Supermodular Functions and the Complexity of Max CSP ⋆

David Cohen [a], Martin Cooper [b], Peter Jeavons [c], Andrei Krokhin [d,*]

[a] *Department of Computer Science, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK*

[b] *IRIT, University of Toulouse III, 31062 Toulouse, France*

[c] *Computing Laboratory, University of Oxford, Oxford OX1 3QD, UK*

[d] *Department of Computer Science, University of Durham, Durham, DH1 3LE, UK, tel: +44 191 3341743, fax: +44 191 3341701*

Abstract

In this paper we study the complexity of the maximum constraint satisfaction problem (Max CSP) over an arbitrary finite domain. An instance of Max CSP consists of a set of variables and a collection of constraints which are applied to certain specified subsets of these variables; the goal is to find values for the variables which maximize the number of simultaneously satisfied constraints. Using the theory of sub- and supermodular functions on finite lattice-ordered sets, we obtain the first examples of general families of efficiently solvable cases of Max CSP for arbitrary finite domains. In addition, we provide the first dichotomy result for a special class of non-Boolean Max CSP, by considering binary constraints given by supermodular functions on a totally ordered set. Finally, we show that the equality constraint over a non-Boolean domain is non-supermodular, and, when combined with some simple unary constraints, gives rise to cases of Max CSP which are hard even to approximate.

Key words: Complexity, constraint satisfaction problem, optimization, supermodularity

⋆ A preliminary version of some parts of this paper appears in Proceedings of STACS'04, Montpellier, France, 2004.

* Corresponding author.

Email addresses: d.cohen@rhul.ac.uk (David Cohen), cooper@irit.fr (Martin Cooper), peter.jeavons@comlab.ox.ac.uk (Peter Jeavons), andrei.krokhin@durham.ac.uk (Andrei Krokhin).

1 Introduction

The main object of our study in this paper is the maximum constraint satisfaction problem (Max CSP) where one is given a collection of constraints on overlapping sets of variables and the goal is to find an assignment of values to the variables that maximizes the number of satisfied constraints. A number of classic optimization problems including Max 3-Sat, Max Cut and Max Dicut can be represented in this framework, and it can also be used to model optimization problems arising in more applied settings, such as database design [11].

The Max-CSP framework has been well-studied in the Boolean case, that is, when the set of values for the variables is $\{0, 1\}$. Many fundamental results have been obtained, concerning both complexity classifications and approximation properties (see, e.g., [8,9,21,24,25,37]). In the non-Boolean case, a number of results have been obtained that concern approximation properties (see, e.g., [11,14,15,33]). However, there has so far been very little study of efficient exact algorithms, or complexity, for subproblems of non-Boolean Max CSP. This paper presents a general approach which is aimed at filling this gap.

We study a standard parameterized version of the Max CSP, in which restrictions may be imposed on the types of constraints allowed in the instances. In particular, we investigate which restrictions make such problems *tractable*, by allowing a polynomial time algorithm to find an optimal assignment. This setting has been extensively studied and completely classified in the Boolean case [8,9,24,25]. In contrast, we consider here the case where the set of possible values is an *arbitrary finite* set.

Experience in the study of various forms of constraint satisfaction [2–5,23] has shown that the more general form of such problems, in which the domain is an arbitrary finite set, is often considerably more difficult to analyze than the Boolean case. The techniques developed for the Boolean case typically involve the careful manipulation of logical formulas; such techniques do not readily extend to larger domains. For example, Schaefer [31] obtained a complete classification of complexity for the standard constraint satisfaction problem in the Boolean case using such techniques in 1978; although he raised the question of generalizing this result to larger domains in the same paper, little progress was made for the next twenty years.

The key step in the analysis of the standard constraint satisfaction problem [3,4] was the discovery that the characterization of the tractable cases over the Boolean domain can be restated in an algebraic form [23]. This algebraic description of the characterization has also proved to be a key step in the

analysis of the counting constraint satisfaction problem [5] and the quantified constraint satisfaction problem [2]. However, this form of algebraic description does not provide a suitable tool for analyzing the Max CSP, which is our focus here.

The main contribution of this paper is the first general approach to and the first general results about the complexity of subproblems of *non-Boolean* Max CSP. We point out that the characterization of the tractable cases of Max CSP over a Boolean domain can also be restated in an algebraic form, but using a rather different algebraic framework: we show that they can be characterized using the property of *supermodularity*. We also show how this property can be generalized to the non-Boolean case, and hence used to identify large families of tractable subproblems of the non-Boolean Max CSP. Moreover, we give some results to demonstrate how non-supermodularity can cause hardness of the corresponding subproblem.

The properties of sub- and supermodularity have been extensively used to study combinatorial optimization problems in other contexts. In particular, the problem of minimizing a submodular set function has been thoroughly studied, due to its applications across many research areas [17,20,22,26,27]. The dual problem of maximizing a supermodular function has found interesting applications in diverse economic models, such as supermodular games (see [36]). Submodular functions defined on (products of) totally ordered sets correspond precisely to Monge matrices and arrays (see, for example, survey [6]) which play an important role in solving a number of optimization problems including travelling salesman, assignment and transportation problems [6]. Hence this paper also unifies, for the first time, the study of the Max CSP with many other areas of combinatorial optimization.

The structure of the paper is as follows. In Section 2 we discuss the Max CSP problem, its Boolean case, its complexity, and the relevance of sub- and supermodularity. In Sections 3 and 4, we give two different generalizations for the (unique) non-trivial tractable case of Boolean Max CSP: one to general supermodular constraints on restricted types of ordered domains (distributive lattices), and the other to a restricted form of supermodular constraint on more general ordered domains (arbitrary lattices). For the second case, we are able to give a cubic time algorithm, based on a reduction to the Min Cut problem. Section 5 describes a first dichotomy result for non-Boolean Max CSP, namely, for the case when the set of allowed constraints contains all binary supermodular functions on a chain. As further evidence that non-supermodularity causes hardness of Max CSP, Section 6 establishes that, in the non-Boolean case, allowing just the (non-supermodular) equality constraint and unary constraints gives rise to versions of Max CSP that are hard even to approximate. Finally, in Section 7 we discuss our ideas in the light of the results obtained, and describe possible future work.

2 Preliminaries

Throughout the paper, let D denote a finite set, with $|D| > 1$. Let $R_D^{(m)}$ denote the set of all m-ary predicates over D, that is, functions from D^m to $\{0,1\}$, and let $R_D = \bigcup_{m=1}^{\infty} R_D^{(m)}$.

Definition 2.1 *A* constraint *over a set of variables $V = \{x_1, x_2, \ldots, x_n\}$, is an expression of the form $f(\mathbf{x})$ where*

- *$f \in R_D^{(m)}$ is called the* constraint function*;*
- *$\mathbf{x} = (x_{i_1}, \ldots, x_{i_m})$ is called the* constraint scope*.*

The constraint f is said to be satisfied *on a tuple $\mathbf{a} = (a_{i_1}, \ldots, a_{i_m}) \in D^m$ if $f(\mathbf{a}) = 1$.*

Definition 2.2 *An instance of* Max CSP *is a finite collection of constraints $\{f_1(\mathbf{x}_1), \ldots, f_q(\mathbf{x}_q)\}$, $q \geq 1$, over a set of variables $V = \{x_1, \ldots, x_n\}$, where $f_i \in R_D$ for all $1 \leq i \leq q$. The goal is to find an assignment $\phi : V \to D$ that maximizes the number of satisfied constraints.*

Arguably, it is more appropriate for our purposes to consider the $0, 1$ values taken by constraint functions as *integers* and not as Boolean values; the goal in a Max CSP instance is then to maximize the function $f : D^n \to \mathbb{Z}^+$ (where $\mathbb{Z}^+$ is the set all non-negative integers), defined by

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{q} f_i(\mathbf{x}_i).$$

The *weighted* version of the Max CSP problem, in which each constraint $f_i(\mathbf{x}_i)$ has associated weight $\rho_i \in \mathbb{Z}^+$, can be viewed as the problem of maximizing the function

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{q} \rho_i \cdot f_i(\mathbf{x}_i).$$

In fact, the two versions of Max CSP can be shown to be equivalent (as in [9, Lemma 7.2]).

Throughout the paper, $\mathcal{F}$ will denote a *finite* subset of R_D which does not contain any unsatisfiable predicate (taking the value 0 on all tuples of arguments) , and Max CSP($\mathcal{F}$) will denote the restriction of Max CSP to instances where all constraint functions belong to $\mathcal{F}$. The central problem we consider in this paper is the following.

