CSPs with Few Alien Constraints

Peter Jonsson (Linköping)
Victor Lagerkvist (Linköping)
George Osipov (Oxford + Royal Holloway + Linköping)

- Alien Constraints
- Motivation
- Finite-domain CSPs
- Equality Languages
- Future Work

Alien Constraints

CSP($\mathcal{A}$)

CSP($\mathcal{A}$)

3-Colourability problem: CSP($\mathcal{A}$) with $\mathcal{A} = \{\neq_3\}$

CSP($\mathcal{A}$)

3-Colourability problem: CSP($\mathcal{A}$) with $\mathcal{A} = \{\neq_3\}$

$\neq_3 = \{(1,0),(0,1),(1,2),(2,1),(0,2),(2,0)\}$

$\mathsf{CSP}(\mathcal{A} \cup \mathcal{B})$

CSP($\mathcal{A} \cup \mathcal{B}$)

$\mathcal{A}$ contains *background* relations

$\mathcal{B}$ contains *alien* relations

$\mathsf{CSP}(\mathcal{A} \cup \mathcal{B})$

$\mathcal{A}$ contains *background* relations

$\mathcal{B}$ contains *alien* relations

$\mathsf{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$

$\mathrm{CSP}(\mathcal{A} \cup \mathcal{B})$

$\mathcal{A}$ contains *background* relations

$\mathcal{B}$ contains *alien* relations

$\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$

Instances contain at most k $\mathcal{B}$-constraints

Basic assumption:

(1) CSP($\mathcal{A}$) is polynomial-time solvable
(2) CSP($\mathcal{A} \cup \mathcal{B}$) is NP-hard

Basic assumption:

(1) CSP($\mathcal{A}$) is polynomial-time solvable
(2) CSP($\mathcal{A} \cup \mathcal{B}$) is NP-hard

Basic question: What is the largest k such that $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable?

Motivation

Motivation I

The polynomial-time solvable fragments of CSP($\mathcal{A}$) are known in many cases.

- Finite domains
- Allen's algebra
- Equality languages
- Temporal relations
- ...

Motivation I

The polynomial-time solvable fragments of CSP($\mathcal{A}$) are known in many cases.

- Finite domains
- Allen's algebra
- Equality languages
- Temporal relations
- ...

Quite difficult to work with!

Adding a small number of relations outside $\mathcal{A}$ can be really helpful $\longrightarrow$ $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$

We illustrate this idea with model checking, but there are many other examples. Global constraints (e.g. the all-diff constraint) is one of them.

Motivation I

Typically, there are two possibilities.

(1) $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable for every fixed k, or

(2) there exists a fixed k such that $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is NP-hard.

Motivation I

Typically, there are two possibilities.

(1) $\text{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable for every fixed k, or

(2) there exists a fixed k such that $\text{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is NP-hard.

Case (2) is almost always bad.

Case (1) is better, and sometimes *much* better.

Motivation I

Case (1a) $f(k) \cdot poly(||I||)$

Case (1b) $||I||^{f(k)}$

Motivation I

Case (1a) $f(k) \cdot poly(||I||)$

Case (1b) $||I||^{f(k)}$

Parameterized complexity.

Case (1a) $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is *fixed-parameter tractable* (FPT) with parameter k.

Case (1b) $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is in XP.

Case (2) $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ is pNP-hard.

Motivation II

Redundant($\mathcal{A}$)
"Can one remove constraint c without changing the set of solutions?"

Impl($\mathcal{A}$)
"Is the set of solutions to I_1 a subset of the solutions to I_2?"

Equiv($\mathcal{A}$)
"Do I_1 and I_2 have the same set of solutions?"

Motivation II

Redundant($\mathcal{A}$)
"Can one remove constraint c without changing the set of solutions?"

Impl($\mathcal{A}$)
"Is the set of solutions to I_1 a subset of the solutions to I_2?"

Equiv($\mathcal{A}$)
"Do I_1 and I_2 have the same set of solutions?"

