Max CSP and lattices

Peter Jonsson

Linköpings universitet

Outline:

- The Max CSP problem
- Tools: Lattices, supermodularity, implementations, and cores
- Results and open questions:
  - $|D| = 3$
  - non-distributive lattices
  - single-predicate Max CSP
  - Max CSP with all constants ($x = 0$, $x = 1$,...)

Basics

D – a finite set with $|D| > 1$ (the *domain*)

$R_D^{(m)} = \{f \mid f : D^m \to \{0,1\}\}$ – the set of all m-ary *predicates*

$R_D = \bigcup_{m=1}^{\infty} R_D^{(m)}$

Definition: 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 the *constraint predicate*; and

$\mathbf{x} = (x_{i_1}, \ldots, x_{i_m})$ is the *constraint scope*.

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

Max CSP

A collection $C = \{f_1(\mathbf{x}_1), \ldots, f_m(\mathbf{x}_m)\}$ of constraints over $V = \{x_1, \ldots, x_n\}$;

each constraint $f_i(\mathbf{x}_i)$ has a weight $\alpha_i \in \mathbb{N}$.

Find an *assignment* $\phi : V \to D$ that maximizes the total weight of satisfied constraints; in other words, maximize the function $f : D^n \to \mathbb{N}$, defined by

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

Parameterization of Max CSP

For a finite set of predicates $\Gamma \subseteq R_D$, Max CSP(Γ) is the set of Max CSP instances where all constraint predicates belong to Γ.

We say that Γ is a *constraint language*.

Example

In the Max k-Cut problem, one is given an edge-weighted graph, and the goal is to divide it into k parts so as to maximize the total weight of edges between different parts.

Let neq_k be the disequality predicate on $\{0, \ldots, k-1\}$, that is, $neq_k(x, y) = 1$ iff $x \neq y$. Then Max k-Cut $\equiv$ Max CSP($\{neq_k\}$).

Approximation

PO is the class of optimization problems that can be solved to optimality in polynomial time.

APX is the class of optimization problems that can be optimized (in polynomial time) within some constant $c > 1$:

$$\frac{OPT(I)}{c} \leq m(A(I)) \leq c \cdot OPT(I)$$

Max CSP(Γ) can be approximated within $|D|^a$ where a is the maximum arity of predicates in Γ.

A problem S is **APX**-complete if every problem in **APX** can be AP-reduced to S.

If S is **APX**-complete and **P** $\neq$ **NP**, then

- there exists a constant $c > 0$ such that S is not c-approximable;

- S does not admit a polynomial-time approximation scheme;

- it is **NP**-hard to solve S exactly.

Max CSP($\{neq_k\}$) is **APX**-complete.

Classification when $|D| = 2$

[Creignou] [Khanna, Sudan, Williamson]

Let Γ be a constraint language over $\{0, 1\}$. Max CSP(Γ) $\in$ **PO** if and only if

- Γ is 0-valid; or
- Γ is 1-valid; or
- Γ is 2-monotone.

Otherwise, Γ is **APX**-complete.

A predicate $f : \{0,1\}^n \to \{0,1\}$ is 2-monotone if f can be expressed as follows:

$$f(x_1, \ldots, x_n) = 1$$

$$\Longleftrightarrow$$

$$(x_{i_1} \wedge \ldots \wedge x_{i_s}) \vee (\neg x_{j_1} \wedge \ldots \wedge \neg x_{j_t})$$

Both disjuncts are not required to contain literals.