Problem 1 *Classify the complexity of (weighted)* Max CSP($\mathcal{F}$) *for all possible sets $\mathcal{F}$.*

Though we do not solve this problem completely, we produce substantial evidence that, by further exploiting the new ideas and results in this paper, one can make significant progress on this problem.

Recall that **PO** and **NPO** are optimization analogs of **P** and **NP**; that is, they are classes of optimization problems that can be solved in deterministic polynomial time and non-deterministic polynomial time, respectively. We will call problems in **PO** tractable. An optimization problem is called **NP**-hard if it admits a polynomial time Turing reduction from some **NP**-complete problem. The approximation complexity class **APX** consists of all **NPO** problems for which there is a polynomial time approximation algorithm whose performance ratio is bounded by a constant. A problem in **APX** is called **APX**-complete if every problem in **APX** has a special approximation-preserving reduction, called an AP-reduction, to it. It is not hard to show that every **APX**-complete problem is **NP**-hard. For more detailed definitions of approximation and optimization complexity classes and reductions, the reader is referred to [1,9,29].

Proposition 2.3 MAX CSP($\mathcal{F}$) *belongs to* **APX** *for every* $\mathcal{F}$.

Proof. For the case $|D| = 2$, our statement is Theorem 13.2 [28] or Proposition 5.17 [9]. Generalisation to larger finite domains is almost identical to the proofs of the above mentioned results, as we will now show.

Let $\mathcal{I}$ be an instance of MAX CSP($\mathcal{F}$) with m constraints and n variables $x_1, \ldots, x_n$. Let k be the maximum arity of a predicate in $\mathcal{F}$. We may without loss of generality assume that every constraint $f_i(\mathbf{x}_i)$ in $\mathcal{I}$ has k different variables, some of which may be dummy and hence not mentioned explicitly. If t_i is the number of assignments of values to the k variables of f_i that satisfy f_i, then a random assignment of values to all n variables satisfies f_i with probability $p_i = \frac{t_i}{|D|^k} \geq \frac{1}{|D|^k}$. Hence, a random assignment is expected to satisfy $p(\mathcal{I}) = \sum_{i=1}^{m} p_i \geq \frac{m}{|D|^k}$ constraints. An assignment satisfying at least $\frac{m}{|D|^k}$ constraints can be found deterministically as follows. If $D = \{d_1, \ldots, d_l\}$ then $p(\mathcal{I}) = \frac{1}{l} \sum_{j=1}^{l} p(\mathcal{I}[x_1 = d_j])$ where $\mathcal{I}[[x_1 = d_j]$ is $\mathcal{I}$ with x_1 instantiated with the value d_j. Hence, there is some s, $1 \leq s \leq l$, such that $p(\mathcal{I}[x_1 = d_s]) \geq p(\mathcal{I})$. Clearly, we can compute all values $p(\mathcal{I}[x_1 = d_j])$ in polynomial time (just as we computed $p(\mathcal{I})$), find d_s and fix this value for x_1. If we continue like this with x_2 and so on, we will obtain the required assignment. Since the optimum number of satisfied constraints is obviously not greater than m (the total number of constraints), this is a polynomial-time algorithm that satisfies at least $\frac{1}{|D|^k}$ of the optimum number of satisfied constraints in any instance. ∎

A complete classification of the complexity of MAX CSP($\mathcal{F}$) for a two-element set D was obtained in [8,9,25]. Before stating that result we need to give some definitions.

Definition 2.4 *An* endomorphism *of $\mathcal{F}$ is a unary operation π on D such that $f(a_1, \ldots, a_m) = 1 \Rightarrow f(\pi(a_1), \ldots, \pi(a_m)) = 1$ for all $f \in \mathcal{F}$ and all $(a_1, \ldots, a_m) \in D^m$. We will say that $\mathcal{F}$ is a* core *if every endomorphism of $\mathcal{F}$ is injective (i.e. a permutation).*

The intuition here is that if $\mathcal{F}$ is not a core then it has a non-injective endomorphism π, which implies that, for every assignment ϕ, there is another assignment $\pi\phi$ that satisfies all constraints satisfied by ϕ and uses only a restricted set of values, so the problem can be reduced to a problem over this smaller set. For example, if $D = \{0, 1\}$ then $\mathcal{F}$ is a not a core if and only if $f(a, , \ldots, a) = 1$ for some $a \in D$ and all $f \in \mathcal{F}$. Obviously, in this case the assignment that assigns the value a to all variables satisfies all constraints, so it is optimal, and hence MAX CSP($\mathcal{F}$) is trivial.

Definition 2.5 ([9]) *A function $f \in R^{(n)}_{\{0,1\}}$ is called* 2-monotone *if it can be expressed as follows:*

$$f(x_1, \ldots, x_n) = 1 \Leftrightarrow (x_{i_1} \wedge \ldots \wedge x_{i_s}) \vee (\bar{x}_{j_1} \wedge \ldots \wedge \bar{x}_{j_t}),$$

where either of the two disjuncts may be empty (i.e., the values of s or t may be zero).

Theorem 2.6 ([8,9,25]) *Let $\mathcal{F} \subseteq R_{\{0,1\}}$ be a core. If every $f \in \mathcal{F}$ is 2-monotone, then (weighted)* MAX CSP($\mathcal{F}$) *is in* **PO**, *otherwise it is* **APX**-*complete.*

As we announced in the introduction, the main new tools which we introduce to generalize (the tractability part of) this result will be the conditions of sub- and supermodularity. We will consider the most general type of sub- and supermodular functions, that is, those defined on a (general) lattice, as in [35,36]. Recall that a partial order $\sqsubseteq$ on a set D is called a *lattice* order if, for every $x, y \in D$, there exist a greatest lower bound $x \sqcap y$ and a least upper bound $x \sqcup y$. The algebra $\mathcal{L} = (D, \sqcap, \sqcup)$ on D with two binary operations $\sqcap$ and $\sqcup$ is called a *lattice*, and we have $x \sqsubseteq y \Leftrightarrow x \sqcap y = x \Leftrightarrow x \sqcup y = y$. As is well known, every finite lattice has a least element and a greatest element, which we will denote by $0_{\mathcal{L}}$ and $1_{\mathcal{L}}$, respectively. In fact, since D is finite, the existence of both a greatest lower bound for every pair of elements and a greatest element is sufficient for a partial order to be a lattice order. (For more information about lattices, see, e.g., [12].)

For tuples $\mathbf{a} = (a_1, \ldots, a_n)$, $\mathbf{b} = (b_1, \ldots, b_n)$ in D^n, let $\mathbf{a} \sqcap \mathbf{b}$ and $\mathbf{a} \sqcup \mathbf{b}$ denote the tuples $(a_1 \sqcap b_1, \ldots, a_n \sqcap b_n)$ and $(a_1 \sqcup b_1, \ldots, a_n \sqcup b_n)$, respectively.

Definition 2.7 *Let $\mathcal{L} = (D, \sqcap, \sqcup)$ be a lattice. A function $f : D^n \to \mathbb{Z}^+$ is called* submodular *on $\mathcal{L}$ if*

$$f(\mathbf{a} \sqcap \mathbf{b}) + f(\mathbf{a} \sqcup \mathbf{b}) \leq f(\mathbf{a}) + f(\mathbf{b}) \quad \text{for all } \mathbf{a}, \mathbf{b} \in D^n.$$

It is called supermodular *on* $\mathcal{L}$ *if*

$$f(\mathbf{a} \sqcap \mathbf{b}) + f(\mathbf{a} \sqcup \mathbf{b}) \geq f(\mathbf{a}) + f(\mathbf{b}) \quad \text{for all } \mathbf{a}, \mathbf{b} \in D^n.$$

The sets of all submodular and supermodular functions on $\mathcal{L}$*, will be denoted* $\mathrm{Sbmod}_{\mathcal{L}}$ *and* $\mathrm{Spmod}_{\mathcal{L}}$*, respectively.*

Note that sub- and supermodular functions are usually defined to take values in $\mathbb{R}$, but, in the context of Max CSP, it is appropriate to restrict the range to consist of non-negative integers.

The properties of sub- and supermodularity are most often considered for functions defined on subsets of a set, which corresponds to the special case of Definition 2.7 where $|D| = 2$. A function on subsets of a set is submodular if $f(X \cup Y) + f(X \cap Y) \leq f(X) + f(Y)$ for all subsets X, Y, and it is supermodular if the inverse inequality holds [17,27]. The problem of submodular set function minimization has attracted considerable attention from researchers during the last twenty years (see, e.g., [17,20,22,26,27,32]), in particular, due to its numerous applications in combinatorial optimization. Some results have also been obtained that concern minimization of a submodular function defined on a family of subsets [18,20,22,32], or on a finite grid (or integer lattice) [16,34], or on general lattices [35,36].

