Maximum Constraint Satisfaction on Diamonds

Andrei Krokhin[1] and Benoit Larose[2]

[1] Department of Computer Science
University of Durham
Durham, DH1 3LE, UK
andrei.krokhin@durham.ac.uk

[2] Department of Mathematics and Statistics
Concordia University
Montréal, Qc, Canada, H3G 1M8
larose@mathstat.concordia.ca

Abstract. In this paper we study the complexity of the (weighted) maximum constraint satisfaction problem (Max CSP) over an arbitrary finite domain. In this problem, one is given a collection of weighted constraints on overlapping sets of variables, and the goal is to find an assignment of values to the variables so as to maximize the total weight of satisfied constraints. Max Cut is a typical example of a Max CSP problem. Max CSP is **NP**-hard in general; however, some restrictions on the form of constraints may ensure tractability. Recent results indicate that there is a connection between tractability of such restricted problems and supermodularity of the allowed constraint types with respect to some lattice ordering of the domain. We prove several results confirming this. Diamonds are the smallest lattices in terms of the number of comparabilities, and so are as unordered as a lattice can possibly be. In the present paper, we study Max CSP on diamond-ordered domains. We show that if all allowed constraints are supermodular with respect to such an ordering then the problem can be solved in polynomial (in fact, in cubic) time. We also prove a partial converse: if the set of allowed constraints includes a certain small family of binary supermodular constraints on such a lattice, then the problem is tractable if and only if all of the allowed constraints are supermodular; otherwise, it is **NP**-hard.

Keywords: maximum constraint satisfaction problem, complexity, algorithms, supermodularity, lattice order

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 classical 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 [11].

The Max-CSP framework has been well-studied in the Boolean case, that is, when the set of values for the variables is $\{0, 1\}$. Many fundamental results have been obtained, containing both complexity classifications and approximation properties (see, e.g., [10, 18, 20]). In the non-Boolean case, a number of results have been obtained that concern approximation properties (see, e.g., [11, 14]). However, the study of efficient exact algorithms and complexity for subproblems of non-Boolean Max CSP has started only very recently [8, 21], and the present paper is a contribution towards this line of research.

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. The most well-known examples of such problems are Max k-Sat and Max Cut. In particular, we investigate which restrictions make such problems *tractable*, by allowing a polynomial time algorithm to find an optimal assignment. This setting (in several variations) has been extensively studied and completely classified in the Boolean case [6, 10, 20]. In contrast, we consider here the case where the set of possible values is an *arbitrary finite* set. Let us now formally define these problems.

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 $R_D = \bigcup_{m=1}^{\infty} R_D^{(m)}$. Also, let $\mathbb{Z}^+$ denote the set of all non-negative integers.

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 predicate*; and*
– $\mathbf{x} = (x_{i_1}, \ldots, x_{i_m})$ *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. *For a finite* $\mathcal{F} \subseteq R_D$*, an instance of* Max CSP($\mathcal{F}$) *is a pair* (V, C) *where*

– $V = \{x_1, \ldots, x_n\}$ *is a set of variables taking their values from the set D;*
– C *is a collection of constraints* $f_1(\mathbf{x}_1), \ldots, f_q(\mathbf{x}_q)$ *over* V*, where* $f_i \in \mathcal{F}$ *for all* $1 \leq i \leq q$*; each constraint* $f_i(\mathbf{x}_i)$ *is assigned a weight* $\varrho_i \in \mathbb{Z}^+$.

The goal is to find an assignment $\phi : V \to D$ *that maximizes the total weight of satisfied constraints, that is, to maximize the function* $f : D^n \to \mathbb{Z}^+$*, defined by* $f(x_1, \ldots, x_n) = \sum_{i=1}^{q} \varrho_i \cdot f_i(\mathbf{x}_i)$.

Note that throughout the paper the values 0 and 1 taken by any predicate will be considered, rather unusually, as integers, not as Boolean values, and addition will always denote the addition of integers. Throughout the paper, we assume that $\mathcal{F}$ is finite.

Example 1. The MAX CUT problem is the problem of partitioning the set of vertices of a given undirected graph with weighted edges into two subsets so as to maximize the total weight of edges with ends being in different subsets. Let neq_2 be the binary predicate on $\{0,1\}$ such that $neq_2(x,y) = 1 \Leftrightarrow x \neq y$. Then MAX CUT problem is the same as MAX CSP($\{neq_2\}$). To see this, think of vertices of a given graph as of variables, and apply the predicate to every pair of variables x, y such that (x, y) is an edge in the graph. Let f_{dicut} be the binary predicate on $\{0,1\}$ such that $f_{dicut}(x,y) = 1 \Leftrightarrow x = 0, y = 1$. Then MAX CSP($\{f_{dicut}\}$) is essentially the problem MAX DICUT which is the problem of partitioning the vertices of a digraph with weighted arcs into two subsets V_0 and V_1 so as to maximize the total weight of arcs going from V_0 to V_1. It is well known that both MAX CUT and MAX DICUT are **NP**-hard.

The main research problem we will study in this paper is the following:

Problem 1. What are the sets $\mathcal{F}$ such that MAX CSP($\mathcal{F}$) is tractable?

For the Boolean case, Problem 1 was solved in [9]. It appears that Boolean problems MAX CSP($\mathcal{F}$) exhibit a dichotomy in that such a problem either is solvable exactly in polynomial time or else is **NP**-hard (which cannot be taken for granted because of Ladner's theorem). The paper [9] also describes the boundary between the two cases.

Versions of Problem 1 for other non-Boolean constraint problems (including decision, quantified, and counting problems) have been actively studied in the last years, with many classification results obtained (see, e.g., [1–4, 17]). Experience in the study of various forms of constraint satisfaction (see, e.g., [2, 3]) 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 algebraic combinatorial property of supermodularity (see Section 2) is a well-known source of tractable maximization problems [5, 15]. In combinatorial optimization, this property (or the dual property of submodularity) is usually considered on subsets of a set or on distributive lattices [5, 19]. However, it can be considered on arbitrary lattices, and this has proved useful in operational research [22]. Very recently [8], this general form of supermodularity was proposed as the main tool in tackling Problem 1. Indeed, for $|D| = 2$, this property was shown [8] to completely characterize the tractable cases of MAX CSP (originally, the characterization was obtained [9] in a different form), and moreover, this property also essentially characterizes the tractable cases for $|D| = 3$ [21].

Interestingly, the relevance of an ordering of the domain is not suggested in any way by the formulation of Problem 1. In this paper we further investigate the role of ordering in the study of MAX CSP, and then we determine the complexity of MAX CSP assuming that the domain has a lattice ordering, but the order is a diamond order, that is, it is as loose as a lattice order can possibly be.

The structure of the paper is as follows: in Section 2 we discuss lattices, supermodularity, and their relevance in the study of MAX CSP. In Section 3, we clarify the role that ordering plays in this approach. In Section 4 we describe the structure of supermodular predicates on diamonds, which we use in Section 5 to show that Max CSP with such constraints can be solved in cubic time. In Section 6 we show that a certain small set of supermodular constraints on diamonds gives rise to **NP**-hard problems when extended with any non-supermodular constraint.