$\boxed{\text{Tools}}$

- Lattices and supermodularity

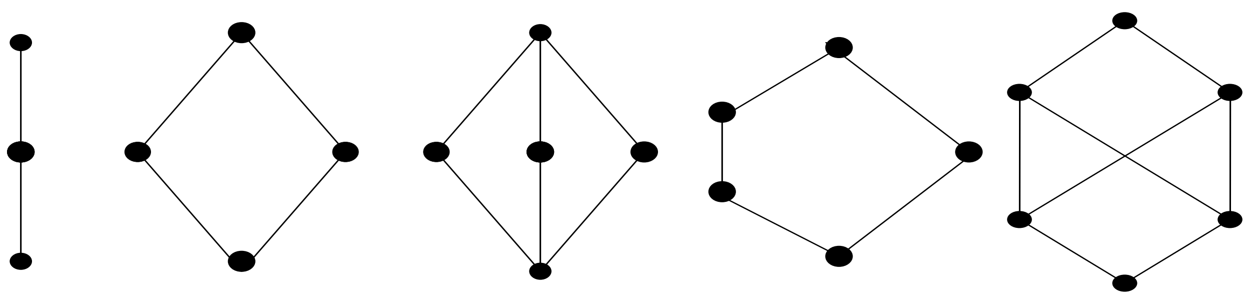
- Strict implementations

- Cores

Lattices

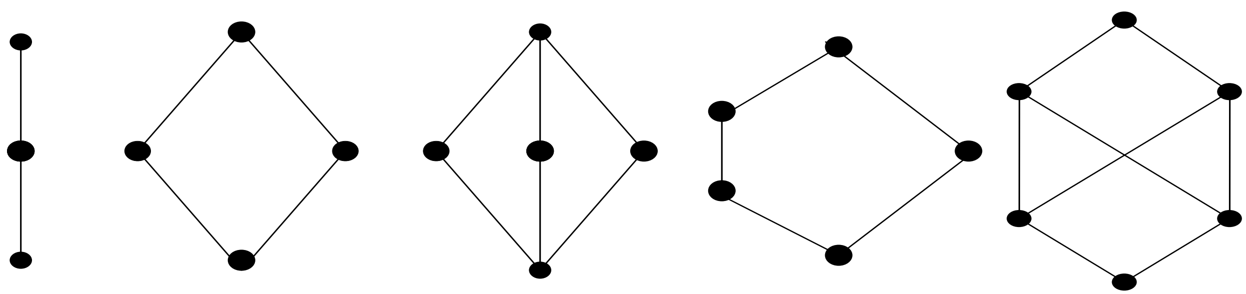
A *lattice* $\mathcal{L}$ is a partial order in which any $a, b \in \mathcal{L}$ have

- a least common upper bound (join) $a \sqcup b$, and
- a greatest common lower bound (meet) $a \sqcap b$



A *chain* is a totally ordered lattice.

A lattice is called *distributive* iff it can be represented by subsets of a set, with lattice operations interpreted as union and intersection.

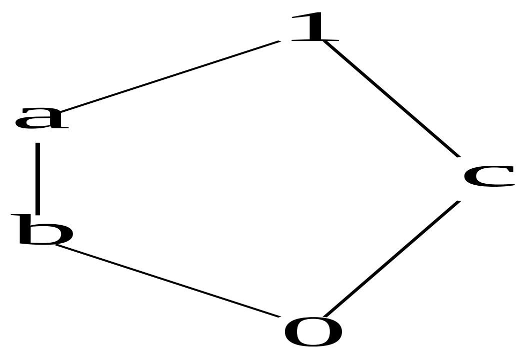


Supermodular functions/predicates

Let $\mathcal{L}$ be a lattice order on D. We say that an n-ary function $f : D^n \rightarrow \mathbb{R}$ is *supermodular on* $\mathcal{L}$ if

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

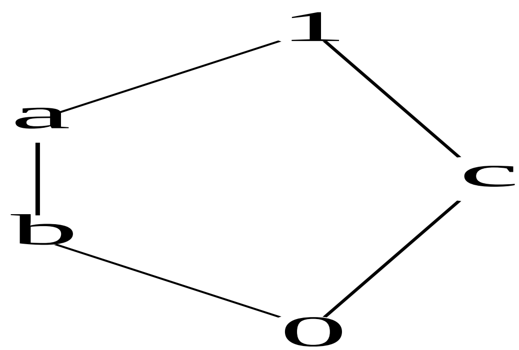
where $\sqcup$ and $\sqcap$ act point-wise.



$$f(a) = f(b) = f(c) = 1$$

$$f(0) = f(1) = 0$$

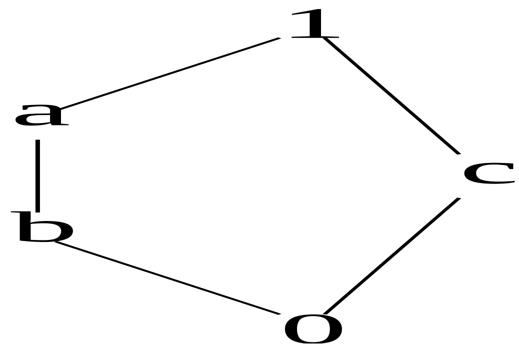
$$f(a) + f(c) = 2 \not\leq f(a \sqcap c) + f(a \sqcup b) = f(0) + f(1) = 0$$



$$f(a) = f(b) = 1$$

$$f(c) = f(0) = f(1) = 0$$

$$f(a) + f(c) = 1 \not\leq f(a \sqcap c) + f(a \sqcup b) = f(0) + f(1) = 0$$



$$f(a) = f(b) = f(0) = f(1) = 1$$

$$f(c) = 0$$

$$x, y \in \{a, b, 0, 1\}$$

$$f(x) + f(c) = 1 \leq f(x \sqcap c) + f(x \sqcup c) = f(0) + f(1) = 2$$

$$f(x) + f(y) = 2 \leq f(x \sqcap y) + f(x \sqcup y) = f(x) + f(y) = 2$$

$$f(c) + f(c) = 0 \leq f(c \sqcap c) + f(c \sqcup c) = f(c) + f(c) = 0$$

More examples

Every 2-monotone predicate is supermodular on $0 \to 1$.