Observation 2.8 *Let* f_1 *and* f_2 *be submodular functions on a lattice* $\mathcal{L}$*.*

- *For any constants,* $\alpha_1, \alpha_2 \in \mathbb{Z}^+$*, the function* $\alpha_1 f_1 + \alpha_2 f_2$ *is also submodular.*
- *For any number* K*, the function* $f' = K - f_1$*, is supermodular.*
- *The function* f_1 *is submodular on the dual lattice* $\mathcal{L}^\partial$ *obtained by reversing the order of* $\mathcal{L}$*.*

(Corresponding statements also hold when the terms submodular and supermodular are exchanged throughout.)

The next proposition shows that the non-trivial tractable case of Boolean Max CSP identified in Theorem 2.6 can be characterized using supermodularity.

Proposition 2.9 *A function* $f \in R_{\{0,1\}}$ *is 2-monotone if and only if it is supermodular.*

Proof. It is straightforward to verify that every 2-monotone function is supermodular. Indeed, let f be an n-ary 2-monotone function. Fix the lattice order $0 < 1$ on $\{0, 1\}$. Note that the other lattice order on $\{0, 1\}$ is dual, and hence, by Observation 2.8, supermodularity is not affected by the choice of order. Take two n-tuples $\mathbf{a}$ and $\mathbf{b}$ on $\{0, 1\}$. If $f(\mathbf{a}) = 1$ and $\mathbf{a}$ satisfies the disjunct without negations (see Definition 2.5) then so does $\mathbf{a} \sqcup \mathbf{b}$. If $f(\mathbf{a}) = 1$ and $\mathbf{a}$ satisfies the disjunct with negations then so does $\mathbf{a} \sqcap \mathbf{b}$. Obviously, the

two previous assertions hold if we exchange $\mathbf{a}$ and $\mathbf{b}$ throughout. Hence, if at most one of $f(\mathbf{a})$, $f(\mathbf{b})$ is 1, or if $f(\mathbf{a}) = f(\mathbf{b}) = 1$ and $\mathbf{a}$ and $\mathbf{b}$ satisfy different disjuncts, then the supermodularity inequaliy holds. If $f(\mathbf{a}) = f(\mathbf{b}) = 1$ and $\mathbf{a}$ and $\mathbf{b}$ satisfy the same disjunct then it is easy to see that both $\mathbf{a} \sqcap \mathbf{b}$ and $\mathbf{a} \sqcup \mathbf{b}$ satisfy this disjunct. Hence, $f(\mathbf{a} \sqcap \mathbf{b}) = f(\mathbf{a} \sqcup \mathbf{b}) = 1$, and the supermodularity inequality holds as well. It follows from this that, for all choices of $\mathbf{a}$ and $\mathbf{b}$, we have $f(\mathbf{a}) + f(\mathbf{b}) \leq f(\mathbf{a} \sqcap \mathbf{b}) + f(\mathbf{a} \sqcup \mathbf{b})$.

For the converse, we assume that f is supermodular and show that it is 2-monotone. Assume that $\mathbf{c} \sqsubseteq \mathbf{a} \sqsubseteq \mathbf{d}$ (where the order is component-wise), are such that the three tuples are all different and $f(\mathbf{a}) = 1$ while $f(\mathbf{c}) = f(\mathbf{d}) = 0$. Define $\mathbf{b} = (b_1, \ldots, b_n)$ as follows: $b_i = d_i - a_i + c_i$ for all $1 \leq i \leq n$. Note that, since $0 \leq c_i \leq a_i \leq d_i \leq 1$ for all i, we have $b_i \in \{0, 1\}$. Moreover, it can be easily checked that $\mathbf{c} \leq \mathbf{b} \leq \mathbf{d}$ and $\mathbf{a} \sqcap \mathbf{b} = \mathbf{c}$, $\mathbf{a} \sqcup \mathbf{b} = \mathbf{d}$. Then we have $f(\mathbf{a}) + f(\mathbf{b}) \geq 1 > 0 = f(\mathbf{a} \sqcap \mathbf{b}) + f(\mathbf{a} \sqcup \mathbf{b})$, a contradiction with supermodularity of f. It follows that, for every $\mathbf{a}$ such that $f(\mathbf{a}) = 1$, either all $\mathbf{b}$ with $\mathbf{b} \sqsubseteq \mathbf{a}$ satisfy $f(\mathbf{b}) = 1$, or all $\mathbf{b}$ with $\mathbf{b} \sqsupseteq \mathbf{a}$ satisfy $f(\mathbf{b}) = 1$, or both.

Suppose that there is a tuple $\mathbf{a}$ such that $f(\mathbf{b}) = 1$ whenever $\mathbf{b} \sqsubseteq \mathbf{a}$. We will show that there is a unique maximal tuple with this property. Assume, for the contrary, that there are two maximal tuples, $\mathbf{a}_1$ and $\mathbf{a}_2$ with this property. Take any tuple $\mathbf{c}$ such that $\mathbf{c} \sqsubseteq \mathbf{a}_1 \sqcup \mathbf{a}_2$ and let $\mathbf{c}_1 = \mathbf{c} \sqcap \mathbf{a}_1$, $\mathbf{c}_2 = \mathbf{c} \sqcap \mathbf{a}_2$. Note that $f(\mathbf{c}_1) = f(\mathbf{c}_2) = 1$. It is easy to verify that $\mathbf{c} = \mathbf{c}_1 \sqcup \mathbf{c}_2$. By supermodularity of f, we have $f(\mathbf{c}_1) + f(\mathbf{c}_2) = 2 \leq f(\mathbf{c}_1 \sqcap \mathbf{c}_2) + f(\mathbf{c})$. It follows that $f(\mathbf{c}) = 1$, which is a contradiction with the choice of $\mathbf{a}_1, \mathbf{a}_2$. Therefore, if $f(0, \ldots, 0) = 1$ then there is unique maximal element $\mathbf{a}$ such that $f(\mathbf{b}) = 1$ whenever $\mathbf{b} \sqsubseteq \mathbf{a}$. Similarly, if $f(1, \ldots, 1) = 1$ then there is unique minimal element $\mathbf{a}'$ such that $f(\mathbf{b}) = 1$ whenever $\mathbf{b} \sqsupseteq \mathbf{a}'$. Now let $\{j_1, \ldots, j_t\}$ be the set of all indices j such that the j-th component of $\mathbf{a}$ is 0 (if $\mathbf{a}$ exists), and, dually, let $\{i_1, \ldots, i_s\}$ be the set of all indices i such that the i-th component of $\mathbf{a}'$ is 1 (if $\mathbf{a}'$ exists). Clearly, f can now be expressed as shown in Definition 2.5. ∎

Proposition 2.9 is a key step in extending tractability results for Max CSP from the Boolean case to an arbitrary finite domain, as it allows us to re-state Theorem 2.6 in the following form.

Corollary 2.10 *Let $\mathcal{F} \subseteq R_{\{0,1\}}$ be a core. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\{0,1\}}$, then (weighted)* Max CSP($\mathcal{F}$) *is in* **PO**, *otherwise it is* **APX**-*complete.*

3 Supermodular constraints on distributive lattices

In this section we consider constraints given by supermodular functions on a finite *distributive* lattice. Recall that a finite lattice $\mathcal{D} = (D, \sqcap, \sqcup)$ is distributive if and only if it can be represented by subsets of a set A, where the operations $\sqcup$ and $\sqcap$ are interpreted as set-theoretic union and intersection, respectively [12]. It is well-known [12] that A can be chosen so that $|A| \leq |D|$, and the standard representation for $\mathcal{D}$ (see Theorem 5.12 [12]) can clearly be found in constant time for any fixed D. Note that if $\mathcal{D}$ is a finite distributive lattice, then the product lattice $\mathcal{D}^n = (D^n, \sqcap, \sqcup)$ is also a finite distributive lattice, which can be represented by subsets of a set of size at most $|D| \cdot n$, since every element of $\mathcal{D}$ can be represented using at most $|D|$ bits.

It was shown in [22,32] that a submodular function on a finite distributive lattice [1], which is representable by subsets of an n-element set, can be minimized in polynomial time in n (assuming that computing the value of the function on a given argument is a primitive operation). The complexity of the best known algorithm is $O(n^5 \min\{\log nM, n^2 \log n\})$ where M is an upper bound for the values taken by the function [22].

Using this result, and the correspondence between sub- and supermodular functions, we obtain the following general result about tractable subproblems of Max CSP.

