Programming Research Group

SUPERMODULAR FUNCTIONS AND THE COMPLEXITY OF MAX CSP

David Cohen
Department of Computer Science
Royal Holloway, University of London, UK

Martin Cooper
IRIT
University of Toulouse III, 31062 Toulouse, France

Peter Jeavons
OUCL
University of Oxford, Oxford, UK

Andrei Krokhin
Department of Computer Science
University of Warwick, UK

PRG-RR-04-01

Oxford University Computing Laboratory
Wolfson Building, Parks Road, Oxford OX1 3QD

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. We describe for the first time a general approach to the question of classifying the complexity of Max CSP for different types of constraints. This approach is based on establishing a novel connection with the theory of sub- and supermodular functions on finite lattice-ordered sets. Using this connection, we are able to identify large classes of efficiently solvable subproblems of Max CSP arising from certain restrictions on the constraint types. All previously known complexity classification results for Max CSP were restricted to constraints over a 2-valued (Boolean) domain. Here we show that these previous results can be restated using supermodularity, and from this 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.

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 [10].

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., [7, 8, 21, 24, 25, 36]). In the non-Boolean case, a number of results have been obtained that concern approximation properties (see, e.g., [10, 13, 14, 32]). 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 [7, 8, 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, 3, 4, 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 [30] 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 [35]). 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 the surprising result 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 *consists of 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 [8, Lemma 7.2]).

Throughout the paper, $\mathcal{F}$ will denote a *finite* subset of R_D, 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. We will also make use of some standard optimization and approximation complexity classes. The logically defined optimization complexity class **MAX SNP** introduced in [28] consists of all maximization problems of corresponding to problems from **SNP**, that is, to problems of the form $\exists S \forall \overline{x} \psi(\overline{x}, G, S)$ where ψ is a quantifier-free first-order expression involving the variables in $\overline{x}$, the structure G (which is the input) and S (which is a solution). The approximation complexity class **APX** (which is a mild generalization of **MAX SNP**) 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, 8, 28].

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

Proof. It was shown in [16] that CSP($\mathcal{F}$), the decision problem of whether all constraints in a given instance can be satisfied, belongs to **SNP** (in fact, to **MMSNP** which is a subclass of **SNP**). It then follows from the definition of **MAX SNP** [28], that MAX CSP($\mathcal{F}$) belongs to **MAX SNP**. Finally, by [28, Theorem 1], **MAX SNP** is a subclass of **APX**. ∎

A complete classification of the complexity of MAX CSP($\mathcal{F}$) for a two-element set D was obtained in [7, 8, 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 ([8]) *A function* $f \in R_{\{0,1\}}^{(n)}$ *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 ([7, 8, 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 [34, 35]. 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., [11].)

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, 31]), 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, 31], or on a finite grid (or integer lattice) [15, 33], or on general lattices [34, 35].

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.*
- *If $K \in \mathbb{Z}^+$ is an upper bound for the values taken by f_1, then 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 $\mathbf{a} \sqcap \mathbf{b}$. It follows from this that, for all choices of $\mathbf{a}$ and $\mathbf{b}$, we have $f(\mathbf{a}) + f(\mathbf{b}) \le f(\mathbf{a} \sqcap \mathbf{b}) + f(\mathbf{a} \sqcup \mathbf{b})$.

For the converse, assume that f is not 2-monotone. It is shown in (the proof of) Lemma 5.36 in [8] that this implies that there exists a non-2-monotone binary function h that can be obtained from f by using identification of variables and substitution of 0 and 1. It can be easily checked that all *binary* supermodular functions in $R_{\{0,1\}}$ are 2-monotone, hence h is not supermodular. Since identification of variables and substitution of constants preserve supermodularity, we conclude that f is not supermodular. ∎

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 [11]. It is well-known [11] that A can be chosen so that $|A| \le |D|$. 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, 31] that a submodular function on a finite distributive lattice[1] 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 of Theorem 3.1. 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. Thus, we can apply any of the submodular set function minimization algorithms from [20, 22, 31] to maximize f in polynomial time. Note that the algorithm in [31] has the additional assumption that, if $\mathcal{D}^n$ is represented by subsets of a set A, then we can pre-compute, for any $a \in A$, the minimal set M_a that represents an element of $\mathcal{D}^n$ and contains a. This assumption is easily checked to be satisfied in our setting, since $|A| \leq |D|n$, that is, every element of $\mathcal{D}^n$ is represented by a Boolean tuple of fixed length not exceeding $|D|n$. ∎

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.

[1] referred to in [31] as a *ring family*.

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*

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

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.

