Supermodularity on chains and complexity of maximum constraint satisfaction

Vladimir Deineko[1] and Peter Jonsson[2] and Mikael Klasson[3] and Andrei Krokhin[4]

[1] *Warwick Business School, University of Warwick, UK*
email: Vladimir.Deineko@wbs.ac.uk
[2] *Department of Computer and Information Science, University of Linköping, Sweden*
email: peter.jonsson@ida.liu.se
[3] *Department of Computer and Information Science, University of Linköping, Sweden*
email: mikael.klasson@ida.liu.se
[4] *Department of Computer Science, University of Durham, UK*
email: andrei.krokhin@durham.ac.uk

Keywords: maximum constraint satisfaction, complexity, supermodularity, Monge properties, digraph H-colouring

1 Introduction and Related Work

The *constraint satisfaction problem* (CSP) is a general framework in which a variety of combinatorial problems, including propositional satisfiability and graph colourability, can be expressed in a natural way [5, 8]. In this paper, we study the *maximum constraint satisfaction problem* (MAX CSP) which is a natural optimization version of CSP. Informally speaking, in a CSP, one is given a finite collection of constraints on overlapping sets of variables, and the goal is to assign values to the variables so as to satisfy all constraints; in MAX CSP, the constraints are weighted, and the goal is to maximize the total weight of satisfied constraints. Well-known examples of MAX CSP problems include MAX k-SAT and MAX CUT.

Constraint problems can be naturally parameterised by the types of constraints allowed in instances, and the study of complexity of such parameterised problems is a very active research area [5, 8]. Two classical results in the area are classifications of complexity for two important special cases: Schaefer's classification of Boolean CSPs (or SAT) (see [5]), when the set of possible values is $\{0,1\}$, and Hell and Nešetřil's classification of GRAPH H-COLOURING (see [8]), when there is only one allowed (binary) constraint type which is specified by an undirected graph H. The general classification problem, both for CSP and for MAX CSP, is open and known to be very difficult, though significant progress has recently been made on it, largely due to algebraic techniques (e.g., [1, 2, 4, 9]).

Sub- and supermodularity and *(anti-)Monge* properties are well-known sources of tractability in combinatorial optimization [3, 7, 11]. Two forms of sub- and supermodular functions have been extensively studied in the literature. One form is set functions, defined on subsets of a set [7], which plays a significant role in many areas in combinatorics. The other form is functions defined on products of finite totally ordered sets (or *chains*) [3]. Such functions are usually represented in the form of arrays or matrices, which are called *(anti-)Monge* arrays and matrices; they have been considered mostly in operations research [3]. A more general form of sub- and supermodularity is the one where functions are defined on general (algebraic) lattices. This form subsumes the previous two forms, but it is not so widely used in combinatorics. Curiously, this form of sub- and supermodularity is popular in mathematical economics where it is used to model games in which an optimal strategy can be found efficiently (e.g., supermodular games) [11].

This general form of supermodularity has been recently shown to be highly relevant in the study of MAX CSP where it captures all currently known tractable cases [4, 9]. In this paper we continue investigation of MAX CSP via supermodularity and provide further evidence to the thesis that the (general) property of supermodularity captures tractability in MAX CSP. We consider MAX CSP under a (very) mild assumption that the set of allowed constraint types contains all constraints of the form $x = a$. The only form of supermodularity applicable in this case is supermodularity on chains, and our main result states that it precisely characterizes tractable MAX CSP problems (under the above assumption).

The main technical result of the paper contributes to a line of research in combinatorics which aims at characterizing matrices avoiding given submatrices after certain transformations (see, e.g., [3, 6, 10]). For example, a number of results on 0-1 matrices, which become Monge after some permutations of rows and columns, is obtained in [10]. However, the case when the matrix is square and the rows and the columns must be permuted by the *same* permutation is left there as an open question, with almost nothing known about it (even now). We obtain several results clarifying the picture in this interesting case.

We conclude the paper with a discussion of the list H-colouring optimization problem for digraphs.

2 Supermodularity and Monge properties

Throughout the paper, let D be a finite set. A lattice on D is a partial order on D in which each pair (a, b) of elements has a greatest lower bound $a \sqcap b$ and a least upper bound $a \sqcup b$. Any lattice can be considered as an algebra with two binary operations $\sqcap$ and $\sqcup$. Let $\mathcal{L} = (D, \sqcap, \sqcup)$ be a lattice on D, and extend the notation to tuples: for $\mathbf{a} = (a_1, \ldots, a_n)$, $\mathbf{b} = (b_1, \ldots, b_n)$ 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. A function $f : D^n \to \mathbb{R}$ is called *supermodular* on $\mathcal{L}$ if

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

and f is called *submodular* on $\mathcal{L}$ if $-f$ is supermodular on $\mathcal{L}$.

