Complexity Classification of Boolean Constraint Satisfaction Problems*, ser. SIAM Monographs on Discrete Mathematics and Applications. SIAM, 2001, vol. 7.

[19] V. Dalmau and A. Krokhin, "Robust satisfiability for CSPs: algorithmic and hardness results," 2011, in preparation.

[20] V. Dalmau and J. Pearson, "Set Functions and Width 1 Problems," in *Proceedings of the 5th International Conference on Constraint Programming (CP'99)*, ser. Lecture Notes in Computer Science, vol. 1713. Springer, 1999, pp. 159–173.

[21] V. Deineko, P. Jonsson, M. Klasson, and A. Krokhin, "The approximability of Max CSP with fixed-value constraints," *Journal of the ACM*, vol. 55, no. 4, 2008.

[22] T. Feder and M. Y. Vardi, "The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory," *SIAM Journal on Computing*, vol. 28, no. 1, pp. 57–104, 1998.

[23] U. Feige, V. S. Mirrokni, and J. Vondrák, "Maximizing Non-monotone Submodular Functions," *SIAM Journal on Computing*, vol. 40, no. 4, pp. 1133–1153, 2011.

[24] E. C. Freuder, "Synthesizing Constraint Expressions," *Communications of the ACM*, vol. 21, no. 11, pp. 958–966, 1978.

[25] S. Fujishige and S. Iwata, "Bisubmodular Function Minimization," *SIAM Journal on Discrete Mathematics*, vol. 19, no. 4, pp. 1065–1073, 2005.

[26] D. Gale, *The Theory of Linear Economic Models*. McGraw-Hill, 1960.

[27] A. Huber and V. Kolmogorov, "Towards Minimizing k-Submodular Functions," in *Proceedings of the 2nd International Symposium on Combinatorial Optimization (ISCO'12)*, 2012.

[28] P. M. Idziak, P. Markovic, R. McKenzie, M. Valeriote, and R. Willard, "Tractability and learnability arising from algebras with few subpowers," *SIAM Journal on Computing*, vol. 39, no. 7, pp. 3023–3037, 2010.

[29] S. Iwata, L. Fleischer, and S. Fujishige, "A combinatorial strongly polynomial algorithm for minimizing submodular functions," *Journal of the ACM*, vol. 48, no. 4, pp. 761–777, 2001.

[30] P. G. Jeavons, "On the Algebraic Structure of Combinatorial Problems," *Theoretical Computer Science*, vol. 200, no. 1-2, pp. 185–204, 1998.

[31] P. G. Jeavons, D. A. Cohen, and M. Gyssens, "Closure Properties of Constraints," *Journal of the ACM*, vol. 44, no. 4, pp. 527–548, 1997.

[32] P. Jonsson, A. A. Krokhin, and F. Kuivinen, "Hard constraint satisfaction problems have hard gaps at location 1," *Theoretical Computer Science*, vol. 410, no. 38-40, pp. 3856–3874, 2009.

[33] P. Jonsson, F. Kuivinen, and J. Thapper, "Min CSP on Four Elements: Moving Beyond Submodularity," in *Proceedings of the 17th International Conference on Principles and Practice of Constraint Programming (CP'11)*, ser. Lecture Notes in Computer Science, vol. 6876. Springer, 2011, pp. 438–453.

[34] S. Khanna, M. Sudan, L. Trevisan, and D. Williamson, "The approximability of constraint satisfaction problems," *SIAM Journal on Computing*, vol. 30, no. 6, pp. 1863–1920, 2001.

[35] S. Khot, "On the unique games conjecture (invited survey)," in *Proceedings of the 25th Annual IEEE Conference on Computational Complexity (CCC'10)*. IEEE Computer Society, 2010, pp. 99–121.

[36] V. Kolmogorov, "Submodularity on a tree: Unifying $l^\natural$-convex and bisubmodular functions," in *Proceedings of the 36th International Symposium on Mathematical Foundations of Computer Science (MFCS'11)*, ser. Lecture Notes in Computer Science, vol. 6907. Springer, 2011, pp. 400–411.

[37] V. Kolmogorov and S. Živný, "The complexity of conservative valued CSPs," in *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*. SIAM, 2012, pp. 750–759, full version available on arXiv:1110.2809.

[38] A. Krokhin and B. Larose, "Maximizing Supermodular Functions on Product Lattices, with Application to Maximum Constraint Satisfaction," *SIAM Journal on Discrete Mathematics*, vol. 22, no. 1, pp. 312–328, 2008.

[39] F. Kuivinen, "On the complexity of submodular function minimisation on diamonds," *Discrete Optimization*, vol. 8, no. 3, pp. 459–477, 2011.

[40] G. Kun, R. O'Donnell, S. Tamaki, Y. Yoshida, and Y. Zhou, "Linear programming, width-1 CSPs, and robust satisfaction," in *Proceedings of the 3rd Innovations in Theoretical Computer Science (ITCS'12)*. ACM, 2012, pp. 484–495.

[41] S. L. Lauritzen, *Graphical Models*. Oxford University Press, 1996.

[42] A. K. Mackworth, "Consistency in Networks of Relations," *Artificial Intelligence*, vol. 8, pp. 99–118, 1977.

[43] O. L. Mangasarian, *Nonlinear programming*, ser. Classics in applied mathematics. SIAM, 1994.

[44] S. T. McCormick and S. Fujishige, "Strongly polynomial and fully combinatorial algorithms for bisubmodular function minimization," *Mathematical Programming*, vol. 122, no. 1, pp. 87–120, 2010.

[45] U. Montanari, "Networks of Constraints: Fundamental properties and applications to picture processing," *Information Sciences*, vol. 7, pp. 95–132, 1974.

[46] P. Raghavendra, "Optimal algorithms and inapproximability results for every CSP?" in *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC'08)*. ACM, 2008, pp. 245–254.

[47] T. J. Schaefer, "The Complexity of Satisfiability Problems," in *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC'78)*. ACM, 1978, pp. 216–226.

[48] T. Schiex, H. Fargier, and G. Verfaillie, "Valued Constraint Satisfaction Problems: Hard and Easy Problems," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, 1995, pp. 631–637.

[49] A. Schrijver, "A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time," *Journal of Combinatorial Theory, Series B*, vol. 80, no. 2, pp. 346–355, 2000.

[50] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.