Every unary predicate is supermodular on every chain.

Max CSP and supermodularity

Fact. If f_1 and f_2 are supermodular predicates on $\mathcal{L}$, then $\alpha \cdot f_1 + \beta \cdot f_2$, $\alpha, \beta \geq 0$, is supermodular on $\mathcal{L}$.

Theorem. [Schrijver]
Let $\mathcal{L}$ be a distributive lattice order on a finite set D. A function $f : D^n \to \mathbb{R}$ that is supermodular on $\mathcal{L}$ can be maximized in polynomial time, if f and $\mathcal{L}$ satisfy some mild restrictions.

Let $\alpha_1, \ldots, \alpha_m \geq 0$. If predicates $f_1, \ldots, f_m$ are supermodular on $\mathcal{L}$, then so is

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

Max CSP

A collection $C = \{f_1(\mathbf{x}_1), \ldots, f_m(\mathbf{x}_m)\}$ of constraints over $V = \{x_1, \ldots, x_n\}$;

each constraint $f_i(\mathbf{x}_i)$ has a weight $\alpha_i \in \mathbb{N}$.

Find an *assignment* $\phi : V \rightarrow D$ that maximizes the total weight of satisfied constraints; in other words, maximize the function $f : D^n \rightarrow \mathbb{N}$, defined by

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

Theorem. [Cohen, Cooper, Jeavons, Krokhin]
If $\mathcal{L}$ is a distributive lattice and Γ consists of supermodular predicates on $\mathcal{L}$, then Max CSP(Γ) is in **PO**.

Strict implementations

Definition. Let $Y = \{y_1, \ldots, y_m\}$ and $Z = \{z_1, \ldots, z_n\}$ be two disjoint sets of variables. Let $g_1(\mathbf{y}_1), \ldots, g_s(\mathbf{y}_s)$, $s > 0$, be constraints over $Y \cup Z$. If $g(y_1, \ldots, y_m)$ is a predicate such that the equality

$$g(y_1, \ldots, y_m) = \max_Z \sum_{i=1}^{s} g_i(\mathbf{y}_i) - \alpha$$

is satisfied for all $y_1, \ldots, y_m$, and some fixed $\alpha > 0$, then g is said to be *strictly implemented* from $\{g_1, \ldots, g_s\}$.

Lemma. If a predicate g can be strictly implemented from Γ and Max CSP($\Gamma \cup \{g\}$) is **APX**-complete then so is Max CSP(Γ)

How to strictly implement eq_2 with neq_2:

$$eq_2(x,y) \;=\; \max_z(neq_2(x,z) + neq_2(y,z)) - 1$$

If $x = y = 1$, then let $z = 0$. Result: 1

If $x = y = 0$, then let $z = 1$. Result: 1

If $x \neq y$, then let $z = 0$ (or $z = 1$). Result: 0

Cores

Definition. An *endomorphism* of Γ is a unary operation γ on D such that

$$f(a_1, \ldots, a_m) = 1 \Rightarrow f(\gamma(a_1), \ldots, \gamma(a_m)) = 1$$

for all $f \in \Gamma$ and all $(a_1, \ldots, a_m) \in D^m$. We will say that Γ is a *core* if every endomorphism of Γ is injective (i.e. a permutation).

Intuition. If Γ is *not* a core then Max CSP(Γ) reduces to a similar problem over a smaller domain obtained by removing elements *not* in $image(\gamma)$.

Fact. For $|D| = 2$, Γ is not a core iff there is $a \in D$ such that $f(a, \ldots, a) = 1$ for all $f \in \Gamma$. In this case Max CSP(Γ) is trivial.

Classification when $|D| = 2$ (version 2)

Let Γ be a constraint language over $\{0, 1\}$ and assume (without loss of generality) that Γ is a core. Then, Max CSP(Γ) $\in$ **PO** if and only if Γ is supermodular on $0 \to 1$. Otherwise, Γ is **APX**-complete.