Note that, in the case when $D = \{0, 1\}$, this definition precisely corresponds to the usual definitions of sub- and supermodular set functions [7]. Totally ordered lattices are typically known as *chains*. In this case, the operations $\sqcap$ and $\sqcup$ are simply $\min$ and $\max$. Clearly, every binary function f on a chain can be considered as a matrix (by setting $f(i, j) = M_{ij}$), the total order being the order of indices. Matrices obtained in this way from binary supermodular functions are known as *anti-Monge* (or *a-Monge*, for short) matrices [3]. In other words, a matrix $M = (m_{ij})$ is a-Monge if $m_{is} + m_{rj} \leq m_{ij} + m_{rs}$ for all $i < r$ and $j < s$. In the same way, n-ary supermodular functions correspond to anti-Monge n-dimensional arrays (see Observation 6.1 in [3]). Monge matrices and arrays are obtained in the same way from submodular functions. The structure of Monge matrices is described in Lemma 2.3 [3]. Using this result, it is easy to show that a 0-1 matrix is a-Monge if and only if it can be obtained by the following simple procedure: arbitrarily choose at most one top-left rectanglular area and at most one bottom-right rectanglular area in the matrix; these areas may intersect. Assign 1 to all entries in these chosen areas (if there are any). After that, if there are rows or columns which do not contain any assigned entries, then choose *either* some (possibly none) of such rows *or* some (possibly none) of such columns, and assign 1 to all entries in the chosen rows or columns. Finally, assign 0 to all unassigned entries.

If there is a permutation π that *simultaneously* permutes rows and columns of M so that the resulting matrix is an a-Monge matrix, then the matrix M is called a *permuted* a-Monge matrix and the permutation is called an *a-Monge permutation* for M. For a set B of indices of M, we write $M[B]$ to denote the matrix obtained from M by deleting all rows and columns *not* contained in B. Clearly, if, for some B, $M[B]$ is not permuted a-Monge then neither is M. How small can the set B be chosen? Our main technical result provides an answer to this question.

Theorem 2.1 *If a square matrix M is not a permuted a-Monge matrix, then there exists a set of indices B such that $|B| \leq 4$ and $M[B]$ is not a permuted a-Monge matrix.*

It can be easily shown that the above bound on $|B|$ is tight. The proof of Theorem 2.1 is based on (an elaborated version of) the algorithm from [6]. For our analysis of MAX CSP problems, we will actually use the following non-trivial corollary from Theorem 2.1 and its proof.

Corollary 2.2 *Suppose that $M_1, \ldots, M_k$, $k \geq 1$, are square 0-1 matrices of the same size such that no permutation of indices is an a-Monge permutation for all of $M_1, \ldots, M_k$. Then there exists a set of indices B with $|B| \leq 4$ such that no permutation on B is an a-Monge permutation for all of $M_1[B], \ldots, M_k[B]$.*

3 Constraint satisfaction problems

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 $R_D = \bigcup_{m=1}^{\infty} R_D^{(m)}$. We will view the 0-1 values taken by predicates as integers, not as Boolean values.

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 $f(\mathbf{a}) = 1$. Let $\mathcal{F}$ be a *finite* subset of R_D. An instance of the problem CSP($\mathcal{F}$) is a pair (V, C) where $V = \{x_1, \ldots, x_n\}$ is a set of variables and C is a collection of constraints $f_1(\mathbf{x}_1), \ldots, f_q(\mathbf{x}_q)$ over V, with $f_i \in \mathcal{F}$ for all $1 \leq i \leq q$. The question is whether there is an assignment $\varphi : V \to D$ which satisfies all constraints in C.

For a subset $D' \subseteq D$, let $u_{D'}$ denote the predicate such that $u_{D'}(x) = 1$ if and only if $x \in D'$. Let $\mathcal{U}_D = \{u_{D'} \mid D' \subseteq D\}$ and $\mathcal{C}_D = \{u_{\{d\}} \mid d \in D\}$. Note that predicates from $\mathcal{C}_D$ give rise to constraints of the form $x = d$. The problems of the form CSP($\mathcal{F} \cup \mathcal{U}_D$) are known as conservative (or list) CSPs, and their complexity has been completely classified in [1], while a complexity classification for the problems of the form CSP($\mathcal{F} \cup \mathcal{C}_D$) would imply a classification for all problems CSP($\mathcal{F}$) [2].

