Identifying Efficiently Solvable Cases of Max CSP

David Cohen[1], Martin Cooper[2], Peter Jeavons[3], and Andrei Krokhin[4]

[1] Department of Computer Science, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, UK
d.cohen@rhul.ac.uk
[2] IRIT, University of Toulouse III, 31062 Toulouse, France
cooper@irit.fr
[3] Computing Laboratory, University of Oxford, Oxford OX1 3QD, UK
peter.jeavons@comlab.ox.ac.uk
[4] Department of Computer Science, University of Warwick, Coventry, CV4 7AL, UK
ak@dcs.warwick.ac.uk

Abstract. In this paper we study the complexity of the maximum constraint satisfaction problem (Max CSP) over an arbitrary finite domain. We describe a novel connection between this problem and the supermodular function maximization problem (which is dual to the submodular function minimization problem). 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. Until now, the only known polynomial-time solvable cases for this form of optimization problem were restricted to constraints over a 2-valued (Boolean) domain. Here we obtain the first examples of general families of efficiently solvable cases of Max CSP for arbitrary finite domains, by considering supermodular functions on finite lattices. 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 k-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 [9].

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, containing both complexity classifications and approximation properties (see, e.g., [7, 17, 20]). In the non-Boolean case, a number of results have been obtained that concern approximation properties (see, e.g., [9, 11]). However, there has so far been very little study of efficient exact algorithms and complexity for subproblems of non-Boolean Max CSP, and the present paper 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, 20]. 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, 19] 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 [7]; such techniques do not readily extend to larger domains. For example, Schaefer [25] 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 [19]. 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 a variety of research areas [13, 16, 18, 21, 22]. The dual problem of maximizing a supermodular function has found interesting applications in diverse economic models, such as supermodular games (see [28]).

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 an even more efficient algorithm for all *binary* supermodular constraints on a totally ordered domain and then shows that the only tractability-preserving way of extending this set of constraints is with further supermodular functions; all other extensions give rise to hard problems. 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 light of the results obtained, and describe possible future work. Proofs of all results are omitted due to space constraints.

2 Preliminaries

Throughout the paper, let D denote a finite set, $|D| > 1$. 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 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. *An instance of* Max CSP *is a 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

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

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

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

In fact, the two versions of MAX CSP can be shown to be equivalent (as in [7, 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. Identify efficiently solvable problems of the form MAX CSP($\mathcal{F}$).

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 well-known 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, 7, 23].

Proposition 1. MAX CSP($\mathcal{F}$) *belongs to* **APX** *for every* $\mathcal{F}$.

A complete classification of the complexity of MAX CSP($\mathcal{F}$) for a two-element set D can be found in [7]. Before stating that result we need to give some definitions.

Definition 3. *An* endomorphism *of $\mathcal{F}$ is a unary operation π on D such that $f(a_1, \dots, a_m) = 1 \Rightarrow f(\pi(a_1), \dots, \pi(a_m)) = 1$ for all $(a_1, \dots, a_m) \in D^m$ and all $f \in \mathcal{F}$. 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 $f(a, , \dots, a) = 1$ 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 4 ([7]). *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 1 ([7]). *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 [28]. 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. (For more information about lattices, see, e.g., [10].)

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 5. *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 5 where $|D| = 2$. Recall that a function on subsets of a set is submodular if $f(X \cap Y) + f(X \cup Y) \leq f(X) + f(Y)$ for all subsets X, Y, and it is supermodular if the inverse inequality holds [13, 22]. The problem of minimizing a submodular set function is tractable [16, 18, 26]. Some results have also been obtained that concern minimization of a submodular function defined on a family of subsets [14, 16, 18, 26], or on a finite grid (or integer lattice) [27], or on general lattices [28].

Observation 2 *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 1 can be characterized using supermodularity.

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

Proposition 2 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 1 in the following form.

Corollary 1. *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 [10]. It is well-known [10] that A can be chosen so that $|A| \leq |D|$. 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 represented using at most $|D|$ bits.

It was shown in [18, 26] that a submodular function on a finite distributive lattice[5] 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 [18].

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

[5] referred to in [26] as a *ring family.*

Theorem 3. *Weighted* Max CSP($\mathcal{F}$) *is in* **PO** *whenever* $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{D}}$ *for some distributive lattice* $\mathcal{D}$ *on* D.

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 [18]). 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.

Definition 6. *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 $\mathbf{x} = (x_1, \ldots, x_n)$, *and* $a_{i_1}, \ldots, a_{i_s}, b_{j_1}, \ldots, b_{j_t} \in D$, *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 [15].