2 Preliminaries

In this section we discuss the well-known combinatorial algebraic property of supermodularity [22] which will play a crucial role in classifying the complexity of Max CSP problems.

A partial order on a set D is called a *lattice order* if, for every $x, y \in D$, there exists a greatest lower bound $x \sqcap y$ (called the *meet* of a and b) and a least upper bound $x \sqcup y$ (called the *join*). The corresponding algebra $\mathcal{L} = (D, \sqcap, \sqcup)$ is called a *lattice*. It is well-known that any finite lattice $\mathcal{L}$ has a greatest element $1_{\mathcal{L}}$ and a least element $0_{\mathcal{L}}$. A lattice is called *distributive* (also known as a *ring family*) if it can be represented by subsets of a set, the operations being set-theoretic intersection and union. An n-*diamond* (or simply a diamond), denoted $\mathcal{M}_n$, is a lattice on an $(n+2)$-element set such that all n elements in $\mathcal{M}_n \setminus \{0_{\mathcal{M}_n}, 1_{\mathcal{M}_n}\}$ are pairwise incomparable. The Hasse diagram of $\mathcal{M}_n$ is given in Fig. 1. Since

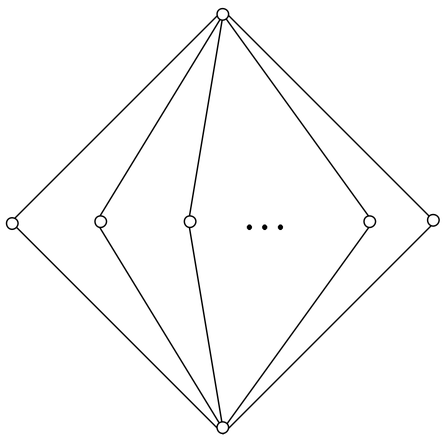

Fig. 1. A diamond lattice $\mathcal{M}_n$.

every element of any lattice must be comparable with both the top and the bottom elements, diamonds are as unordered as lattices can possibly be. The middle elements of $\mathcal{M}_n$ are called *atoms*. Note that, for every distinct atoms a and b, we have $a \sqcap b = 0_{\mathcal{M}_n}$ and $a \sqcup b = 1_{\mathcal{M}_n}$. It is well-known (and easy to see) that a distributive lattice cannot contain $\mathcal{M}_3$ (and hence any $\mathcal{M}_n$ with $n \geq 3$) as a sublattice. In the literature (e.g., [12]), the lattice $\mathcal{M}_3$ is often called *the* diamond. For more information on lattices and orders, see [12].

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

Definition 3. *Let $\mathcal{L}$ be a lattice on D. A function $f : D^n \to \mathbb{R}$ is called* supermodular *on $\mathcal{L}$ if it satisfies*

$$f(\mathbf{a}) + f(\mathbf{b}) \leq f(\mathbf{a} \sqcap \mathbf{b}) + f(\mathbf{a} \sqcup \mathbf{b}) \tag{1}$$

for all $\mathbf{a}, \mathbf{b} \in D^n$, and f is called submodular *on $\mathcal{L}$ if the inverse inequality holds. The set of all supermodular predicates on $\mathcal{L}$ will be denoted* $\mathrm{Spmod}_{\mathcal{L}}$.

It is easy to see that f is supermodular on a lattice if and only if $-f$ is submodular on it. Also, it follows directly from the definition that f is supermodular on $\mathcal{L}$ if and only if it is supermodular on its dual lattice $\mathcal{L}^{\partial}$ (obtained by reversing the order).

The standard definition of sub- and supermodular (set) functions [5, 15] corresponds to the special case of the above definition when $|D| = 2$. Recall that a *chain* is a totally ordered lattice. Sub- and supermodular functions on finite chains have been studied in combinatorial optimization under the name of Monge and inverse Monge matrices and arrays (see survey [5]). Note that chains and diamonds represent "opposite" types of lattices: chains have all possible comparabilities, while diamonds have as few comparabilities as possible.

The following results have been previously obtained in classifying the complexity of Max CSP.

Theorem 1 ([8]). *If $\mathcal{F}$ is supermodular on some distributive lattice on D, then* Max CSP($\mathcal{F}$) *is tractable.*

Call a problem Max CSP($\mathcal{F}$) *trivial* if, for some $a \in D$, $f(a, \ldots, a) = 1$ for all $f \in \mathcal{F}$ (clearly, in this case all constraints in any instance can be satisfied).

Theorem 2 ([8, 10]). *Let $D = \{0, 1\}$, $\mathcal{F} \subseteq R_D$, and assume that* Max CSP($\mathcal{F}$) *is non-trivial. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{C}}$ for some chain $\mathcal{C}$ on D then* Max CSP($\mathcal{F}$) *is tractable. Otherwise,* Max CSP($\mathcal{F}$) *is* **NP***-hard.*

Call a value $d \in D$ *redundant for* $\mathcal{F}$ if, when replacing this value by some other (fixed) value in any assignment ϕ, one would satisfy all constraints satisfied by ϕ. Hence, if d is redundant for $\mathcal{F}$ then it can be ignored and Max CSP($\mathcal{F}$) reduces to a similar problem over domain $D \setminus \{d\}$.

Theorem 3 ([21]). *Let $D = \{0, 1, 2\}$, $\mathcal{F} \subseteq R_D$, and assume that there are no redundant values for $\mathcal{F}$. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{C}}$ for some chain $\mathcal{C}$ on D then* Max CSP($\mathcal{F}$) *is tractable. Otherwise,* Max CSP($\mathcal{F}$) *is* **NP***-hard.*

In the next section we will show that Theorems 2 and 3 (as stated) cannot be generalized to larger domains because non-chain lattices must also play a role.

Proofs of all theorems cited above in this section enjoyed support of known results on classical submodular (set) functions and on Monge matrices and arrays. However, if we are unable to represent the lattice operations by set-theoretic ones then the analysis of supermodular functions becomes significantly more difficult.

We will now consider a form of supermodular constraints that can be defined on any lattice.

Definition 4. *A predicate $f \in R_D^{(n)}$ will be called* 2-monotone[3] *on a poset $\mathcal{P}$ on D if it can be expressed as follows*

$$f(\mathbf{x}) = 1 \Leftrightarrow ((x_{i_1} \sqsubseteq a_{i_1}) \wedge \ldots \wedge (x_{i_s} \sqsubseteq a_{i_s})) \vee ((x_{j_1} \sqsupseteq b_{j_1}) \wedge \ldots \wedge (x_{j_t} \sqsupseteq b_{j_t})) \quad (2)$$

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 straightforward to check every 2-monotone predicate on a lattice is supermodular on it. The next theorem is, to the best of our knowledge, the only one available on the complexity of supermodular constraints on arbitrary lattices.