Let us now turn to optimization problems. An instance of the *weighted* MAX CSP($\mathcal{F}$) problem is a tuple (V, C, ρ) where V and C are the same as for CSP($\mathcal{F}$), and $\rho : C \to \mathbb{Z}^+$ is a function that assigns a (positive integral) weight ρ_i to each constraint $f_i(\mathbf{x}_i)$. The goal is to maximize the total weight of satisfied constraints, i.e, to maximize the function $f : D^n \to \mathbb{Z}^+$ defined by $f(x_1, \ldots, x_n) = \sum_{i=1}^{q} \rho_i \cdot f_i(\mathbf{x}_i)$.

It is well-known [5] (and not difficult to see) that the case when $D = \{0, 1\}$ and $\mathcal{F} = \{h\}$ with $h(x, y) = 1 \Leftrightarrow x \neq y$ precisely corresponds to the well-known MAX CUT problem.

We say that a set $\mathcal{F} \subseteq R_D$ is supermodular on a chain on D if every predicate in $\mathcal{F}$ has this property.

Theorem 3.1 *[5, 4, 9] Let $|D| \leq 3$ and $\mathcal{F} \subseteq R_D$. Then either the problem* MAX CSP($\mathcal{F}$) *is equivalent to a similar problem on a smaller domain obtained by removing some value from D, or $\mathcal{F}$ is supermodular on some chain on D and then* MAX CSP($\mathcal{F}$) *is tractable, or, otherwise,* MAX CSP($\mathcal{F}$) *is* **NP**-*hard.*

Note that, for $|D| \leq 3$, the only lattices on D are chains. Now let D be an arbitrary finite set. Another natural case where chains are the only lattices that must be considered is the following one. It follows from Lemma 1 [4] (and is easy to see) that every predicate from $\mathcal{C}_D$ (or from $\mathcal{U}_D$) is supermodular on a lattice on D if and only if the lattice is a chain. Therefore, it probably is natural to expect that supermodularity on chains will play a role in classifying problems of the form MAX CSP($\mathcal{F} \cup \mathcal{C}_D$) and MAX CSP($\mathcal{F} \cup \mathcal{U}_D$). And, indeed, this turns out to be the case, as our main result shows.

Theorem 3.2 *If $\mathcal{F}$ is supermodular on some chain on D, then the problems* MAX CSP($\mathcal{F} \cup \mathcal{C}_D$) *and* MAX CSP($\mathcal{F} \cup \mathcal{U}_D$) *are tractable. Otherwise, both problems are* **NP**-*hard.*

Let us now outline the proof of Theorem 3.2. The tractability part immediately follows from Theorem 3 in [4]. To prove the hardness part, we use two reduction techniques: *strict implementations* and *domain restrictions*. Strict implementations [5, 9] provide a way of constructing new predicates from a given set of predicates. If $g_1, \ldots, g_s \in \mathcal{F}$ and $g(y_1, \ldots, y_m)$ is a predicate such that, for all $y_1, \ldots, y_m$, we have $g(y_1, \ldots, y_m) + (\alpha - 1) = \max_W \sum_{i=1}^{s} g_i(\mathbf{y}_i)$ where $\alpha \in \mathbb{Z}^+$ and W is some set of variables appearing in the $\mathbf{y}_i$'s, then we say that $\mathcal{F}$ strictly implements g. For $D' \subseteq D$, let $\mathcal{F}|_{D'} = \{f|_{D'} \mid f \in \mathcal{F}\}$.

Lemma 3.3 *Suppose that $\mathcal{F}$ strictly implements a predicate g, and* MAX CSP($\mathcal{F} \cup \{g\}$) *is* **NP**-*hard. Then* MAX CSP($\mathcal{F}$) *is* **NP**-*hard as well.*

Lemma 3.4 *Suppose that $u_{D'} \in \mathcal{F}$. If* MAX CSP($\mathcal{F}|_{D'}$) *is* **NP**-*hard then so is* MAX CSP($\mathcal{F}$).

Proof sketch: (of Theorem 3.2) First we use Lemma 3.3 to show that the problems MAX CSP($\mathcal{F} \cup \mathcal{C}_D$) and MAX CSP($\mathcal{F} \cup \mathcal{U}_D$) are polynomial-time equivalent, thus we can consider only the latter problem.