Theorem 3.1 *Weighted* Max CSP($\mathcal{F}$) *is in* **PO** *whenever* $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{D}}$ *for some distributive lattice* $\mathcal{D}$ *on* D*.*

Proof. Assume that $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{D}}$, and let

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{q} \rho_i \cdot f_i(\mathbf{x}_i)$$

be an instance of weighted Max CSP($\mathcal{F}$). If we set $W = \sum_{i=1}^{q} \rho_i$, then $f' = W - f$ is an n-ary submodular function on $\mathcal{D}$, and the minimizers of f' are exactly the maximizers of f. Clearly, computing the value of f' on a given argument can be done in linear time. Note that f' can be seen as a unary submodular function on the lattice $\mathcal{D}^n$. Since $\mathcal{D}$ can be represented by Boolean tuples of (fixed) length at most $|D|$, the lattice $\mathcal{D}^n$ can be represented by Boolean tuples of (fixed) length at most $|D|n$, that is, by subsets of a set with at most $|D|n$ elements. Thus, we can apply, for example, the submodular set function minimization algorithm from [22] to maximize f in polynomial time. ∎

[1] Referred to in [32] as a *ring family*.

It is currently not known whether submodular functions on non-distributive lattices can be minimized in polynomial time, and this problem itself is of interest due to some applications (see [22]). Obviously, any progress in this direction would imply that MAX CSP for supermodular constraints on the corresponding lattices could also be solved efficiently.

4 Generalized 2-monotone constraints

In this section we give a cubic-time algorithm for solving MAX CSP($\mathcal{F}$) when $\mathcal{F}$ consists of supermodular functions of a special form which generalizes the class of 2-monotone Boolean constraints defined above. Throughout this section $\mathcal{L}$ denotes an arbitrary (that is, not necessarily distributive) finite lattice.

Definition 4.1 *A function $f \in R_D^{(n)}$ will be called* generalized 2-monotone *on a lattice $\mathcal{L}$ on D if it can be expressed as follows*

$$f(\mathbf{x}) = 1 \Leftrightarrow ((x_{i_1} \sqsubseteq a_{i_1}) \wedge \ldots \wedge (x_{i_s} \sqsubseteq a_{i_s})) \vee ((x_{j_1} \sqsupseteq b_{j_i}) \wedge \ldots \wedge (x_{j_t} \sqsupseteq b_{j_t})) \quad (1)$$

where $\mathbf{x} = (x_1, \ldots, x_n)$, $a_{i_1}, \ldots, a_{i_s}, b_{j_1}, \ldots, b_{j_t} \in D$, and either of the two disjuncts may be empty (i.e., the value of s or t may be zero).

It is easy to check that all generalized 2-monotone functions are supermodular (but the converse is not true in general). To obtain an efficient algorithm for MAX CSP($\mathcal{F}$) when $\mathcal{F}$ consists of generalized 2-monotone functions, we construct a reduction to the MIN CUT problem, which is known to be solvable in cubic time [19].

To describe the reduction, we need to give some more notation and definitions. Recall that a *principal ideal* in a lattice $\mathcal{L}$ is a set of the form $\{x \in \mathcal{L} \mid x \sqsubseteq a\}$, for some $a \in \mathcal{L}$, and a *principal filter* (or dual ideal) is a set of the form $\{x \in \mathcal{L} \mid x \sqsupseteq b\}$, for some $b \in \mathcal{L}$. For any generalized 2-monotone function f, we will call the first disjunct in Equation 1 of Definition 4.1 (containing conditions of the form $x \sqsubseteq a$), the *ideal part* of f, and the second disjunct in this equation (containing conditions of the form $x \sqsupseteq b$), the *filter part* of f.

For any lattice $\mathcal{L}$, and any $c, d \in \mathcal{L}$, we shall write $c \prec d$ if $c \sqsubset d$ and there is no $u \in \mathcal{L}$ with $c \sqsubset u \sqsubset d$. Finally, let B_b denote the set of all maximal elements in $\{x \in \mathcal{L} \mid x \not\sqsupseteq b\}$. Now we are ready to describe the digraph used in the reduction.

Definition 4.2 Let $\mathcal{L}$ be a lattice on a finite set D, and let $\mathcal{F}$ be a set of generalized 2-monotone functions on $\mathcal{L}$.

Let $\mathcal{I} = \{\rho_1 \cdot f_1(\mathbf{x}_1), \ldots, \rho_q \cdot f_q(\mathbf{x}_q)\}$, $q \geq 1$, be an instance of weighted

MAX CSP($\mathcal{F}$), over a set of variables $V = \{x_1, \ldots, x_n\}$, and let ∞ denote an integer greater than $\sum \rho_i$.

We construct a digraph $G_{\mathcal{I}}$ as follows:

- The vertices of $G_{\mathcal{I}}$ are as follows
  - $\{T, F\} \cup \{x_d \mid x \in V, d \in D\} \cup \{e_i, \bar{e}_i \mid i = 1, 2, \ldots, q\}$.
    For each f_i where the ideal part is empty, we identify the vertices e_i and F. Similarly, for each f_i where the filter part is empty, we identify the vertices $\bar{e}_i$ and T.
- The arcs of $G_{\mathcal{I}}$ are defined as follows:
  - For each $c \prec d$ in $\mathcal{L}$ and for each $x \in V$, there is an arc from x_c to x_d with weight ∞;
  - For each f_i, there is an arc from $\bar{e}_i$ to e_i with weight ρ_i;
  - For each f_i, and each conjunct of the form $x \sqsubseteq a$ in f_i, there is an arc from e_i to x_a with weight ∞;
  - For each f_i, and each conjunct of the form $x \sqsupseteq b$ in f_i, there is an arc from every x_u, where $u \in B_b$, to $\bar{e}_i$ with weight ∞.

Arcs with weight less than ∞ will be called *constraint* arcs.

It is easy to see that $G_{\mathcal{I}}$ is a digraph with source T and sink F. The number of vertices in $G_{\mathcal{I}}$ is at most $2 + n \cdot |D| + 2q$, and the number of edges at most $n|D|^2 + q(1 + |D| + |D|^2)$.

Example 1 Let $\mathcal{L}_\diamond$ be the lattice on $\{0, a, b, 1\}$ such that $0 = 0_{\mathcal{L}_\diamond}$, $1 = 1_{\mathcal{L}_\diamond}$, and the "middle" elements a and b are incomparable. Consider the instance $\mathcal{I}$ of MAX CSP($\mathcal{F}$) corresponding to maximizing the following function:

$$f(x, y) = \rho_1 \cdot f_1(x) + \rho_2 \cdot f_2(x) + \rho_3 \cdot f_3(x, y) + \rho_4 \cdot f_4(y)$$

where the constraint functions f_i are defined as follows:

$$\begin{aligned} f_1(x) &= 1 \Leftrightarrow (x \sqsubseteq a) \\ f_2(x) &= 1 \Leftrightarrow (x \sqsupseteq b) \\ f_3(x, y) &= 1 \Leftrightarrow (y \sqsubseteq 0) \vee (x \sqsupseteq 1) \\ f_4(y) &= 1 \Leftrightarrow (y \sqsupseteq 1) \end{aligned}$$

Note that, in $\mathcal{L}_\diamond$, $B_1 = \{a, b\}$, and $B_b = \{a\}$. One can check that the digraph shown in Figure 1 is the graph $G_{\mathcal{I}}$ specified in Definition 4.2 above.

We will now show how any instance $\mathcal{I}$ of weighted MAX CSP($\mathcal{F}$) can be reduced to computing a minimum cut in the graph $G_{\mathcal{I}}$.

Theorem 4.3 *Let $\mathcal{L}$ be a lattice on a finite set D. If $\mathcal{F}$ consists of generalized 2-monotone functions on $\mathcal{L}$, then (weighted)* MAX CSP($\mathcal{F}$) *is solvable in*

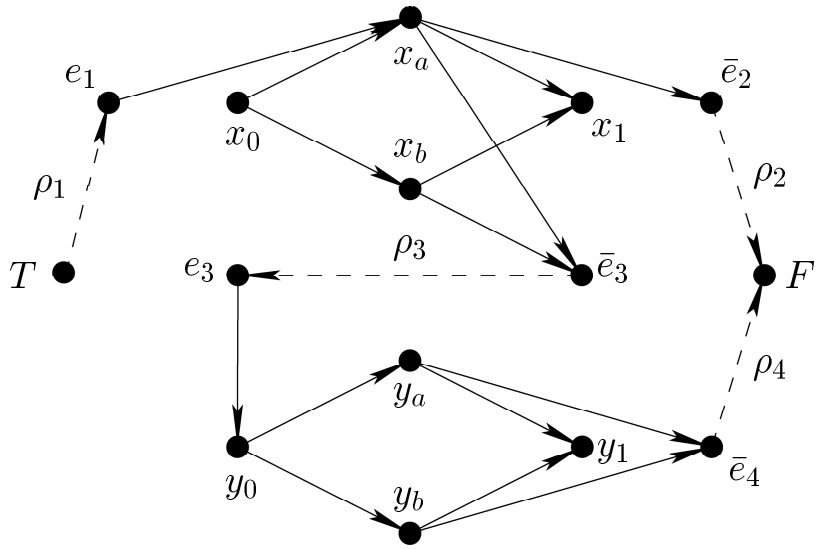

Fig. 1. Example of digraph $G_{\mathcal{I}}$. Dashed lines denote constraint arcs, and solid lines denote arcs of weight ∞.