Many applications are described in the literature.

Complexity classifications are known in special cases (e.g. Boolean domains).

Motivation II

These three problems are polynomial-time interreducible.

Motivation II

These three problems are polynomial-time interreducible.

We show that the complexity of them can be described in terms of $\mathrm{CSP}_{\leq 1}(\mathcal{A} \cup \mathcal{B})$ for suitably chosen $\mathcal{B}$.

Motivation III

"Horn-like" disjunctive constraints have many applications.

$(x \vee \neg y \vee \neg z) \wedge (y \vee \neg w) \wedge (z)$

Motivation III

"Horn-like" disjunctive constraints have many applications.

$(x \vee \neg y \vee \neg z) \wedge (y \vee \neg w) \wedge (z)$

The computational complexity of such constraints is known.

CSP($\mathcal{A} \vee \mathcal{B}^*$) is in P

if and only if

(1) $\mathcal{A}$ and $\mathcal{B}$ satisfy an algebraic condition (*1-independence*) and
(2) $\text{CSP}_{\leq 1}(\mathcal{A} \cup \mathcal{B})$ is in P.

Finite-domain CSPs

Theorem. Let $\mathcal{A}, \mathcal{B}$ be constraint languages over a finite domain A. Then, $\mathrm{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is either in FPT or pNP-hard.

Theorem. Let $\mathcal{A}, \mathcal{B}$ be constraint languages over a finite domain A. Then, $\mathrm{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is either in FPT or pNP-hard.

The proof is based on ideas from the universal-algebraic approach to CSPs.

Theorem. Let $\mathcal{A}, \mathcal{B}$ be constraint languages over a finite domain A. Then, $\mathsf{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is either in FPT or pNP-hard.

The proof is based on ideas from the universal-algebraic approach to CSPs.

Warning: This does not imply that $\mathsf{CSP}_{\leq 1}(\mathcal{A} \cup \mathcal{B})$ is NP-hard whenever $\mathsf{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is pNP-hard.

Thus, we cannot say much about problems such as Redundant($\mathcal{A}$), Impl($\mathcal{A}$), and Equiv($\mathcal{A}$).

Theorem. Let $\mathcal{A}, \mathcal{B}$ be constraint languages over the two-element domain $\{0,1\}$. Assume CSP($\mathcal{A}$) is polynomial-time solvable and CSP($\mathcal{A} \cup \mathcal{B}$) is NP-hard. Then the following hold.

(1) If $\mathcal{A}$ is Schaefer, then $\text{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is in FPT.

(2) If (i) $\mathcal{A}$ is not Schaefer, (ii) $\mathcal{A}$ is both 0- and 1-valid, (iii) $\mathcal{B}$ contains a 0/1-pair, and (iv) $\mathcal{B}$ is 0- or 1-valid, then $\text{CSP}_{\leq 2}(\mathcal{A} \cup \mathcal{B})$ is NP-hard and $\text{CSP}_{\leq 1}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable.

(3) Otherwise, $\text{CSP}_{\leq 2}(\mathcal{A} \cup \mathcal{B})$ is NP-hard.

Theorem. Let $\mathcal{A}, \mathcal{B}$ be constraint languages over the two-element domain $\{0,1\}$. Assume CSP($\mathcal{A}$) is polynomial-time solvable and CSP($\mathcal{A} \cup \mathcal{B}$) is NP-hard. Then the following hold.

(1) If $\mathcal{A}$ is Schaefer, then $\text{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is in FPT.

(2) If (i) $\mathcal{A}$ is not Schaefer, (ii) $\mathcal{A}$ is both 0- and 1-valid, (iii) $\mathcal{B}$ contains a 0/1-pair, and (iv) $\mathcal{B}$ is 0- or 1-valid, then $\text{CSP}_{\leq 2}(\mathcal{A} \cup \mathcal{B})$ is NP-hard and $\text{CSP}_{\leq 1}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable.