Assume now that $\mathcal{F}$ is not supermodular on any chain on D. It follows from Lemma 6.3 [3] that an n-ary ($n \geq 2$) function f is supermodular on a chain if and only if the following holds: every binary function obtained from f by replacing any given $n-2$ variables by any constants is supermodular on the chain. By using this together with Lemma 3.3, we show that it is enough to consiuder the case when $\mathcal{F}$ only contains binary predicates. By using Corollary 2.2 and the correspondence between a-Monge matrices and binary supermodular predicates, we derive that there is a subset $B \subseteq D$, $|B| \leq 4$, such that $\mathcal{F}|_B$ is not supermodular on any chain on B. Moreover, it can be proved that there exist at most three binary predicates in $\mathcal{F}$ such that these predicates are not simultaneously supermodular on any chain on B. Consequently, we may henceforth assume that $|\mathcal{F}| \leq 3$ and, by Theorem 3.1 and Lemma 3.4, that the predicates in $\mathcal{F}$ are over a domain B such that $|B| = 4$. This reduces the proof to a relatively small number of cases. By using Theorem 3.1 and employing a number of symmetries, we reduce the number of cases under consideration down to 54. We then show that each of these 54 possible sets $\mathcal{F}$ strictly implements some predicates with an **NP**-hard MAX CSP problem. This can easily be done by a computer-assisted case analysis, or, with more effort, combinatorially. □

4 List H-colouring optimization

In this section, we consider the case when $\mathcal{F}$ consists of a single binary predicate h. This predicate specifies a digraph H such that $V_H = D$ and (u, v) is an arc in H if and only if $h(u, v) = 1$. Any instance $I = (V, C)$ of CSP($\{h\}$) can be associated with a digraph G_I whose nodes are the variables in V and whose arcs are the scopes of constraints in C. It is not difficult to see that the question whether I is satisfiable is equivalent to the question whether there is a homomorphism from G_I to H. Therefore, the problem CSP($\{h\}$) is precisely the GRAPH H-COLOURING problem for the digraph H [8]. The problems

CSP($\{h\} \cup \mathcal{U}_D$) and CSP($\{h\} \cup \mathcal{C}_D$) are equivalent to the LIST H-COLOURING and H-RETRACTION problems, respectively. In the former problem, every vertex of an input digraph G gets a list of allowed target vertices in H, and the question is whether G has an H-colouring subject to the list constraints. The latter problem is the same except that each list contains either one or all vertices of H. These problems attract much attention in graph theory [8].

The problem MAX CSP($\{h\} \cup \mathcal{U}_D$) can then be viewed as follows: for every vertex v of an input digraph G, there is a list $L_v \subseteq V_H$ along with a function $\rho_v : L_v \to \mathbb{Z}^+$ that indicates the 'score' which a mapping $V_G \to V_H$ gets if it sends v to a certain vertex (if a mapping sends v to a vertex outside of L_v then this adds nothing to the 'value' of this mapping). Then the goal is to maximize the combined 'value' of such a mapping which is obtained by adding weights of preserved arcs and 'scores' from the lists. The 'score' functions ρ_v arise as the result of the possible presence in C of several weighted constraints of the form $u_{D'}(v)$ for different $D' \subseteq D$ and the same v. Thus, Theorem 3.2 in the case when $\mathcal{F} = \{h\}$ presents a complexity classification of list H-colouring optimization problems. Digraphs H corresponding to the tractable cases of this problem are the digraphs that have an a-Monge adjacency matrix (under some total ordering on V_H). For a description of such matrices (and, hence, of such digraphs), see Section 2.

References

[1] A. Bulatov. Tractable conservative constraint satisfaction problems. In *Logic in Computer Science, LICS'03*, pages 321–330, 2003.

[2] A. Bulatov, P. Jeavons, and A. Krokhin. Classifying complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.

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

[4] 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.

[5] 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.

[6] V.G. Deineko, R. Rudolf, and G.J. Woeginger. A general approach to avoiding 2×2 submatrices. *Computing*, 52:371–388, 1994.

[7] S. Fujishige. *Submodular Functions and Optimization*. North-Holland, 1991.

[8] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[9] P. Jonsson, M. Klasson, and A. Krokhin. The approximability of three-valued Max CSP. Technical Report cs.CC/0412042, CoRR, 2004.

[10] B. Klinz, R. Rudolf, and G. Woeginger. Permuting matrices to avoid forbidden submatrices. *Discrete Applied Mathematics*, 60:223–248, 1995.

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