$O(q^3 + n^3|D|^3)$ time, where q is the number of constraints and n is the number of variables in an instance.

Proof. Let $\mathcal{L}$ be an arbitrary lattice on the finite set D, and let $\mathcal{F}$ be a set of generalized 2-monotone functions on $\mathcal{L}$.

Let $\mathcal{I} = \{\rho_1 \cdot f_1(\mathbf{x}_1), \ldots, \rho_q \cdot f_q(\mathbf{x}_q)\}$, $q \geq 1$, be an instance of weighted Max CSP($\mathcal{F}$), over a set of variables $V = \{x_1, , \ldots, , x_n\}$.

Define the *deficiency* of an assignment ϕ as the difference between $\sum_{i=1}^{q} \rho_i$ and the evaluation of ϕ on $\mathcal{I}$. In other words, the deficiency of ϕ is the total weight of constraints not satisfied by ϕ. We will prove that minimal cuts in $G_{\mathcal{I}}$ exactly correspond to optimal assignments to $\mathcal{I}$. More precisely, we will show that, for each minimal cut in $G_{\mathcal{I}}$ with weight ρ, there is an assignment for $\mathcal{I}$ with deficiency at most ρ, and, for each assignment to $\mathcal{I}$ with deficiency ρ', there is a cut in $G_{\mathcal{I}}$ with weight ρ'.

The semantics of the construction of $G_{\mathcal{I}}$ will be as follows: the vertices of the form x_a correspond to assertions of the form $x \sqsubseteq a$, and arcs between these vertices denote implications about these assertions. Given a minimal cut in $G_{\mathcal{I}}$, we will call a vertex x_a *reaching* if F can be reached from it without crossing the cut. Furthermore, if a vertex x_a is reaching then this will designate that the corresponding assertion is false, and otherwise the corresponding assertion is true. A constraint is not satisfied if and only if the corresponding constraint arc crosses the cut.

Let C be a minimal cut in $G_{\mathcal{I}}$. Obviously, C contains only constraint arcs. First we show that, for every variable $x \in V$, there is a unique minimal element

$a \in \mathcal{L}$ (depending on x) such that x_a is non-reaching. Indeed, assume that there are two such minimal elements, a and a'. Let $c = a \sqcap a'$. Then x_c is reaching, that is, there is a path in $G_{\mathcal{I}}$ from x_c to F not crossing the cut. Consider the first arc in this path containing a vertex *not* of the form x_r. By construction of $G_{\mathcal{I}}$, it has to be an arc of the form $(x_{c'}, \bar{e}_i)$ for some $c' \sqsupseteq c$ such that $\mathcal{I}$ contains a constraint $f_i(\mathbf{x}_i)$ whose filter part has a conjunct $x \sqsupseteq b$ and $c' \in B_b$. Assume first that both $a \sqsupseteq b$ and $a' \sqsupseteq b$. Then, by the choice of c, we have $c' \sqsupseteq c \sqsupseteq b$ which contradicts the condition $c' \in B_b$. Now assume without loss of generality that $a \not\sqsupseteq b$. Then there is $d \in B_b$ such that $d \sqsupseteq a$. It follows that $G_{\mathcal{I}}$ contains an arc $(x_d, \bar{e}_i)$, as part of the construction of $G_{\mathcal{I}}$ corresponding to the constraint $f_i(\mathbf{x}_i)$ whose filter part has a conjunct $x \sqsupseteq b$. Then there is a path from x_a to $\bar{e}_i$ consisting of non-constraint arcs (and hence not crossing the cut), and a path from $\bar{e}_i$ to F (which is a part of the path from x_c to F) that does not cross the cut either. It follows that there is a path from x_a to F that does not cross the cut, which contradicts the assumption that x_a is non-reaching. So, we cannot have more than one minimal element $a \in \mathcal{L}$ such that x_a is non-reaching. It remains to notice that $x_{1_{\mathcal{L}}}$ is always non-reaching, since $1_{\mathcal{L}} \notin B_b$ for any $b \in \mathcal{L}$.

Define an assignment ϕ_C as follows:

$$\phi_C(x) \text{ is the unique minimal element } a \text{ such that } x_a \text{ is non-reaching.}$$

Suppose that a constraint arc is not in the cut. The assignment satisfies the filter part of the corresponding constraint if the arc is on the F side of the cut, and it satisfies the ideal part of the constraint otherwise. To establish this, suppose first that the constraint arc is of the form (T, e_i), that is, it corresponds to a constraint with an empty filter part. Then, for every vertex x_a such that there is an arc (e_i, x_a), the assertion $\phi_C(x) \sqsubseteq a$ is true, since otherwise x_a is reaching and F would be reachable from T. Similarly, if the constraint arc is of the form $(\bar{e}_i, F)$, then every vertex x_a, such that $(x_a, \bar{e}_i)$ is an arc, is reaching, and, therefore, the assertion $\phi_C(x) \sqsubseteq a$ is false. This implies that the filter part of the constraint f_i is satisfied, since, for any x, if $\phi_C(x) \not\sqsubseteq a$ for all $a \in B_b$ then $\phi_C(x) \sqsupseteq b$. Finally, suppose that the arc is of the form $(\bar{e}_i, e_i)$. Then, if there is a reaching vertex x_a such that there is an arc (e_i, x_a) then every vertex y_c, where c is such that there is an arc $(y_c, \bar{e}_i)$, is also reaching, which implies that $\phi_C(y) \not\sqsubseteq c$ for such y and c, and hence the filter part of the constraint f_i is satisfied. If all such vertices x_a are non-reaching then all assertions $\phi_C(x) \sqsubseteq a$ are true and the ideal part of f_i is satisfied. Therefore, the deficiency of ϕ_C is not greater than the weight of C.

Conversely, let ϕ be an assignment to $\mathcal{I}$, and let K be the set of constraints in $\mathcal{I}$ that give a zero evaluation on ϕ. Consider any path from T to F. By construction of $G_{\mathcal{I}}$, this path has the following structure: the first two arcs are (T, e_i) and (e_i, x_a) for some i, $x \in V$, and $a \in D$. Then the path goes up in the

x-copy of the digraph representing $\mathcal{L}$ (which is, in fact, the Hasse diagram of $\mathcal{L}$) to some vertex x_b with $a \sqsubseteq b$. Then the path goes via arcs of the form $(x_b, \bar{e}_j)$, $(\bar{e}_j, e_j)$, (e_j, y_c) to the y-copy of the digraph representing $\mathcal{L}$, where $y \in V$ and $c \in D$. It travels up this copy to some other vertex y_d and then via a triple of arcs as above, and so on. The final part of this path consists of arcs $(z_t, \bar{e}_k)$, $(\bar{e}_k, F)$. We examine, in order, the constraint arcs along this path, replacing every disjunct of the form $(x_1 \sqsupseteq b_1) \wedge \ldots \wedge (x_t \sqsupseteq b_t)$ in every constraint by an equivalent expression $(\bigwedge_{c \in B_{b_1}} \neg(x_1 \sqsubseteq c)) \wedge \ldots \wedge (\bigwedge_{c \in B_{b_t}} \neg(x_t \sqsubseteq c))$. Then we obtain a sequence of assertions of the following form:

$$
\begin{array}{rl}
 & (\ldots \wedge (x_{i_1} \sqsubseteq a_1) \wedge \ldots) \\
(\ldots \wedge \neg(x_{i_1} \sqsubseteq b_2) \wedge \ldots) \vee (\ldots \wedge (x_{i_2} \sqsubseteq a_2) \wedge \ldots) & \text{for some } b_2 \sqsupseteq a_1 \\
 & \vdots \\
(\ldots \wedge \neg(x_{i_{k-1}} \sqsubseteq b_k) \wedge \ldots) \vee (\ldots \wedge (x_{i_k} \sqsubseteq a_k) \wedge \ldots) & \text{for some } b_k \sqsupseteq a_{k-1} \\
(\ldots \wedge \neg(x_{i_k} \sqsubseteq b_{k+1}) \wedge \ldots) & \text{for some } b_{k+1} \sqsupseteq a_k
\end{array}
$$

Since the second part of each assertion contradicts the first part of the next, these assertions cannot all hold simultaneously, so one of the corresponding constraints must in fact give a zero evaluation on ϕ. Hence, every path from T to F includes at least one edge corresponding to a constraint from K, and so the edges corresponding to the set K form a cut in $G_{\mathcal{I}}$. Furthermore, by the choice of K, the weight of this cut is equal to the deficiency of ϕ.