Theorem 4 ([8]). *Let $\mathcal{L}$ be a lattice on a finite set D. If $\mathcal{F}$ consists of 2-monotone predicates on $\mathcal{L}$, then* Max CSP($\mathcal{F}$) *is tractable.*

3 The role of ordering

In this section we prove results clarifying the role of ordering in the analysis of the complexity of Max CSP and of the structure of supermodular predicates on lattices.

The next proposition shows that Theorem 4 cannot be generalized to non-lattice posets, and, therefore, lattice orders seem to play a special role in the study of Max CSP.

Proposition 1. *Let $\mathcal{P}$ be a non-lattice poset. If $\mathcal{F}$ contains all at most binary 2-monotone constraints on $\mathcal{P}$ then* Max CSP($\mathcal{F}$) *is* **NP**-*hard.*

Proof. Since $\mathcal{P}$ is not a lattice, there are two elements a, b in $\mathcal{P}$ which either do not have a greatest lower bound or do not have a least upper bound. Assume that $a \sqcap b$ does not exist, the other case is dual. Consider first the case when a and b have no common lower bound at all. It is easy to see that, in this case, a and b can be chosen to be minimal elements in $\mathcal{P}$. Then it is easy to see that the binary predicate $(x \sqsubseteq a) \wedge (y \sqsubseteq b)$ is 2-monotone on $\mathcal{P}$ and, moreover, this predicate is satisfied on a unique tuple which is (a, b). Hence this predicate can simulate f_{dicut} (see Example 1), and the problem is **NP**-hard. Let us now consider the

[3] In [8], such predicates are called *generalized* 2-monotone.

case when a and b have a common lower bound. It then follows that they have two maximal common lower bounds c and d. Let $\mathcal{P}'$ be the subposet of $\mathcal{P}$ such that $x \in \mathcal{P}' \Leftrightarrow (x \sqsupseteq c) \wedge (x \sqsupseteq d)$. Clearly, $\mathcal{P}'$ is non-empty, since it contains both a and b. Moreover, $\mathcal{P}'$ is not a lattice, since, by the choice of c, d, the elements a and b have no common lower bound (and, in fact, are minimal) in $\mathcal{P}'$.

Now we reduce Max CSP($\{f_{dicut}\}$) to Max CSP($\mathcal{F}$). Assume that the domain for Max CSP($\{f_{dicut}\}$) is $\{a, b\}$ where a plays the role of 0 and b that of 1. Let $g \in \mathcal{F}$ be such that $g(x, y) = 1 \Leftrightarrow [(x \sqsubseteq a) \wedge (y \sqsubseteq b)]$. Take an arbitrary instance $\mathcal{I}$ of Max CSP($\{f_{dicut}\}$). Let W be the total weight of all constraints in $\mathcal{I}$, plus 1. Modify $\mathcal{I}$ to an instance $\mathcal{I}'$ of Max CSP($\mathcal{F}$) as follows: replace every constraint $f_{dicut}(x, y)$ by $g(x, y)$ with the same weight, and, for every variable x in $\mathcal{I}$, add constraints $x \sqsupseteq c$ and $x \sqsupseteq d$ with weight W each. It is easy to see that any optimal solution to $\mathcal{I}$ is also an optimal solution to $\mathcal{I}'$. It is clear that the large-weight constraints force all values in an optimal solution to $\mathcal{I}'$ to be in $\mathcal{P}'$. Moreover, the choice of g implies that there is always an optimal solution to $\mathcal{I}'$ which uses only values a and b. Clearly, this solution would also be optimal for $\mathcal{I}$. $\square$

The next proposition shows that classes of supermodular predicates on two different lattices are pairwise incomparable, up to duality. As a consequence, if we want to prove that supermodularity on lattices gives rise to tractable Max CSP problems then we cannot exclude *any* lattice from our analysis (except that mutually dual lattices can be identified).

Proposition 2. *Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be finite lattices on the same base set, and $\mathcal{L}_1 \neq \mathcal{L}_2, \mathcal{L}_2^\partial$. Then there exists an at most binary predicate f which is 2-monotone on $\mathcal{L}_1$ but not supermodular on $\mathcal{L}_2$.*

Proof. For $i = 1, 2$, let $\sqsubseteq_i$, $\sqcap_i$ and $\sqcup_i$ denote the order and the operations of $\mathcal{L}_i$, respectively.

Suppose first that there exists a pair of elements $\{a, b\}$ that are incomparable in $\mathcal{L}_1$ but $a < b$ in $\mathcal{L}_2$. Consider the predicate f defined by $f(x, y) = 1$ if and only if $(x, y) \geq_1 (a, b)$ or $(x, y) \leq_1 (b, a)$. Then f is 2-monotone on $\mathcal{L}_1$, and $f(a, a) = f(b, b) = 0$ shows that it is not supermodular on $\mathcal{L}_2$.

Now we consider the case where every pair comparable in $\mathcal{L}_2$ is also comparable in $\mathcal{L}_1$. There are two possibilities: (i) suppose there exist elements a, b, c, d such that $a <_1 b$ and $a <_2 b$, and also $c <_1 d$ and $d <_2 c$. Consider the predicate f defined by $f(x, y) = 1$ if and only if $(x, y) \leq_1 (a, c)$. Obviously, it is 2-monotone on $\mathcal{L}_1$. However, in $\mathcal{L}_2$ we have that $(a, c) \sqcup_2 (b, d) = (b, c)$, $(a, c) \sqcap_2 (b, d) = (a, d)$, and $f(a, d) + f(b, c) = 0$ while $f(a, c) + f(b, d) = 1$ so f is not supermodular on $\mathcal{L}_2$.

(ii) It remains to check the case where $\mathcal{L}_1$ is an extension of $\mathcal{L}_2$ or $\mathcal{L}_2^\partial$. Since obviously a predicate is supermodular (2-monotone) on a lattice if and only if it is supermodular (2-monotone) on its dual, it suffices to consider, without loss of generality, the case where $\mathcal{L}_1$ is an extension of $\mathcal{L}_2$. There exists a pair of elements $u \leq_1 v$ that are incomparable in $\mathcal{L}_2$: then the predicate f defined by $f(x) = 1$ if and only if $x \leq_1 v$ is 2-monotone on $\mathcal{L}_1$ but not on $\mathcal{L}_2$: indeed, if $w = u \sqcup_2 v$ then $v <_2 w$ implies that $v <_1 w$ and hence $f(w) = 0$ while $f(u) + f(v) = 2$. □

Now we are able to show that Theorems 2 and 3 (as stated above) cannot be generalized to larger domains.