Results an   pen questi ns

- $|D| = 3$

- Non-distributive lattices

- Single-predicate Max CSP

- Constraint languages that contain all constants

Classification when $|D| = 3$

[Jonsson, Klasson, Krokhin]

Let Γ be a constraint language over $\{0, 1, 2\}$ and assume (without loss of generality) that Γ is a core. Then, Max CSP(Γ) $\in$ **PO** if and only if Γ is supermodular on some chain over $\{0, 1, 2\}$. Otherwise, Γ is **APX**-complete.

The proof has many similarities with the proof for constraint languages with all constants.

pen question:

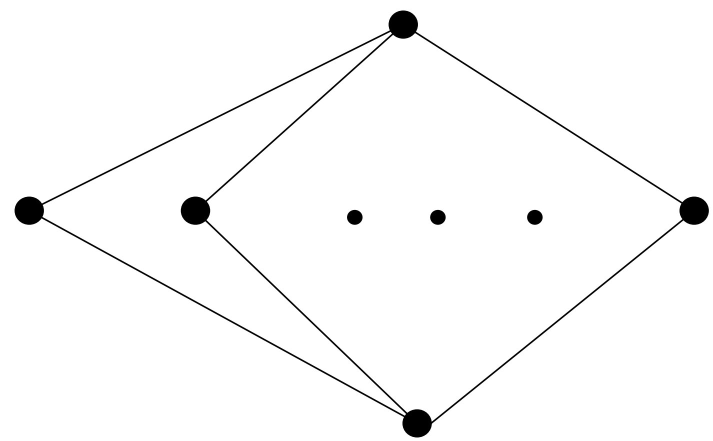
What is the complexity/approximability of Max CSP(Γ) when $|D| > 3$?

Hypothesis

Classification when $|D| = k > 3$

Let Γ be a constraint language over $\{0, \ldots, k-1\}$ and assume that Γ is a core. Then, Max CSP(Γ) $\in$ **PO** if and only if Γ is supermodular on some distributive lattice over $\{0, \ldots, k-1\}$. Otherwise, Γ is **APX**-complete.

There exist constraint languages Γ that are supermodular on



but not on any distributive lattice [Krokhin, Larose].

Theorem. [Krokhin, Larose]
If Γ consists of predicates that are supermodular on the k-diamond, then Max CSP(Γ) is in **PO**.

The algorithm runs in $O(n^3)$ and it is inspired by algorithms for the Min Cut/Max Flow problem.

If $\mathbf{V}$,$\mathbf{W}$ are classes of lattices, then $\mathbf{V} \circ \mathbf{W}$ consists of all lattices $\mathcal{L}$ such that there is a congruence θ on $\mathcal{L}$ with the following properties:

- the congruence lattice $\mathcal{L}/\theta \in \mathbf{W}$; and
- every θ-class is a lattice in $\mathbf{V}$

Theorem. [Krokhin, Larose]
Suppose that $\mathbf{V}$,$\mathbf{W}$ are finite classes of finite lattices. If supermodular optimization over $\mathbf{V}$ and $\mathbf{W}$ is in $\mathbf{PO}$, then supermodular optimization over $\mathbf{V} \circ \mathbf{W}$ is in $\mathbf{PO}$, too.

Corollary.
If Γ consists of predicates that are supermodular on the pentagon, then Max CSP(Γ) is in $\mathbf{PO}$.

Let Γ be a core.

pen question:

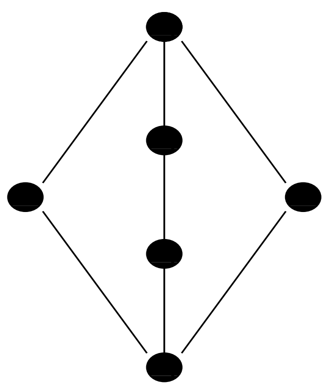
Is Max CSP(Γ) $\in$ **PO** whenever Γ is supermodular on *some* lattice?

pen question:

Is Max CSP(Γ) **APX**-complete whenever Γ is not supermodular on *any* lattice?

pen question:

Assume that Γ is supermodular on



Is Max CSP(Γ) in **PO**?

Complexity of single-predicate Max CSP

[Jonsson, Krokhin]

Let $f : D^n \to \{0, 1\}$ be a predicate such that $n > 1$.
Max CSP($\{f\}$) is in **PO** if and only if there exists a $d \in D$ such that $f(d, \ldots, d) = 1$. Otherwise, Max CSP($\{f\}$) is **NP**-complete.