It follows that the standard algorithm [19] for the Min Cut problem can be used to find an optimal assignment for any instance of Max CSP($\mathcal{F}$). This algorithm runs in $O(k^3)$ where k is the number of vertices in the graph. Since the number of vertices in $G_{\mathcal{I}}$ is at most $2 + n \cdot |D| + 2q$, the result follows. ∎

Note that, unlike in the previous section, the lattice $\mathcal{L}$ is *not* required to be represented (as a poset) by subsets of a set, which may have required exponential blow-up.

Theorem 4.3 shows that when the constraints in a Max CSP instance are described by generalized 2-monotone functions, then an optimal solution can be found much more efficiently than by invoking the general algorithm for minimizing submodular functions. Moreover, for non-distributive lattices $\mathcal{L}$, the obtained class of constraints will, in general, not be a subclass of the constraints studied in the previous section, and hence the known forms of SFM algorithms may not be applicable at all in this case.

5 Binary supermodular constraints on a chain

In this section we consider supermodular functions on a finite *totally ordered* lattice, or *chain*. One reason why chains are especially interesting in our study is the following lemma.

Lemma 5.1 *Every unary function is supermodular on a lattice $\mathcal{L}$ if and only if $\mathcal{L}$ is a chain.*

Proof. It is straightforward to check that if $\mathcal{L}$ is a chain then every function $f \in R_D^{(1)}$ is supermodular on $\mathcal{L}$, as the inequality in the supermodularity condition becomes equality. For the converse, assume that $\mathcal{L}$ is not a chain. This implies that $|D| > 2$, and $\mathcal{L}$ has two incomparable elements a, b. Since a and b are incomparable, we have $\{a \sqcup b, a \sqcap b\} \cap \{a, b\} = \emptyset$. Consider the function f such that $f(a) = 1$ and $f(x) = 0$ otherwise. It is easy to see that f is not supermodular. ∎

It is easy to see that a chain is a distributive lattice, which implies that Theorem 3.1 can be applied, and hence that Max CSP($\mathcal{F}$) is tractable for all sets $\mathcal{F}$ consisting of supermodular constraints on a chain. Furthermore, by Lemma 5.1, such sets of functions can include all unary functions.

We will now show that, for supermodular constraints which are at most *binary*, this result can be further strengthened, to obtain a more efficient optimization algorithm.

Theorem 5.2 *Let $\mathcal{C}$ be a chain on a finite set D. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{C}}$, and each $f \in \mathcal{F}$ is at most binary, then* Max CSP($\mathcal{F}$) *is solvable in $O(n^3|D|^3)$ time, where n is the number of variables in an instance.*

Proof. Let $f(x_1, \ldots, x_n) = \sum_{i=1}^{q} \rho_i \cdot f_i(\mathbf{x}_i)$ be an instance $\mathcal{I}$ of Max CSP($\mathcal{F}$). Consider the function $f'(x_1, \ldots, x_n) = \sum_{i=1}^{q} \rho_i \cdot (1 - f_i(\mathbf{x}_i))$. Note that the minimizers of f' are exactly the maximizers of f and that, for every $1 \leq i \leq q$, the function $1 - f_i(\mathbf{x}_i)$ is submodular on $\mathcal{C}$. Theorem 4.7 [7] states that the problem of minimizing functions of the form $\sum_{i=1}^{q} g_i(\mathbf{x}_i)$ where every g_i is submodular on $\mathcal{C}$ and at most binary can be solved exactly in $O(n^3|D|^3)$ time, and the result follows. ∎

The next theorem is the main result of this section. It shows that the *only* tractability-preserving way of extending the set $\mathcal{F}$ from Theorem 5.2 is with further supermodular functions; all other extensions give rise to hard problems. Hence, it provides the first *dichotomy* result for a large class of non-Boolean Max CSP problems.

Theorem 5.3 *Let $\mathcal{C}$ be a chain on a finite set D, and let $\mathcal{F} \subseteq R_D$ contain all at most binary supermodular functions on $\mathcal{C}$. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{C}}$, then (weighted)* Max CSP($\mathcal{F}$) *is in* **PO**, *otherwise it is* **NP**-*hard.*

Proof. If all functions in $\mathcal{F}$ are supermodular then the result follows from Theorem 3.1. For the converse, assume that $\mathcal{F}$ contains a non-supermodular function $g \in R_D^{(k)}$. We will show that in this case Max CSP($\mathcal{F}$) is **NP**-hard. Since g is not supermodular on $\mathcal{C}$, there exist $\mathbf{a}, \mathbf{b} \in D^k$ such that $g(\mathbf{a} \sqcap \mathbf{b}) + g(\mathbf{a} \sqcup \mathbf{b}) < g(\mathbf{a}) + g(\mathbf{b})$. Note that, since $\mathcal{C}$ is a chain, both $a_i \sqcup b_i$ and $a_i \sqcap b_i$ are in $\{a_i, b_i\}$ for all $1 \leq i \leq k$. For $1 \leq i \leq k$, define functions $t_i : \{0,1\} \to \{a_i, b_i\}$ by the following rule.

- if $a_i = b_i$ then $t_i(0) = t_i(1) = a_i$;
- if $a_i \sqsubset b_i$ then $t_i(0) = a_i$ and $t_i(1) = b_i$;
- if $b_i \sqsubset a_i$ then $t_i(0) = b_i$ and $t_i(1) = a_i$.

Then it is easy to check that the function $g' \in R^{(k)}_{\{0,1\}}$ defined by the rule

$$g'(x_1, \ldots, x_k) = g(t_1(x_1), \ldots, t_k(x_k))$$

is a Boolean non-supermodular function. We will need unary functions c'_0, c'_1 on $\{0,1\}$ which are defined as follows $c'_i(x)$ is 1 if $x = i$ and 0 otherwise. It follows from Theorem 2.6 and Proposition 2.9 that Max CSP($\mathcal{F}'$) on $\{0,1\}$, where $\mathcal{F}' = \{g', c'_0, c'_1\}$, is **NP**-hard. (Note that that we include c'_0, c'_1 to ensure that $\mathcal{F}'$ is a core). We will give a polynomial time reduction from this problem to (weighted) Max CSP($\mathcal{F}$).

In the reduction, we will use functions $h_i(x, y)$, $1 \leq i \leq k$, defined by the rule

$$h_i(x, y) = 1 \Leftrightarrow ((x \sqsubseteq 0) \wedge (y \sqsubseteq t_i(0))) \vee ((x \sqsupseteq 1) \wedge (y \sqsupseteq t_i(1))).$$

It is easy to see that these functions are generalized 2-monotone. In particular, they are supermodular on $\mathcal{C}$. Assume without loss of generality that $0, 1 \in D$. Other functions used in the reduction are from $R_D^{(1)}$, and are defined as follows:

- for each $d \in D$, let $c_d(x) = 1$ if and only if $x = d$;
- for each $1 \leq i \leq k$, let $\bar{c}_i(x) = 1$ if and only if $x \in \{a_i, b_i\}$;
- let $c_{01}(x) = 1$ if and only if $x \in \{0, 1\}$.

By Lemma 5.1, all these functions are supermodular.

Let $f'(x_1, , \ldots, , x_n) = \sum_{i=1}^{q} \rho_i \cdot f'_i(\mathbf{x}_i)$ be an instance $\mathcal{I}'$ of Max CSP($\mathcal{F}'$), over the set $V = \{x_1, \ldots, x_n\}$ of variables. Let $W = \sum \rho_i + 1$. Construct an instance $\mathcal{I}$ of Max CSP($\mathcal{F}$) containing all variables from V and further variables and constraints as follows.

- For every $1 \leq i \leq q$ such that $f'_i(\mathbf{x}_i) = g'(x_{j_1}, \ldots, x_{j_k})$, introduce

  - k new variables $y^i_{j_1}, \dots, y^i_{j_k}$,
  - constraint $g(y^i_{j_1}, \dots, y^i_{j_k})$ with weight ρ_i,
  - constraints $\bar{c}_1(y^i_{j_1}), \dots, \bar{c}_k(y^i_{j_k})$, each with weight W
  - constraints $h_1(x_{j_1}, y^i_{j_1}), \dots, h_k(x_{j_k}, y^i_{j_k})$, each with weight W;
- for every $1 \leq i \leq q$ such that $f'_i(\mathbf{x}_i) = c'_0(x_{j_1})$, introduce constraint $c_0(x_{j_1})$ with weight ρ_i;
- for every $1 \leq i \leq q$ such that $f'_i(\mathbf{x}_i) = c'_1(x_{j_1})$, introduce constraint $c_1(x_{j_1})$ with weight ρ_i;
- for every variable $x_i \in V$, introduce constraint $c_{01}(x_i)$ with weight W.