Example 4.3 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 following instance $\mathcal{I}$ of Max CSP($\mathcal{F}$)

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

where the constraint functions f_i are defined as follows:

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

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.

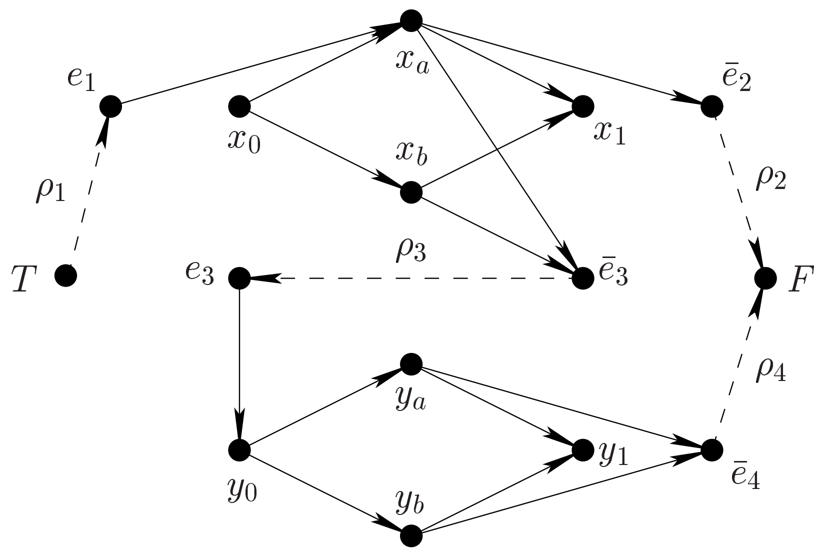

Figure 1: Example of digraph $G_{\mathcal{I}}$. Dashed lines denote constraint arcs, and solid lines denote arcs of weight ∞.

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 4.4 *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 $O(q^3 + n^3|D|^3)$ 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 denote implications about these assertions. Furthermore, given a cut in $G_\mathcal{I}$, vertices x_a from which F can be reached without crossing the cut 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 $a \in \mathcal{L}$ such that $x \sqsubseteq a$ is true. Indeed, assume that there are two such minimal elements, a and a'. Let $c = a \sqcap a'$. Then $x \sqsubseteq c$ is false, 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 $x \sqsubseteq a$ is true. So, we cannot have more than one minimal element $a \in \mathcal{L}$ such that $x \sqsubseteq a$ is true. It remains to notice that $x \sqsubseteq 1_\mathcal{L}$ is always true, since $1_\mathcal{L} \notin B_b$ for any $b \in \mathcal{L}$.

Define an assignment ϕ_C as follows:

$$\phi_C(x) \text{ is the unique minimal element } a \text{ such that } x \sqsubseteq a \text{ is true.}$$

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. Indeed, suppose 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 $x \sqsubseteq a$ is true, since otherwise F would be reachable from T. Similarly, if the constraint arc is of the form $(\bar{e}_i, F)$, then every assertion $x \sqsubseteq a$, where $(x_a, \bar{e}_i)$ is an arc, is false, which implies that the filter part of the constraint f_i is satisfied, since, for any x, if $x \not\sqsubseteq a$ for all $a \in B_b$ then $x \sqsupseteq b$. Finally, suppose that the arc is of the form $(\bar{e}_i, e_i)$. Then, if $x \sqsubseteq a$ is false for some x and a such that there is an arc (e_i, x_a) then every assertion $y \sqsubseteq c$, where c is such that there is an arc $(y_c, \bar{e}_i)$, is also false, which implies that the filter part of the constraint f_i is satisfied. If all such assertions $x \sqsubseteq a$ are true then 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. We examine, in order, the constraint arcs along this path and we obtain a sequence of assertions of the following form:

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

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($\mathcal{F}$). Since this algorithm is cubic in the number of vertices in the graph, the result follows. ∎

Theorem 4.4 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.

This result suggests that it may be worthwhile to look for other forms of constraints for which there exist efficient optimization algorithms. In the next section, we show that when we consider *totally ordered* lattices then such efficient algorithms can be obtained for a wide range of supermodular constraints.

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 special-purpose optimization algorithm. We obtain this result by using a reduction to the MIN CUT problem as in the preceding section, but in this case we can eliminate the dependence of the running time on the number of constraints. The proof uses the fact that binary submodular functions on a chain precisely correspond to square Monge matrices and hence can be decomposed into a sum of simpler functions [6].