Corollary 1. *If $|D| \geq 4$ then there exists $\mathcal{F} \subseteq R_D$ such that* Max CSP($\mathcal{F}$) *is tractable but $\mathcal{F} \not\subseteq \mathrm{Spmod}_{\mathcal{C}}$ for any chain $\mathcal{C}$ on D.*

Proof. Let $\mathcal{L}$ be the following lattice on D: arbitrarily choose the top element, then choose two other elements and make them incomparable, then let all other elements form a chain lying below the first three elements. It is easy to see that, for $|D| \geq 4$, $\mathcal{L}$ is a distributive lattice which is not a chain. By Proposition 2, the set $\mathcal{F}$ of all binary predicates on $\mathcal{L}$ is not supermodular on any chain on D. However Max CSP($\mathcal{F}$) is tractable by Theorem 1. □

4 The structure of supermodular predicates on diamonds

In the rest of this paper we consider supermodular constraints on diamonds $\mathcal{M}_n$ which are non-distributive lattices when $n \geq 3$. Throughout the rest of this paper, let $\mathcal{L}$ be an arbitrary (fixed) n -diamond, $n \geq 2$.

In this section, we describe the structure of supermodular predicates on $\mathcal{L}$ by representing them as logical formulas involving constants (elements of $\mathcal{L}$) and the order relation $\sqsubseteq$ of $\mathcal{L}$.

For a subset $D' \subseteq D$, let $u_{D'}$ denote the predicate such that $u_{D'}(x) = 1 \Leftrightarrow x \in D'$. The following lemma can be easily derived directly from the definition of supermodularity.

Lemma 1. *A unary predicate $u_{D'}$ is in $\mathrm{Spmod}_{\mathcal{L}}$ if and only if either both $0_{\mathcal{L}}, 1_{\mathcal{L}} \in D'$ or else $|D'| \leq 2$ and at least one of $0_{\mathcal{L}}, 1_{\mathcal{L}}$ is in D'.*

Theorem 5. *Every predicate $f(x_1, \ldots, x_n)$ in $\mathrm{Spmod}_{\mathcal{L}}$, such that f takes both values 0 and 1, can be represented as one of the following logical implications:*

1. $[(x_i \sqsubseteq a_1) \vee \ldots \vee (x_i \sqsubseteq a_l)] \implies (x_i \sqsubseteq 0_{\mathcal{L}})$ *where the a_j's are atoms;*
2. $\neg(\mathbf{y} \sqsupseteq \mathbf{c}) \implies (\mathbf{z} \sqsubseteq \mathbf{d})$ *where $\mathbf{y}$ and $\mathbf{z}$ are some subsequences of $(x_1, \ldots, x_n)$, and $\mathbf{c}$, $\mathbf{d}$ are tuples of elements of $\mathcal{L}$ (of corresponding length) such that $\mathbf{c}$ contains no $0_{\mathcal{L}}$ and $\mathbf{d}$ no $1_{\mathcal{L}}$;*
3. $[(x_i \sqsubseteq b_1) \vee \cdots \vee (x_i \sqsubseteq b_k) \vee \neg(\mathbf{y} \sqsupseteq \mathbf{c})] \implies (x_i \sqsubseteq a)$ *where the b_j's are atoms, $\mathbf{y}$ does not contain x_i, and $a \neq 1_{\mathcal{L}}$;*
4. $\neg(x_i \sqsupseteq b) \implies [\neg(x_i \sqsupseteq a_1) \wedge \cdots \wedge \neg(x_i \sqsupseteq a_l) \wedge (\mathbf{y} \sqsubseteq \mathbf{c})]$ *where the a_j's are atoms, $\mathbf{y}$ does not contain x_i, and $b \neq 0_{\mathcal{L}}$;*
5. $\neg(\mathbf{y} \sqsupseteq \mathbf{c}) \implies$ ***false*** *where $\mathbf{y}$ is a subsequence of $(x_1, \ldots, x_n)$ and $\mathbf{c}$ contains no $0_{\mathcal{L}}$;*
6. ***true*** $\implies (\mathbf{y} \sqsubseteq \mathbf{c})$ *where $\mathbf{y}$ is a subsequence of $(x_1, \ldots, x_n)$ and $\mathbf{c}$ contains no $1_{\mathcal{L}}$.*

Conversely, every predicate that can be represented in one of the above forms belongs to $\mathrm{Spmod}_{\mathcal{L}}$.

Example 2. The unary predicate of type (1) above is the same as $u_{D'}$ where $D' = D \setminus \{a_1, \ldots, a_l\}$. The predicates $u_{D'} \in \mathrm{Spmod}_{\mathcal{L}}$ with $|D'| \leq 2$ are the unary predicates of types (5) and (6).

Remark 1. Note that constraints of types (2),(5), and (6) are 2-monotone on $\mathcal{L}$, while constraints of types (3) and (4) (and most of those of type (1)) are not.

Proof. It is straightforward to verify that all the predicates in the list are actually supermodular. Now we prove the converse. Consider first the case where the predicate f is essentially unary, i.e., there is a variable x_i such that $f(x_1, \ldots, x_n) = u_{D'}(x_i)$ for some $D' \subsetneq D$. If $D' = \{x : x \sqsubseteq a\}$ or $D' = \{x : x \sqsupseteq a\}$ for some atom a then f is of the form (5) or (6); otherwise both $0_{\mathcal{L}}$ and $1_{\mathcal{L}}$ are in D' by Lemma 1, and if $a_1, \ldots, a_l$ denote the atoms of the lattice that are not in D', then it is clear that f is described by the implication (1).

Now we may assume that f is not essentially unary. If it is 2-monotone, then it is easy to see that f must be described by an implication of type (2), (5) or (6). So now we assume that f is not essentially unary and it is not 2-monotone; we prove that it is described by an implication of type (3) or (4). We require a few claims:

Claim 0. *The set* X *of all tuples* $\mathbf{u}$ *such that* $f(\mathbf{u}) = 1$ *is a sublattice of* $\mathcal{L}^n$*, i.e. is closed under join and meet.*

This follows immediately from the supermodularity of f.

Claim 1. *There exist indices* $1 \leq i_1, \ldots, i_k, j_1, \ldots, j_l \leq n$*, atoms* $e_1, \ldots, e_k$ *and* $b_1, \ldots, b_l$ *of* $\mathcal{L}$ *such that* $f(\mathbf{x}) = 1$ *if and only if*

$$[\neg(x_{i_1} \sqsupseteq e_1) \wedge \cdots \wedge \neg(x_{i_k} \sqsupseteq e_k)] \bigvee [\neg(x_{j_1} \sqsubseteq b_1) \wedge \cdots \wedge \neg(x_{j_l} \sqsubseteq b_l)].$$

Notice first that the set Z of tuples $\mathbf{u}$ such that $f(\mathbf{u}) = 0$ is *convex* in $\mathcal{L}^n$, i.e. if $\mathbf{u} \sqsubseteq \mathbf{v} \sqsubseteq \mathbf{w}$ with $f(\mathbf{u}) = f(\mathbf{w}) = 0$ then $f(\mathbf{v}) = 0$. To show this we construct a tuple $\mathbf{v}'$ as follows: for each coordinate i it is easy to find an element v'_i such that $v_i \sqcap v'_i = u_i$ and $v_i \sqcup v'_i = w_i$. Hence $\mathbf{v} \sqcap \mathbf{v}' = \mathbf{u}$ and $\mathbf{v} \sqcup \mathbf{v}' = \mathbf{w}$ so by supermodularity of f neither $\mathbf{v}$ nor $\mathbf{v}'$ is in X. It follows in particular that neither $0_{\mathcal{L}^n}$ nor $1_{\mathcal{L}^n}$ is in Z; indeed, if $0_{\mathcal{L}^n} \in Z$, let $\mathbf{a}$ be the smallest element in X (the meet of all elements in X), which exists by Claim 0. Since Z is convex it follows that every element above $\mathbf{a}$ is in X so f is 2-monotone, a contradiction. The argument for $1_{\mathcal{L}^n}$ is identical.