It is easy to see that $\mathcal{I}$ can be built from $\mathcal{I}'$ in polynomial time. Let l be the number of constraints with weight W in $\mathcal{I}$.

For every assignment ϕ' to $\mathcal{I}'$, let ϕ be an assignment to $\mathcal{I}$ which coincides with ϕ' on V, and, for every variable $y^i_{j_s}$, set $\phi(y^i_{j_s}) = t_s(\phi'(x_{j_s}))$. It is easy to see that ϕ satisfies all constraints of weight W. Moreover, every constraint of the form $c'_i(x_{j_1})$, $i \in \{0, 1\}$, in $\mathcal{I}'$ is satisfied if and only the corresponding constraint $c_i(x'_{j_1})$ in $\mathcal{I}$ is satisfied. It follows from the construction of the function g' and the choice of functions h_i, $\bar{c}_i$, and c_{01} in $\mathcal{I}$ that a constraint $f'_i(\mathbf{x}_i)$ in $\mathcal{I}'$ with the constraint function g' is satisfied if and only if the corresponding constraint with constraint function g in $\mathcal{I}$ is satisfied. Hence, if the total weight of satisfied constraints in $\mathcal{I}'$ is ρ then the total weight of satisfied constraints in $\mathcal{I}$ is $l \cdot W + \rho$.

In the other direction, it is easy to see that every optimal assignment ϕ to $\mathcal{I}$ satisfies all constraints of weight W, therefore its weight is $l \cdot W + \rho$ for some $\rho < W$. In particular, it follows that $\phi(x) \in \{0, 1\}$ for every $x \in V$. Let ϕ' be an assignment to $\mathcal{I}'$ that is the restriction of ϕ to V. Then the total weight of satisfied constraints in $\mathcal{I}'$ is ρ. Indeed, this follows from the fact that all constraints of the form h_i, $\bar{c}_i$, and c_{01} are satisfied, that all variables $y^i_{j_s}$, $1 \leq s \leq k$, take values in the corresponding sets $\{a_s, b_s\}$, and these values can always be recovered from the values of the variables x_{j_s} by using the functions t_s. Thus, optimal assignments to $\mathcal{I}$ and to $\mathcal{I}'$ exactly correspond to each other, and the result follows. ∎

6 A simple non-supermodular constraint

We have established in the previous section that for chains, in the presence of all binary supermodular functions, supermodularity is the only possible reason for tractability. It can be shown using results of [30] that the binary supermodular functions on a finite chain determine the chain (up to reverse order). However, by Lemma 5.1, all *unary* functions are supermodular on *every* chain.

It is therefore an interesting question to determine whether supermodularity on a chain is the only possible reason for tractability of MAX CSP($\mathcal{F}$) when $\mathcal{F}$ contains all *unary* functions [2].

In this section we give some evidence in favour of a positive answer to this question, by considering a simple equality constraint. Interestingly, in all of the various versions of the constraint satisfaction problem for which complexity classifications have previously been obtained, an equality constraint can be combined with any tractable set of constraints without affecting tractability. However, we show here that such a constraint gives rise to hard subproblems of MAX CSP, in the presence of some simple unary constraints.

Definition 6.1 *Let D be a finite set. We define the function $f_{eq} \in R_D^{(2)}$, and the functions $c_d \in R_D^{(1)}$ for each $d \in D$, as follows*

$$f_{eq}(x, y) = 1 \Leftrightarrow (x = y),$$
$$c_d(x) = 1 \Leftrightarrow (x = d).$$

It is easy to check that f_{eq} on D is supermodular if $|D| = 2$. However, the next result shows that $|D| = 2$ is the only case for which this is true.

Lemma 6.2 *If $|D| > 2$ then $f_{eq}(x, y)$ is not supermodular on any lattice on D.*

Proof: If $\mathcal{L}$ is a lattice on D, and $|D| > 2$, then there exists $a \in \mathcal{L}$ such that $0_{\mathcal{L}} \sqsubset a \sqsubset 1_{\mathcal{L}}$. It is easy to check that the supermodularity condition for f_{eq} fails on the pairs $(0_{\mathcal{L}}, 1_{\mathcal{L}})$ and (a, a). ∎

Note that MAX CSP($\{f_{eq}\}$) is clearly tractable. However, this does not give us an interesting tractable subproblem of MAX CSP, since $\{f_{eq}\}$ is not a core. In fact, the core obtained from $\{f_{eq}\}$ is one-element.

The next theorem shows that the equality constraint f_{eq}, when considered together with the set of unary functions c_d (to make a core), gives rise to a hard problem.

In the proof of Theorem 6.4, we will use a form of reduction known as an L-reduction [1,29], which is defined as follows.

[2] We remark that if $\mathcal{F}$ contains all unary functions, then a problem of the form MAX CSP($\mathcal{F}$) is the optimization version of a *conservative* constraint satisfaction problem [4], in which one can specify arbitrary constraints restricting the domain for individual variables.

Definition 6.3 ([1,29]) *An* L-reduction *from an optimization problem* A *to an optimization problem* B *is a quadruple* (f, g, α, β)*, where* f *and* g *are polynomial time algorithms and* $\alpha, \beta > 0$ *are constants, such that the following conditions hold.*

(a) given any instance a *of* A*, algorithm* f *produces an instance* $b = f(a)$ *of* B*, such that the cost of an optimal solution for* b*,* $OPT(b)$*, is at most* $\alpha \cdot OPT(a)$*;*
(b) given $a, b = f(a)$*, and any solution* y *to* b*, algorithm* g *produces a solution* x *to* a *such that* $|cost(x) - OPT(a)| \leq \beta \cdot |cost(y) - OPT(b)|$*.*

By Lemma 8.2 of [1], any problem in **APX** which has an L-reduction from an **APX**-complete problem is itself **APX**-complete.

Theorem 6.4 *For any finite set* D *with* $|D| > 2$*, if* $\mathcal{F} \supseteq \{c_d \mid d \in D\} \cup \{f_{eq}\}$*, then* Max CSP($\mathcal{F}$) *is* **APX***-complete.*

Proof. Let $\mathcal{F} \supseteq \{c_d \mid d \in D\} \cup \{f_{eq}\}$. By Proposition 2.3, Max CSP($\mathcal{F}$) is in **APX**.

To establish **APX**-completeness, we will give an L-reduction from the Max Cut problem, which is known to be **APX**-complete [1,29]. In this problem, one is given an undirected graph and the goal is to partition the vertices into two classes so that the number of edges connecting a vertex in one class to a vertex in the other is as large as possible.

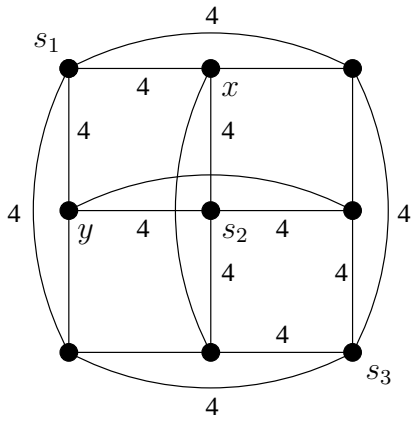

Fig. 2. "Gadget" graph C adapted from [10].

Let $G = (V, E)$ be a graph. Using a construction adapted from [10], we first construct from G another graph F, as follows. For each edge (x, y) in G, the graph F contains a copy of the "gadget" graph C (see Fig. 2), containing the vertices x and y, the (fixed) vertices s_1, s_2, s_3, and four other vertices which are distinct in each different copy of C. Note that F contains a total of $|V| + 3 + 4|E|$ vertices and $18|E|$ edges.

Given a graph F as above, we construct an instance $\mathcal{I}_F$ of MAX CSP($\mathcal{F}$), as follows. The variables are the vertices of F. For every edge $e = (u, v)$ in F, introduce the constraint $f_{eq}(u, v)$ with weight 1 if e is unmarked in its copy of C (see Fig. 2) and with weight 4 otherwise. Hence, the equality constraints corresponding to each single copy of C have total weight 54, and the total weight of all equality constraints in $\mathcal{I}_F$ is $54|E|$. Assume without loss of generality that $1, 2, 3 \in D$. For each vertex s_i, $i = 1, 2, 3$, introduce the constraint $c_i(s_i)$ with weight $60|E|$.