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. First we introduce some auxilliary functions we will use in the proof. Consider functions $h^{a,b}$, for each pair of values $a, b \in \mathcal{C}$, defined as follows

$$h^{a,b}(x, y) = \begin{cases} 0 & \text{if } x \sqsubset a \text{ or } y \sqsupset b; \\ 1 & \text{otherwise.} \end{cases}$$

It is easy to check that these functions are submodular on $\mathcal{C}$. Moreover, every unary function $g(x)$ on $\mathcal{C}$ can be expressed as $\sum_{d \in D} (g(d) \cdot h^{d,d}(x, x))$.

Assume that $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{C}}$, and 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($\mathcal{F}$). Then, for every $1 \leq i \leq q$, the function $1 - f_i(\mathbf{x}_i)$ is submodular on $\mathcal{C}$. Lemma 2.3 in [6], when re-stated in terms of submodular functions rather than Monge matrices, implies that $1 - f_i(\mathbf{x}_i)$ can be represented as $\sum_{j=1}^{k_i} g_{ij}(\mathbf{x}_{ij})$ where, for all i, j, there exist a, b such that $g_{ij} = h^{a,b}$. Let $W = \sum_{i=1}^{q} \rho_i$ and $W' = \sum_{i=1}^{q} \rho_i \cdot k_i$. We have

$$f(\mathbf{x}) + (W' - W) = W' - (W - f(\mathbf{x})) =$$

$$= \sum_{i=1}^{q} \rho_i \cdot k_i - \sum_{i=1}^{q} \rho_i \cdot (1 - f_i(\mathbf{x}_i)) = \sum_{i=1}^{q} (\rho_i \cdot (k_i - (1 - f_i(\mathbf{x}_i))) =$$

$$= \sum_{i=1}^{q} \sum_{j=1}^{k_i} \rho_i \cdot (1 - g_{ij}(\mathbf{x}_{ij})).$$

We see that $\mathcal{I}$ is equivalent to another instance $\mathcal{I}'$ of Max CSP($\mathcal{F}$) where all constraint functions are of the form $1 - h^{a,b}$. It is easy to check that these functions are generalized 2-monotone on $\mathcal{C}$, and we therefore can apply Theorem 4.4. It is easy to check that, due to the form of the constraint functions, and the fact that $\mathcal{C}$ is a chain, all constraint arcs in the graph $G_{\mathcal{I}'}$, specified in Definition 4.2, satisfy the following condition: for every vertex of the form e_i or $\bar{e}_i$ (except those identified with T or F), there is exactly one arc coming into this vertex, and exactly one coming out. This implies that we can modify $G_{\mathcal{I}'}$ by removing all vertices corresponding to constraints and replacing all arcs incident to such a vertex by one arc. More precisely,

- replace every pair of arcs of the form (T, e_i), (e_i, x_a) by a single arc (T, x_a) with weight ρ_i;
- replace every pair of arcs of the form $(x_a, \bar{e}_i)$, $(\bar{e}_i, F)$ by a single arc (x_a, F) with weight ρ_i;
- replace every triple of arcs of the form $(x_a, \bar{e}_i)$, $(\bar{e}_i, e_i)$, (e_i, y_b) by a single arc (x_a, y_b) with weight ρ_i.

Since the number of variables in the instances $\mathcal{I}$ and $\mathcal{I}'$ is the same, Theorem 5.2 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 $1 \le i \le k$. For $1 \le i \le k$, 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 $g' \in R_{\{0,1\}}^{(k)}$ 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. 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 $y^i_{j_1}, \ldots , y^i_{j_k}$,
  - constraint $g(y^i_{j_1}, \ldots , y^i_{j_k})$ with weight ρ_i,
  - constraints $\bar{c}_1(y^i_{j_1}), \ldots , \bar{c}_k(y^i_{j_k})$, each with weight W
  - constraints $h_1(x_{j_1}, y^i_{j_1}), \ldots , h_k(x_{j_k}, y^i_{j_k})$, 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 [29] 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*

$$\begin{aligned} f_{eq}(x, y) = 1 \quad &\Leftrightarrow \quad (x = y), \\ c_d(x) = 1 \quad &\Leftrightarrow \quad (x = d). \end{aligned}$$

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.*

[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.

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, 28], which is defined as follows.

Definition 6.3 ([1, 28]) *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 $|cost(x) - OPT(a)| \leq \beta \cdot |cost(y) - OPT(b)|$.

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. We will give an L-reduction from the MINIMUM 3-TERMINAL CUT problem [9] which is known to be **APX**-complete even when restricted to certain instances described below (see [1, 9]). In this problem, one is given a weighted graph with three designated vertices (terminals), and the goal is to find a minimum weight set of edges whose removal disconnects each terminal from all others.