Now let $\mathbf{w} \in Z$ be minimal, and let $\mathbf{v} \sqsubseteq \mathbf{w}$. As above we can find a tuple $\mathbf{v}'$ such that $\mathbf{v} \sqcup \mathbf{v}' = \mathbf{w}$; by supermodularity of f it follows that $\mathbf{v} = \mathbf{w}$ or $\mathbf{v}' = \mathbf{w}$. It is easy to deduce from this that there exists a coordinate s such that w_s is an atom of $\mathcal{L}$ and $w_t = 0_{\mathcal{L}}$ for all $t \neq s$. A similar argument shows that every maximal element of Z has a unique coordinate which is an atom and all others are equal to $1_{\mathcal{L}}$.

Since Z is convex, we have that $f(\mathbf{x}) = 0$ if and only if $\mathbf{x}$ is above some minimal element of Z and below some maximal element of Z; Claim 1 then follows immediately.

For each index $i \in \{i_1, \ldots, i_k\}$ that appears in the expression in Claim 1, there is a corresponding condition of the form

$$\neg(x_i \sqsupseteq e_{s_1}) \wedge \cdots \wedge \neg(x_i \sqsupseteq e_{s_r});$$

let I_i denote the set of elements of $\mathcal{L}$ that satisfy this condition. Obviously it cannot contain $1_{\mathcal{L}}$ and must contain $0_{\mathcal{L}}$. Similarly, define for each index $j \in \{j_1, \ldots, j_l\}$ the set F_j of all elements of $\mathcal{L}$ that satisfy the corresponding condition of the form

$$\neg(x_j \sqsubseteq b_{t_1}) \wedge \cdots \wedge \neg(x_j \sqsubseteq b_{t_q});$$

it is clear that $0_{\mathcal{L}} \notin F_j$ and $1_{\mathcal{L}} \in F_j$.

The condition of Claim 1 can now be rephrased as follows: $f(\mathbf{x}) = 1$ *if and only if* $x_i \in I_i$ *for all* $i \in \{i_1, \ldots, i_k\}$ *or* $x_j \in F_j$ *for all* $j \in \{j_1, \ldots, j_l\}$. It is straightforward to verify that since f is not 2-monotone, one of the I_i or one of the F_j must contain 2 distinct atoms. We consider the first case, and we show that the predicate f is of type (4). The case where some F_j contains two atoms is dual and will yield type (3).

Claim 2. *Suppose that I_i contains distinct atoms c and d for some $i \in \{i_1, \ldots, i_k\}$. Then (a) i is the only index with this property, (b) $\{j_1, \ldots, j_l\} = \{i\}$ and (c) F_i does not contain 2 distinct atoms.*

We prove (b) first. We have that

$$f(0_{\mathcal{L}}, \ldots, 0_{\mathcal{L}}, c, 0_{\mathcal{L}}, \ldots, 0_{\mathcal{L}}) = f(0_{\mathcal{L}}, \ldots, 0_{\mathcal{L}}, d, 0_{\mathcal{L}}, \ldots, 0_{\mathcal{L}}) = 1$$

(where c and d appear in the i-th position) and by supermodularity it follows that $f(0_{\mathcal{L}}, \ldots, 0_{\mathcal{L}}, 1_{\mathcal{L}}, 0_{\mathcal{L}}, \ldots, 0_{\mathcal{L}}) = 1$ also. Since I_i does not contain $1_{\mathcal{L}}$, we have that $x_j \in F_j$ for each $j \in \{j_1, \ldots, j_l\}$; since F_j never contains $0_{\mathcal{L}}$, (b) follows immediately. Since $\{j_1, \ldots, j_l\}$ is non-empty, (a) follows immediately from (b). Finally, if F_i contained distinct atoms then by dualising the preceding argument we would obtain that $\{i_1, \ldots, i_k\} = \{i\}$ from which it would follow that f would be essentially unary, contrary to our assumption. This concludes the proof of the claim.

Let b denote the minimal element in F_i, and for each index $s \in \{i_1, \ldots, i_k\}$ different from i let c_s denote the (unique) maximal element of I_s; then we can describe f as follows: $f(\mathbf{x}) = 1$ if and only if

$$[x_i \in I_i \wedge (\mathbf{y} \sqsubseteq \mathbf{c})] \vee (x_i \sqsupseteq b)$$

where $\mathbf{y}$ is a tuple of variables different from x_i and $\mathbf{c}$ is the tuple whose entries are the c_s defined previously. It remains to rewrite the condition $x_i \in I_i$. Suppose first that there exists at least one atom of $\mathcal{L}$ outside I_i, and let $a_1, \ldots, a_l$ denote the atoms outside I_i. Then it is clear that $x_i \in I_i$ if and only if $\neg(x_i \sqsupseteq a_1) \vee \cdots \vee \neg(x_i \sqsupseteq a_l)$ holds, so the predicate f is of type (4) (simply restate the disjunction as an implication). Now for the last possibility, where I_i contains all of D except $1_{\mathcal{L}}$; then it is easy to see that f can be described by the following:

$$[\neg(x_i \sqsupseteq b) \wedge (\mathbf{y} \sqsubseteq \mathbf{c})] \vee (x_i \sqsupseteq b)$$

and this completes the proof of the theorem. □

We remark that the preceding theorem can be extended to give a similar characterisation of the supermodular constraints on lattices in the larger class of so-called *relatively complemented lattices*.

5 Supermodular constraints on diamonds are tractable

In this section we prove the main tractability result of this paper.

Theorem 6. *If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{L}}$ then* Max CSP($\mathcal{F}$) *can be solved (to optimality) in $O(t^3 \cdot |\mathcal{L}|^3 + q^3)$ time where t is the number of variables and q is the number of constraints in an instance.*

Construction.
Let $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{L}}$. 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 $V = \{x_1, \ldots, x_t\}$, and let ∞ denote an integer greater than $\sum \rho_i$. For each constraint f_i, fix a representation as described in Theorem 5. In the following construction, we will refer to the type of f_i which will be a number from 1 to 6 according to the type of representation. Every condition of the form $(\mathbf{y} \sqsubseteq \mathbf{c})$ will be read as $\bigwedge(x_{i_s} \sqsubseteq c_{i_s})$, and every condition of the form $\neg(\mathbf{y} \sqsupseteq \mathbf{c})$ as $\bigvee \neg(x_{i_s} \sqsupseteq c_{i_s})$, where i_s runs through the indices of variables in $\mathbf{y}$. Moreover, we replace every (sub)formula of the form $\neg(x \sqsupseteq 1_{\mathcal{L}})$ by $\bigvee_{i=1}^{n} \neg(x \sqsupseteq a_i)$ where $a_1, \ldots, a_n$ are the atoms of $\mathcal{L}$.