(3) Otherwise, $\text{CSP}_{\leq 2}(\mathcal{A} \cup \mathcal{B})$ is NP-hard.

The proof is based on a Schaefer-like analysis of the relations (not so much algebra).

Corollary. Let $\mathcal{A}$ be a language over a two-element domain. Then Redundant($\mathcal{A}$), Impl($\mathcal{A}$), and Equiv($\mathcal{A}$) are either polynomial-time solvable or NP-hard.

Equality Languages

A constraint language $\mathcal{A}$ is an *equality language* if the relations in $\mathcal{A}$ are first-order definable in $(\mathbb{N}; =)$. N.B. Infinite domain!

A constraint language $\mathcal{A}$ is an *equality language* if the relations in $\mathcal{A}$ are first-order definable in $(\mathbb{N}; =)$. N.B. Infinite domain!

$$R(x, y, z) \equiv (x = y) \vee (x = z)$$

$$S(x, y, z) \equiv (x = y \vee x \neq z) \wedge (y = z \vee x \neq y)$$

A constraint language $\mathcal{A}$ is an *equality language* if the relations in $\mathcal{A}$ are first-order definable in $(\mathbb{N}; =)$. N.B. Infinite domain!

$$R(x, y, z) \equiv (x = y) \vee (x = z)$$

$$S(x, y, z) \equiv (x = y \vee x \neq z) \wedge (y = z \vee x \neq y)$$

The complexity of CSP($\mathcal{A}$) is known for all equality languages $\mathcal{A}$.

The complexity of equality languages is a necessary ingredient in *all* classifications of more expressive classes.

Thus, natural to study $\mathrm{CSP}_{\leq k}(\mathcal{A} \cup \mathcal{B})$ for equality languages $\mathcal{A}, \mathcal{B}$.

Theorem. Let $\mathcal{A}, \mathcal{B}$ be equality languages such that $\mathrm{CSP}(\mathcal{A})$ is polynomial-time solvable and $\mathrm{CSP}(\mathcal{A} \cup \mathcal{B})$ is NP-hard.

(1) If $\mathcal{A}$ is Horn, $\mathrm{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is in FPT.

(2) If $\mathcal{A}$ is not Horn, $\mathrm{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is pNP-hard. Moreover, there exists an integer $c = c(\mathcal{A})$ such that $\mathrm{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable whenever $\neq_c \notin \langle \mathcal{A} \cup \mathcal{B} \rangle_{\leq k}$, and is NP-hard otherwise.

Theorem. Let $\mathcal{A}, \mathcal{B}$ be equality languages such that CSP($\mathcal{A}$) is polynomial-time solvable and CSP($\mathcal{A} \cup \mathcal{B}$) is NP-hard.

(1) If $\mathcal{A}$ is Horn, $\text{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is in FPT.

(2) If $\mathcal{A}$ is not Horn, $\text{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is pNP-hard. Moreover, there exists an integer $c = c(\mathcal{A})$ such that $\text{CSP}_{\leq}(\mathcal{A} \cup \mathcal{B})$ is polynomial-time solvable whenever $\neq_c \notin \langle \mathcal{A} \cup \mathcal{B} \rangle_{\leq k}$, and is NP-hard otherwise.

The proof of based on the universal-algebraic approach combined with a recent complexity classification of MinCSP for equality constraints.

Corollary. Let $\mathcal{A}$ be an equality language. Then Redundant($\mathcal{A}$), Impl($\mathcal{A}$), and Equiv($\mathcal{A}$) are either polynomial-time solvable or NP-hard.

Future Work

Refined classification for finite domains (that covers Redundant($\mathcal{A}$), Impl($\mathcal{A}$), and Equiv($\mathcal{A}$)).

Classification for other infinite-domain CSPs.

Improved algebraic toolbox for alien constraints.

Are there $\mathcal{A}$ and $\mathcal{B}$ such that $\text{CSP}_{\leq}$ is in XP but not in FPT?

The alien constraints framework is one way of expanding the concept of "tractable CSP". Other ways?