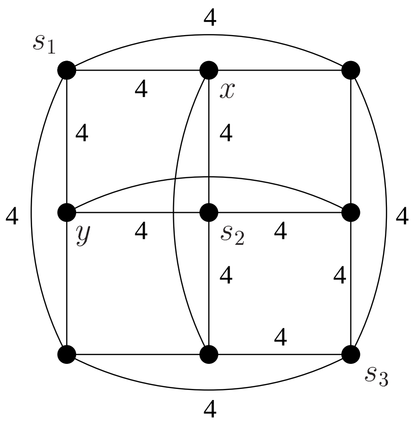

Figure 2: "Gadget" graph C from [9].

Let $G = (V, E)$ be a graph. Following [9], construct another graph F as follows. Let F contain all vertices from V. Introduce three new vertices s_1, s_2, s_3 (the terminals). For each edge (x, y) in G, the graph F contains a copy of the "gadget" graph C (see Fig. 2), with vertices s_1, s_2, s_3 identified with the corresponding terminals, and vertices x and y identified with the corresponding vertices from V. All other vertices will be distinct in different copies of C. Note that the graph C contains 12 edges of weight 4 and six (unmarked) edges of weight 1. It was shown in [9] that Minimum 3-terminal Cut restricted to graphs such as the graph F just described is **APX**-complete (in fact, it was shown to admit an L-reduction from an (**APX**-complete) problem Max Cut, but since Minimum 3-terminal Cut is in **APX** [9], it is **APX**-complete, by [1, Lemma 8.2]).

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 $e = (u, v)$ of weight $\rho_e \in \{1, 4\}$ in F, introduce constraint $f_{eq}(u, v)$ with weight ρ_e. Assume without loss of generality that $1, 2, 3 \in D$. For every terminal vertex s_i, $i = 1, 2, 3$, introduce constraint $c_i(s_i)$ with weight $60|E|$ (which is greater than $52|E|$, the total weight of all edges in F), that is, we oblige every vertex s_i to take value i in all optimal solutions. So the total weight of constraints in $\mathcal{I}$ is $232|E|$. This completes the construction. Obviously, $\mathcal{I}_F$ can be built from F in polynomial time.

It is easy to see that the optimal solutions of F and $\mathcal{I}_F$ precisely correspond to each other in the following way: a minimal 3-terminal cut in F corresponds to an optimal solution to $\mathcal{I}_F$ in which all constraints but those corresponding to edges in the cut are satisfied. It was shown in (the proof of) Theorem 5 of [9] that the optimal solution for F has weight ρ such that $27|E| \leq \rho \leq 28|E|$. Then the optimal solution to $\mathcal{I}_F$ satisfies constraints with total weight $232|E| - \rho$. Then $OPT(\mathcal{I}_F) \leq 205|E|$ and $OPT(F) \geq 27|E|$. It follows that $OPT(\mathcal{I}_F) \leq \frac{205}{27} OPT(F)$, that is property (a) of an L-reduction is satisfied with $\alpha = \frac{205}{27}$. For property (b), note that for every solution ϕ, of weight $232|E| - \rho'$, to $\mathcal{I}_F$ which satisfies all three constraints $c_i(s_i)$, there is a solution E' to F (consisting of all edges with the corresponding constraints $f_{eq}(u, v)$ not satisfied by ϕ) of weight ρ'. If a solution ϕ violates one of the constraints $c_i(s_i)$, then let the corresponding solution E' to F contain all edges in F. It is easy to check that then property (b) is satisfied with $\beta = 1$.

It remains to notice that, by Proposition 2.3, Max CSP($\mathcal{F}$) is in **APX**, and Lemma 8.2 [1] says that in this case an L-reduction guarantees **APX**-completeness. ∎

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, 4, 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, 12, 29, 35]).

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 [7, 8, 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] N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *Journal of Computer and System Sciences*, 51:511–522, 1995.

[8] 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.

[9] 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.

[10] 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.

[11] B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990.

[12] 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.

[13] L. Engebretsen. The non-approximability of non-Boolean predicates. In *Proceedings of RANDOM-APPROX 2001*, volume 2129 of *LNCS*, pages 241–248, 2001.

[14] L. Engebretsen and V. Guruswami. Is constraint satisfaction over two variables always easy? In *Proceedings of RANDOM 2002*, volume 2483 of *LNCS*, pages 224–238, 2002.

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

[16] T. Feder and M.Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal on Computing*, 28:57–104, 1998.

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

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

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

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

[32] 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.

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

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

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

[36] 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.