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

- The vertices of $G_{\mathcal{I}}$ are as follows
  - $\{T, F\} \cup \{x_d \mid x \in V, d \in \mathcal{L}\} \cup \{\bar{x}_d \mid x \in V, d \in \mathcal{L} \text{ is an atom}\} \cup \{e_i, \bar{e}_i \mid i = 1, 2, \ldots, q\}$[4].

  For each f_i of type (5), we identify the vertex e_i with F. Similarly, for each f_i of type (6), we identify the vertex $\bar{e}_i$ with T.
- The arcs of $G_{\mathcal{I}}$ are defined as follows:
  - For each atom c in $\mathcal{L}$ and for each $x \in V$, there is an arc from $x_{0_{\mathcal{L}}}$ to x_c with weight ∞, and an arc from $\bar{x}_c$ to $x_{1_{\mathcal{L}}}$ with weight ∞;
  - For each pair of distinct atoms c, d in $\mathcal{L}$ and for each $x \in V$, there is an arc from x_c to $\bar{x}_d$ 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 of types (1-4), and each subformula of the form $(x \sqsubseteq a)$ or $\neg(x \sqsupseteq a)$ in the consequent of f_i, there is an arc from e_i to x_a or $\bar{x}_a$, respectively, with weight ∞;

[4] The vertices x_d will correspond to the expressions $x \sqsubseteq d$ and $\bar{x}_d$ to $\neg(x \sqsupseteq d)$.

- For each f_i of types (1-4), and each subformula of the form $(x \sqsubseteq a)$ or $\neg(x \sqsupseteq a)$ in the antecedent of f_i, there is an arc from x_a or $\bar{x}_a$, respectively, to $\bar{e}_i$, with weight ∞;
- For each f_i of type (5), and each subformula of the form $\neg(x \sqsupseteq a)$ in it, there is an arc from $\bar{x}_a$ to $\bar{e}_i$ with weight ∞;
- For each f_i of type (6), and each subformula of the form $(x \sqsubseteq a)$ in it, there is an arc from e_i to x_a 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 (corresponding to **true**) and sink F (corresponding to **false**). Note that paths of non-constraint arcs between vertices corresponding to any given variable $x \in V$ precisely correspond to logical implications that hold between the corresponding assertions.

Proof. We will show how the problem can be reduced to the well-known tractable problem Min Cut.

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 or $\bar{x}_a$ correspond to assertions of the form $x \sqsubseteq a$ or $\neg(x \sqsupseteq a)$, respectively, and arcs denote implications about these assertions. Given a minimal cut in $G_{\mathcal{I}}$, we will call a vertex x_a *reaching* if F can be reached from it without crossing the cut. Furthermore, if a vertex x_a is reaching then this will designate that the corresponding assertion is false, and otherwise the corresponding assertion is true. A constraint is not satisfied if and only if the corresponding constraint arc crosses the cut.

Let C be a minimal cut in $G_{\mathcal{I}}$. Obviously, C contains only constraint arcs. First we show that, for every variable $x \in V$, there is a unique minimal element $a \in \mathcal{L}$ such that x_a is non-reaching. All we need to show is the following: if c, d are distinct atoms such that both x_c and x_d are both non-reaching then so is $x_{0_{\mathcal{L}}}$. Assume that, on the contrary, $x_{0_{\mathcal{L}}}$ is reaching. Then there is a path from $x_{0_{\mathcal{L}}}$ to F not crossing the cut. By examining the arcs of $G_{\mathcal{I}}$, it is easy to notice that such a path has to go through a vertex $\bar{x}_a$ for some atom $a \in \mathcal{L}$. However, we have an arc from at least one of vertices x_c, x_d to $\bar{x}_a$, and hence at least one of this vertices would have a path to F not crossing the cut, a contradiction.

Note that, for every $x \in V$, there no arcs coming out of $x_{1_{\mathcal{L}}}$. Hence, for every $x \in V$, there is a unique minimal element $v \in \mathcal{L}$ such that F cannot be reached from x_v without crossing the cut.

Define an assignment ϕ_C as follows:

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

We now make some observations. Note that, for all $x \in V$ and $a \in \mathcal{L}$, we have that $\phi_C(x) \sqsubseteq a$ if and only x_a is non-reaching. Moreover, if $\bar{x}_a$ is reaching then, for each atom $b \neq a$, we have an arc from x_b to $\bar{x}_a$ meaning that $\phi_C(x) \not\sqsubseteq b$, and hence $\phi_C(x) \sqsupseteq a$. Furthermore, if $\bar{x}_a$ is non-reaching then $\phi_C(x) \neq a$. Indeed, if $\phi_C(x) = a$ then x_b is reaching for all atoms $b \neq a$, and, since every path from x_b to F has to go through a vertex $\bar{x}_c$ for some c, we have that $\bar{x}_c$ is reaching. Then $c \neq a$, and there is an arc from x_a to $\bar{x}_c$, so x_a is reaching, a contradiction. To summarize,

– if a node of the form x_a or $\bar{x}_a$ is reaching then the corresponding assertion is falsified by the assignment ϕ_C;
– if a node of the form x_a is non-reaching then $\phi_C(x) \sqsubseteq a$;
– if a node of the form $\bar{x}_a$ is non-reaching then the truth value of the corresponding assertion is undecided.

Suppose that a constraint arc corresponding to a constraint f_i is not in the cut. We claim that f_i is satisfied by the assignment ϕ_C. To show this, we will go through the possible types of f_i.

If f_i is of type (1), (2), (5), or (6), then the claim is straightforward. For example, let f_i be of type (1). If the node $x_{0_\mathcal{L}}$ corresponding to the consequent is reaching, then so are all nodes corresponding to the antecedent. Hence, all atomic formulas are falsified by the assignment ϕ_C, and the implication is true. If $x_{0_\mathcal{L}}$ is non-reaching, then $\phi_C(x) = 0_\mathcal{L}$, and the constraint is clearly satisfied. The argument for types (2), (5), (6) is very similar.

Let f_i be of type (3). Then, if the node corresponding to the consequent is non-reaching then the consequent is satisfied by ϕ_C, and so the constraint is satisfied. If this node is reaching then every node corresponding to the disjuncts in the antecedent is reaching. Then both antecedent and consequent are falsified by ϕ_C, and the constraint is satisfied.