This is proved by two induction proofs. In the first part, it is assumed that f is binary and the induction is over $|D|$; cores play an important rôle in the proof. In the second part, the induction is over the arity of f; the main idea is to construct strict implementations that reduce the arity of predicates.

pen question:

Is Max CSP($\{f\}$) **APX**-complete whenever Max CSP($\{f\}$) is **NP**-complete?

Constraint languages containing all constants

Given a finite set D', we define the predicate $u_{D'}$ such that

$$u_{D'}(x) = 1 \iff x \in D'.$$

Let Γ be a constraint language over domain $D = \{0, \ldots, d-1\}$.
Γ contains all constants if $\{u_{\{0\}}, \ldots, u_{\{d-1\}}\} \subseteq \Gamma$.

Note: Γ is a core (the identity is the only endomorphism).

Theorem. [Deineko, Jonsson, Klasson, Krokhin]
Let Γ be a constraint language that contains all constants. Then, Max CSP(Γ) $\in$ **PO** if and only if Γ is supermodular on some chain. Otherwise, Max CSP(Γ) is **APX**-complete.

Every chain is a distributive lattice so we only need to prove the hardness part: Consequently, we assume that Γ is not supermodular on any chain over D.

Step 1. For every $D' \subseteq D$, the predicate $u_{D'}$ can be strictly implemented by Γ. Henceforth, we assume that all unary predicates are in Γ.

Step 2. Γ contains all unary predicates. Then, Γ can strictly implement a constraint language Γ′ such that Γ is not supermodular on any chain and every predicate in Γ′ is at most binary. [Burkard, Klinz, Rudolf]

Step 3. If Γ is not supermodular on any chain, then there exists $D' \subseteq D$ such that

- $|D'| \leq 4$; and
- $\Gamma|_{D'}$ is not supermodular on any chain.

The proof is inspired by how the COM-algorithm works [Deineko, Rudolf, Woeginger].

Is there an AP-reduction from Max CSP($\Gamma|_{D'}$) to Max CSP(Γ)?

However:

Max CSP($\Gamma|_{D'}$)-B AP-reduces to Max CSP(Γ)-B.

Strict implementations increase the degrees of variables, but not too much.

Step 4. If Γ is not supermodular on any chain, then there exists a subset $\Gamma' \subseteq \Gamma$ such that

- $|\Gamma'| \leq 3$; and
- Γ' is not supermodular on any chain.

By steps 1-4, we now have a constraint language Γ' satisfying the following properties:

- $\Gamma' = \{f_1, f_2, f_3\}$ where $f_i : \{0, 1, 2, 3\}^2 \to \{0, 1\}$;

- Γ' is not supermodular on any chain;

- Max CSP(Γ')-B AP-reduces to Max CSP(Γ)-B.

By a computer-generated enumeration of strict implementations, it turns out that some predicate $\neq_E$ with $|E| = 2$ can be strictly implemented by every possible Γ'.

It is known that Max CSP($\neq_E$)-3 is **APX**-complete [Alimonti, Kann] which concludes the proof.

pen question

Is there an elegant way of proving the previous result without using computer-assisted case analyses?

References: Max CSP and lattices.

[1] Nadia Creignou. A dichotomy theorem for maximum generalized satisfiability. *Journal of Computer and System Science* 51(3):511-522, 1995.

[2] Sanjeev Khanna, Madhu Sudan, Luca Trevisan, and David Williamson. The approximability of constraint satisfaction problems. *SIAM Journal on Computing* 30(6):1863-1920, 2000.

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

[4] David Cohen, Martin Cooper, Peter Jeavons, and Andrei Krokhin. Supermodular functions and the complexity of Max CSP. *Discrete Applied Mathematics* 149(1–3):53-72, 2005.

[5] Andrei Krokhin and Benoit Larose. Maximum constraint satisfaction on diamonds. Proc. CP-2005, 388-402, 2005.

[6] Vladimir Deineko, Peter Jonsson, Mikael Klasson, and Andrei Krokhin. Supermodularity on chains and complexity of maximum constraint satisfaction problems. In *Proceedings of the European Conference on Combinatorics, Graph Theory and Applications (EuroCOMB-2005)*, pp. 51–56, 2005. Longer version available from `www.arxiv.org/ps/cs.CC/0602075`.

[7] Peter Jonsson, Mikael Klasson, and Andrei Krokhin. The approximability of three-valued MAX CSP. *SIAM Journal on Computing* 35(3):1329-1349, 2006. Draft version available from `www.arxiv.org/ps/cs.CC/0412042`.