To outline 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 6 (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 7. *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*

- $\{T, F\} \cup \{x_d \mid x \in V, d \in D\} \cup \{e_i, \bar{e}_i \mid i = 1, 2, \dots, q\}$.

  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 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 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:

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

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 7 above.

It can be shown that, for any instance $\mathcal{I}$ of weighted Max CSP($\mathcal{F}$), the total weight of constraints that are *not* satisfied by an optimal solution is equal to the weight of a minimum cut in the graph $G_{\mathcal{I}}$. The proof essentially uses the fact that the order $\sqsubseteq$ is a lattice order. Hence, we get the following result.

Theorem 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.*

Theorem 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 submodular function minimization.

This result suggests that it may be worthwhile to look for other forms of constraint 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.

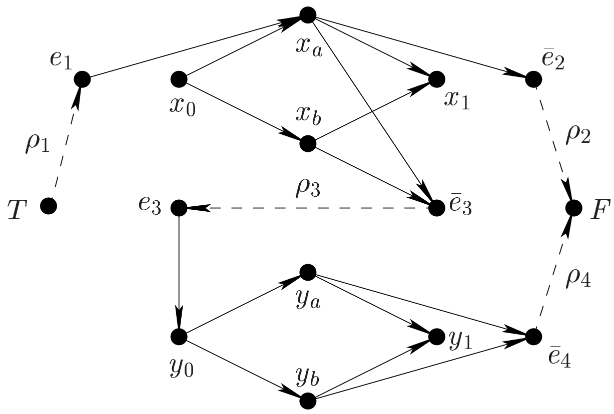

Fig. 1. The digraph $G_{\mathcal{I}}$ corresponding to the MAX CSP instance defined in Example 1. Dashed lines denote constraint arcs, and solid lines denote arcs of weight ∞.

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

It is easy to see that a chain is a distributive lattice, which implies that Theorem 3 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 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. *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.*

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 is with further supermodular functions; all other extensions give rise to hard problems.

Theorem 6. *Let $\mathcal{C}$ be a chain on a finite set D, and let $\mathcal{F} \subseteq R_D$ contain all 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.*

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 [24] that the binary supermodular functions on a finite chain determine the chain (up to reverse order). However, by Lemma 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.

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 8. *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*

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

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

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.

Theorem 7. *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.*

The proof is by reduction from the **APX**-complete Minimum 3-terminal Cut problem [8]. In fact, in Theorem 7, 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.

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, 24, 28]).

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].

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 CSL'03*, volume 2803 of *LNCS*, pages 58–70, 2003.
3. A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings of FOCS'02*, pages 649–658, 2002.
4. A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings of LICS'03*, pages 321–330, 2003.
5. A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. In *Proceedings of FOCS'03*, pages 562-571, 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, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. 2001.
8. 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.

9. 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.
10. B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990.
11. L. Engebretsen and V. Guruswami. Is constraint satisfaction over two variables always easy? In *Proceedings of RANDOM'02*, volume 2483 of *LNCS*, pages 224–238, 2002.
12. 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.
13. S. Fujishige. *Submodular Functions and Optimization*, volume 47 of *Annals of Discrete Mathematics*. North-Holland, Amsterdam, 1991.
14. M.X. Goemans and V.S. Ramakrishnan. Minimizing submodular functions over families of subsets. *Combinatorica*, 15:499–513, 1995.
15. A. Goldberg and R.E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35:921–940, 1988.
16. M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, New York, 1988.
17. J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48:798–859, 2001.
18. 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.
19. P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.
20. P. Jonsson. Boolean constraint satisfaction: Complexity results for optimization problems with arbitrary weights. *Theoretical Computer Science*, 244(1-2):189–203, 2000.
21. H. Narayanan. *Submodular Functions and Electrical Networks*. North-Holland, 1997.
22. G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.
23. C.H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43(3):425–440, 1991.
24. R. Rudolf. Recognition of d-dimensional Monge arrays. *Discrete Applied Mathematics*, 52(1):71–82, 1994.
25. T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings STOC'78*, pages 216–226, 1978.
26. A. Schrijver. A combinatorial algorithm minimizing submodular functions in polynomial time. *Journal of Combinatorial Theory, Ser.B*, 80:346–355, 2000.
27. A. Shioura. Minimization of an M-convex function. *Discrete Applied Mathematics*, 84:215–220, 1998.
28. D. Topkis. *Supermodularity and Complementarity*. Princeton University Press, 1998.