Let f_i be of type (4), that is, of the form

$$\neg(x_i \sqsupseteq b) \implies [\neg(x_i \sqsupseteq a_1) \wedge \cdots \wedge \neg(x_i \sqsupseteq a_l) \wedge (\mathbf{y} \sqsubseteq \mathbf{c})].$$

If a node corresponding to some conjunct in the consequent is reaching, then the node corresponding to the antecedent is also reaching. So $\phi_C(x_i) \sqsupseteq b$, and the constraint is satisfied. More generally, if the node corresponding to the antecedent is reaching then the constraint is satisfied regardless of what happens with the consequent. Assume that all nodes corresponding to conjuncts in the consequent and in the antecedent are non-reaching. Then the conjunct $(\mathbf{y} \sqsubseteq \mathbf{c})$ is satisfied by ϕ_C. Furthermore, we know (see the observations above) that $\phi_C(x_i) \neq b$, and also that $\phi_C(x_i) \neq a_s$ for $1 \leq s \leq l$. If $\phi_C(x_i) = 1_\mathcal{L}$ then both the antecedent and the consequent of f_i are false, and hence f_i is satisfied. Otherwise, $\phi_C(x_i) \not\sqsupseteq b$ and $\phi_C(x_i) \not\sqsupseteq a_s$ for $1 \leq s \leq l$, so f_i is satisfied anyway.

Conversely, let ϕ be an assignment to $\mathcal{I}$, and let K be the set of constraints in $\mathcal{I}$ that are not satisfied by ϕ. Consider any path from T to F. It is clear that if all constraints corresponding to constraint arcs on this path are satisfied, then we have a chain of valid implications starting from **true** and finishing at **false**. Since this is impossible, at least one constraint corresponding to such an arc is not satisfied by ϕ. Hence, the constraints arcs corresponding to constraints in 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 [16] for the Min Cut problem can be used to find an optimal assignment for any instance of Max CSP($\mathcal{F}$). This algorithm runs in $O(k^3)$ where k is the number of vertices in the graph. Since the number of vertices in $G_{\mathcal{I}}$ is at most $2(1 + t \cdot |D| + q)$, the result follows. □

6 A partial converse

We will now prove a partial converse to Theorem 6.

The following theorem shows that, in order to establish that a given function f is supermodular on a given lattice, it is sufficient to prove supermodularity of certain unary and binary functions derived from f by substituting constants for variables. This result was proved in [13] for submodular functions on lattices, but clearly it is also is true for supermodular functions because f is supermodular if and only if $-f$ is submodular.

Theorem 7 ([13]). *An n-ary function f is supermodular on $\mathcal{L}$ if and only if it satisfies inequality (1) for all $\mathbf{a}, \mathbf{b} \in \mathcal{L}^n$ such that*

- *$a_i = b_i$ with one exception, or*
- *$a_i = b_i$ with two exceptions, and, for each i, the elements a_i and b_i are comparable in $\mathcal{L}$.*

Theorem 8. *Let $\mathcal{F}$ contain all at most binary 2-monotone predicates on $\mathcal{L}$. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{L}}$ then* Max CSP($\mathcal{F}$) *is tractable. Otherwise,* Max CSP($\mathcal{F}$) *is* **NP**-*hard.*

Proof. If $\mathcal{F} \subseteq \mathrm{Spmod}_{\mathcal{L}}$ then the result follows from Theorem 6. Otherwise, there is a predicate $f \in \mathcal{F}$ such that $f \notin \mathrm{Spmod}_{\mathcal{L}}$. First, we prove that we can assume f to be at most binary. By Theorem 7, we can substitute constants for all but at most two variables in such a way that the obtained predicate f' is not supermodular on $\mathcal{L}$. We now show that f' can be assumed to be in $\mathcal{F}$. We will consider the case when f' is binary, the other case (when f' is unary) is similar. Assume without loss of generality that $f'(x_1, x_2) = f(x_1, x_2, a_1, \ldots, a_p)$, and reduce Max CSP($\mathcal{F} \cup \{f'\}$) to Max CSP($\mathcal{F}$). Let $\mathcal{I}$ be an instance of Max CSP($\mathcal{F} \cup \{f'\}$) and W the total weight of all constraints in $\mathcal{I}$, plus 1. Transform $\mathcal{I}$ into an instance $\mathcal{I}'$ of Max CSP($\mathcal{F}$) as follows:

1. For each constraint $f_i(\mathbf{x}_i)$ in $\mathcal{I}$ such that $f_i = f'$

  – replace f' with f
  – keep the same first two variables as in the original constraint
  – introduce fresh variables $y^i_1, \dots, y^i_p$ for the last n variables.
2. For every new variable y^i_s introduced in step 1, add
  – a constraint $y^i_s \sqsupseteq a_s$ with weight W, and
  – a constraint $y^i_s \sqsubseteq a_s$, with weight W.

Clearly this transformation can be performed in polynomial time, and the constraints added in step two above ensure that, in every optimal solution to $\mathcal{I}'$, every variable y^i_s takes the value a_s. Hence, optimal solutions to $\mathcal{I}$ and to $\mathcal{I}'$ precisely correspond to each other. So, indeed, f can be assumed to be at most binary. We consider the two cases separately.

Case 1. f is unary.
By Lemma 1, $f = u_{D'}$ for some non-empty $D' \subseteq D$ such that either $0_\mathcal{L}, 1_\mathcal{L} \notin D'$ or else $|D'| > 2$ and at least one of $0_\mathcal{L}, 1_\mathcal{L}$ is not in D'. If $f = u_{\{a\}}$ where a is an atom then we choose another atom b and consider the predicate $u_{\{1_\mathcal{L},a,b\}}$. Note that the predicate $u_{\{1_\mathcal{L},b\}}$ is 2-monotone on $\mathcal{L}$ (and hence belongs to $\mathcal{F}$), and $u_{\{1_\mathcal{L},a,b\}}(x) = u_{\{1_\mathcal{L},b\}}(x) + f(x)$. Hence, we may assume that $u_{\{1_\mathcal{L},a,b\}} \in \mathcal{F}$, since, in any instance, this predicate can be replaced by the sum above. It follows that we can now assume that $f = u_{D'}$ where two distinct atoms a, b belong to D', but at least one of $0_\mathcal{L}, 1_\mathcal{L}$ (say, $0_\mathcal{L}$) does not. We will show how to reduce Max CSP($\{f_{dicut}\}$) (see Example 1) to Max CSP($\mathcal{F}$). Assume that the the domain D for Max CSP($\{f_{dicut}\}$) is $\{a, b\}$ where a plays the role of 0 and b that of 1. Let $g \in \mathcal{F}$ be such that $g(x, y) = 1 \Leftrightarrow [(x \sqsubseteq a) \wedge (y \sqsubseteq b)]$.