It is clear that $\mathcal{I}_F$ can be constructed from G in polynomial time, so it only remains to show that this construction can be extended to an L-reduction.

Since the weight of each unary constraint in $\mathcal{I}_F$ is greater than the combined weight of all the binary equality constraints, it follows that in any optimal solution to $\mathcal{I}_F$ each variable s_i must take the value i.

Now consider a subproblem of $\mathcal{I}_F$ corresponding to a single copy of the gadget graph C, and assume that each variable s_i takes the value i. Lemma 4.1 [10] states the following: if the variables x and y take distinct values from the set $\{1, 2\}$, then the optimal solution breaks equality constraints with total weight 27 (either the vertical constraints or the horizontal constraints in Figure 2), and hence satisfies all other equality constraints, with total weight 27 as well. Similarly, if the variables x and y take equal values from the set $\{1, 2\}$ then the optimal solution breaks equality constraints with total weight 28, and hence satisfies all other equality constraints, with total weight 26. Furthermore, if either of the variables x or y takes values outside of the set $\{1, 2\}$, then it is possible to satisfy equality constraints with total weight at most 26.

It follows that $\mathcal{I}_F$ has an optimal solution which assigns the values 1 or 2 to all variables corresponding to vertices of G, and satisfies constraints with total weight $180|E| + 26|E| + M$, where M is the number of pairs of variables corresponding to adjacent vertices of G that are assigned distinct values. Note that M is equal to the size of a maximal cut of the graph G.

Since, as is well known, any maximal cut of G contains at most $|E|$ and at least $|E|/2$ edges (see, e.g., Theorem 2.14 [1]), we have shown that $OPT(\mathcal{I}_F)/M \leq 207|E|/\frac{1}{2}|E|$, and hence our construction satisfies property (a) of an L-reduction, with $\alpha = 414$.

Now let ϕ be any solution to $\mathcal{I}_F$, and define $g(\phi)$ to be the partition of the vertices of G where one class contains all vertices v such that $\phi(v) = 1$, and the other class contains the remaining vertices. Clearly this partition can be obtained from ϕ in polynomial-time.

If ϕ satisfies all three constraints $c_i(s_i)$, then, by the observations above, it satisfies constraints with a total weight of at most $180|E| + 26|E| + N$, where

N is the number of pairs of variables corresponding to adjacent vertices of G that are assigned distinct values from the set $\{1, 2\}$. On the other hand, if it fails to satisfy one of these constraints, then it satisfies constraints with total weight at most $120|E| + 54|E|$. Hence, in either case, g satisfies property (b) of an L-reduction with $\beta = 1$. ∎

Remark 6.5 *In fact, in Theorem 6.4, it is enough to require that $\mathcal{F}$ contains at least three functions of the form c_d.*

7 Conclusion

We believe that the most interesting feature of the research presented in this paper is that it brings together several different methods and directions in combinatorial optimization which have previously been studied separately: Max CSP, submodular functions, and Monge properties. We hope that the ideas and results presented here will stimulate research in all of these areas, and perhaps also impact on other related areas of combinatorial optimization. In particular, the problem of minimizing submodular functions on non-distributive lattices becomes especially important in view of the links we have discovered.

Problem 2 *Is it true that the following problem is tractable for an arbitrary finite lattice $\mathcal{L}$: given a submodular function f on a product lattice $\mathcal{L}^n$, can f be minimized in polynomial time (in n)?*

Earlier analysis of various forms of CSP has shown that the classification of complexity in the Boolean case, when appropriately restated, gave good conjectures about the boundary of tractability for the general case [3–5]. The close connection we have established between tractable cases of Max CSP and the property of supermodularity leads us to conjecture that supermodularity is the only possible reason for tractability in Max CSP. Regardless of whether this conjecture holds, the results we have given above demonstrate that significant progress can now be made in developing efficient algorithms for all the known tractable cases of Max CSP by exploiting the large body of existing results concerning sub- and supermodularity, and Monge properties (e.g., [6,13,30,36]).

One possible direction to extend our results would be a further study of the approximability of constraint satisfaction problems over arbitrary finite domains. For example, the techniques presented here can be further fine-tuned to establish **APX**-completeness for at least some of the remaining **NP**-hard cases of Max CSP. However, to complete the study of approximability properties,

it is likely to be necessary to define appropriate notions of expressiveness for a given set of constraint functions, and this has previously only been developed for the Boolean case [8,9,25].

References

[1] G. Ausiello, P. Creszenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation*. Springer, 1999.

[2] F. Börner, A. Bulatov, P. Jeavons, and A. Krokhin. Quantified constraints: Algorithms and complexity. In *Proceedings of 17th International Workshop on Computer Science Logic, CSL'03*, volume 2803 of *LNCS*, pages 58–70, 2003.

[3] A.A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings 43rd IEEE Symposium on Foundations of Computer Science, FOCS'02*, pages 649–658, 2002.

[4] A.A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings 18th IEEE Symposium on Logic in Computer Science, LICS'03*, pages 321–330, 2003.

[5] A.A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. In *Proceedings 44th IEEE Symposium on Foundations of Computer Science, FOCS'03*, pages 562–572, 2003.

[6] R.E. Burkard, B. Klinz, and R. Rudolf. Perspectives of Monge properties in optimization. *Discrete Applied Mathematics*, 70:95–161, 1996.

[7] 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.

[8] N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *Journal of Computer and System Sciences*, 51:511–522, 1995.

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

[10] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[11] M. Datar, T. Feder, A. Gionis, R. Motwani, and R. Panigrahy. A combinatorial algorithm for MAX CSP. *Information Processing Letters*, 85(6):307–315, 2003.

[12] B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2002.

[13] B.L. Dietrich and A.J. Hoffman. On greedy algorithms, partially ordered sets, and submodular functions. *IBM Journal of Research and Development*, 47(1):25–30, 2003.

[14] L. Engebretsen. The non-approximability of non-Boolean predicates. *SIAM Journal on Discrete Mathematics*, 18(1):114–129, 2004.

[15] L. Engebretsen and V. Guruswami. Is constraint satisfaction over two variables always easy? *Random Structures and Algorithms*, 25(2):150–178, 2004.

[16] P. Favati and F. Tardella. Convexity in nonlinear integer programming. *Ricerca Operativa*, 53:3–44, 1990.

[17] S. Fujishige. *Submodular Functions and Optimization*, volume 47 of *Annals of Discrete Mathematics*. North-Holland, Amsterdam, 1991.

[18] M.X. Goemans and V.S. Ramakrishnan. Minimizing submodular functions over families of subsets. *Combinatorica*, 15:499–513, 1995.

[19] A. Goldberg and R.E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988.

[20] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, New York, 1988.

[21] J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48:798–859, 2001.

[22] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM*, 48(4):761–777, 2001.

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

[24] P. Jonsson. Boolean constraint satisfaction: Complexity results for optimization problems with arbitrary weights. *Theoretical Computer Science*, 244(1-2):189–203, 2000.

[25] S. Khanna, M. Sudan, L. Trevisan, and D. Williamson. The approximability of constraint satisfaction problems. *SIAM Journal on Computing*, 30(6):1863–1920, 2001.

[26] H. Narayanan. *Submodular Functions and Electrical Networks*. North-Holland, 1997.

[27] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.

[28] C.H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[29] C.H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43(3):425–440, 1991.

[30] R. Rudolf. Recognition of d-dimensional Monge arrays. *Discrete Applied Mathematics*, 52(1):71–82, 1994.

[31] T.J. Schaefer. The Complexity of Satisfiability Problems. In *Proceedings 10th ACM Symposium on Theory of Computing, STOC'78*, pages 216–226, 1978.

[32] A. Schrijver. A combinatorial algorithm minimizing submodular functions in polynomial time. *Journal of Combinatorial Theory, Ser.B*, 80:346–355, 2000.

[33] M. Serna, L. Trevisan, and F. Xhafa. The (parallel) approximability of non-boolean satisfiability problems and restricted integer programming. In *Proceedings STACS'98*, volume 1373 of *LNCS*, pages 488–498, 1998.

[34] A. Shioura. Minimization of an M-convex function. *Discrete Applied Mathematics*, 84:215–220, 1998.

[35] D. Topkis. Minimizing a submodular function on a lattice. *Operations Research*, 26(2):305–321, 1978.

[36] D. Topkis. *Supermodularity and Complementarity*. Princeton University Press, 1998.

[37] U. Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proceedings 9th ACM Symposium on Discrete Algorithms, SODA'98*, pages 201–210, 1998.