Take an arbitrary instance $\mathcal{I}$ of Max CSP($\{f_{dicut}\}$). Replace each constraint $f_{dicut}(x, y)$ by $g(x, y)$ with the same weight. Let W be the total weight of all constraints in $\mathcal{I}$ plus 1. For every variable x in $\mathcal{I}$, add the constraint $f(x)$ with weight W and denote the obtained instance by $\mathcal{I}'$. Note that any solution to $\mathcal{I}'$ that assigns $0_\mathcal{L}$ to any variable is suboptimal because it violates one of the large-weight constraints. Moreover, if a solution assigns a value d to some variable, and $d \notin \{a, b\}$, then d can be changed to one of a, b without decreasing the total weight of the solution. Hence, there is an optimal solution to $\mathcal{I}'$ which uses only values a and b. Clearly, this solution is also optimal for $\mathcal{I}$. The other direction is similar, since any optimal solution to $\mathcal{I}$ is also an optimal solution to $\mathcal{I}'$, or else the transformation of solutions to $\mathcal{I}'$ such as described above would produce a better solution to $\mathcal{I}$.

Case 2. f is binary.
Note that, by Theorem 7, if we cannot use case 1 then the tuples $\mathbf{a} = (a_1, a_2)$ and $\mathbf{b} = (b_1, b_2)$ witnessing non-supermodularity of f can be chosen in such a way that a_i and b_i are comparable for $i = 1, 2$. For $i = 1, 2$, define functions $t_i : \{0, 1\} \to \{a_i, b_i\}$ by the following rule:

– 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 binary function $g' \in R_{\{0,1\}}$ such that $g'(x_1, x_2) = g(t_1(x_1), t_2(x_2))$ 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 that MAX CSP($\mathcal{F}'$) on $\{0,1\}$, where $\mathcal{F}' = \{g', c'_0, c'_1\}$, is **NP**-hard. (Note that that we include c'_0, c'_1 to ensure that MAX CSP($\mathcal{F}'$) is non-trivial). We will give a polynomial time reduction from this problem to MAX CSP($\mathcal{F}$).

In the reduction, we will use functions $h_i(x,y)$, $i = 1,2$, 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 2-monotone on $\mathcal{L}$. In the rest of the proof we identify $0_\mathcal{L}, 1_\mathcal{L}$ with the corresponding elements 0,1 from the domain of $\mathcal{F}'$. Other functions used in the reduction are $u_{\{0\}}$, $u_{\{1\}}$, $u_{\{0,1\}}$, $u_{\{a_1,b_1\}}$, $u_{\{a_2,b_2\}}$. By Lemma 1, all these functions are supermodular on $\mathcal{L}$, and, in fact, they are 2-monotone.

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 $1 \le i \le q$ such that $f'_i(\mathbf{x}_i) = g'(x_{j_1}, x_{j_2})$, introduce
  - two new variables $y^i_{j_1}, y^i_{j_2}$,
  - constraint $g(y^i_{j_1}, y^i_{j_2})$ with weight ρ_i,
  - constraints $u_{\{a_1,b_1\}}(y^i_{j_1}), u_{\{a_2,b_2\}}(y^i_{j_2})$, each with weight W,
  - constraints $h_1(x_{j_1}, y^i_{j_1}), h_2(x_{j_2}, y^i_{j_2})$, each with weight W;
- for every $1 \le i \le q$ 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 $1 \le i \le q$ 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 $y^i_{j_s}$ $(s = 1,2)$, 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 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 $y^i_{j_s}$, $s = 1, 2$, 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. □

7 Conclusion

We have proved that the Max CSP problem for constraints that are supermodular on diamonds is tractable. This is the first result about tractability of all supermodular constraints on non-distributive lattices. One natural extension of this line of research is to establish similar results for other classes of non-distributive lattices. It would interesting to explore methods of proving tractability of Max CSP other than via a reduction to submodular set function minimization (as in [8]) or via an explicit description of predicates (as in this paper). Can the technique of multimorphisms [6, 7] be effectively used in the study of non-Boolean Max CSP? Another interesting direction for future work is to study approximability of hard Max CSP problems. It is known that, for $|D| \leq 3$, all hard problems Max CSP($\mathcal{F}$) are **APX**-complete [10, 21], that is, they do not admit a polynomial-time approximation scheme. Is this true for larger domains?

Acknowledgments

Part of this research was conducted when the first author was employed at the University of Warwick, and the second author was visiting this university. This work was partially supported by the UK EPSRC grant GR/T05325/01.

References

1. F. Börner, A. Bulatov, P. Jeavons, and A. Krokhin. Quantified constraints: Algorithms and complexity. In *CSL'03*, volume 2803 of *LNCS*, pages 58–70, 2003.
2. A. Bulatov. A dichotomy theorem for constraints on a 3-element set. In *FOCS'02*, pages 649–658, 2002.
3. A. Bulatov. Tractable conservative constraint satisfaction problems. In *LICS'03*, pages 321–330, 2003.
4. A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. In *FOCS'03*, pages 562–571, 2003.
5. R.E. Burkard, B. Klinz, and R. Rudolf. Perspectives of Monge properties in optimization. *Discrete Applied Mathematics*, 70:95–161, 1996.
6. D. Cohen, M. Cooper, and P. Jeavons. A complete characterization of complexity for Boolean constraint optimization problems. In *CP'04*, volume 3258 of *LNCS*, 2004.
7. D. Cohen, M. Cooper, P. Jeavons, and A. Krokhin. Soft constraints: complexity and multimorphisms. In *CP'03*, volume 2833 of *LNCS*, pages 244–258, 2003.

8. D. Cohen, M. Cooper, P. Jeavons, and A. Krokhin. Identifying efficiently solvable cases of Max CSP. In *STACS'04*, volume 2996 of *LNCS*, pages 152–163, 2004.
9. N. Creignou. A dichotomy theorem for maximum generalized satisfiability problems. *Journal of Computer and System Sciences*, 51:511–522, 1995.
10. 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.
11. M. Datar, T. Feder, A. Gionis, R. Motwani, and R. Panigrahy. A combinatorial algorithm for MAX CSP. *Information Processing Letters*, 85(6):307–315, 2003.
12. B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2nd edition, 2002.
13. B.L. Dietrich and A.J. Hoffman. On greedy algorithms, partially ordered sets, and submodular functions. *IBM Journal of Research and Development*, 47(1):25–30, 2003.
14. L. Engebretsen and V. Guruswami. Is constraint satisfaction over two variables always easy? *Random Structures and Algorithms*, 25(2):150–178, 2004.
15. S. Fujishige. *Submodular Functions and Optimization*, volume 47 of *Annals of Discrete Mathematics*. North-Holland, 1991.
16. A. Goldberg and R.E. Tarjan. A new approach to the maximum flow problem. *J. ACM*, 35:921–940, 1988.
17. M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. In *FOCS'03*, pages 552–561, 2003.
18. J. Håstad. Some optimal inapproximability results. *J. ACM*, 48:798–859, 2001.
19. S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *J. ACM*, 48(4):761–777, 2001.
20. P. Jonsson. Boolean constraint satisfaction: Complexity results for optimization problems with arbitrary weights. *Theoret. Comput. Sci.*, 244(1-2):189–203, 2000.
21. P. Jonsson, M. Klasson, and A. Krokhin. The approximability of three-valued Max CSP. Submitted for publication, 2004.
22. D. Topkis. *Supermodularity and Complementarity*. Princeton University Press, 1998.