A theory of gadget reductions for CSP

Andrei Krokhin

Durham University

Useful surveys

Much of what is covered in this lecture can be found in survey

- Polymorphisms, and how to use them.
  L. Barto, A. Krokhin, and R. Willard.

It is written specifically for those without algebra background!

See also the full (open-access) volume of surveys:

- The Constraint Satisfaction Problem: Complexity and Approximability. Editors: A. Krokhin and S. Živný.
  Dagstuhl Follow-Ups series, Volume 7, 2017.
  http://drops.dagstuhl.de/portals/dfu/index.php?semnr=16027

Constraint Satisfaction Problem (CSP)

Fix finite relational structure $\mathbb{A} = (A; R_1, \ldots, R_n)$ (aka constraint language) where each $R_i \subseteq A^{k_i}$ or $R_i : A^{k_i} \to \{\text{true}, \text{false}\}$.

Definition

An instance of $\mathrm{CSP}(\mathbb{A})$ is a list of constraints over vars V, e.g.

$$R_1(x, y, z), R_1(z, y, w), R_2(z), R_3(x, w), R_3(y, y)$$

where each R_i is from $\mathbb{A}$.

Question: Is there $s : V \to A$ satisfying all constraints?

Many other variants, e.g.:

- infinite A (but the instance is still finite)
- nothing (or something other than relations) is fixed
- real-valued functions instead of relations (for optimisation)
- many questions other than plain satisfiability

Examples, a conjecture, and a theorem

- k-Col: $\mathbb{A} = ([k]; \{\neq\})$
- k-NAE: $\mathbb{A} = ([k]; \{(a,b,c) \in [k]^3 \mid a \neq b \vee a \neq c \vee b \neq c\})$
  — (essentially) k-colouring for 3-uniform hypergraphs
- 3-Sat: $\mathbb{A} = (\{0,1\}; (x \vee y \vee z), (x \vee y \vee \neg z), \ldots)$
- Horn 3-Sat: as above, each clause has ≤ 1 unneg var
- 3-Lin$p$: $\mathbb{A} = (\mathbb{Z}_p;\ x+y+z=0,\ x+2y+3z=7, \ldots)$
- Unique Games-k: $\mathbb{A} = ([k]; \{(a, \pi(a)) \mid a \in [k]\})$ where π runs through all permutations on $[k]$.

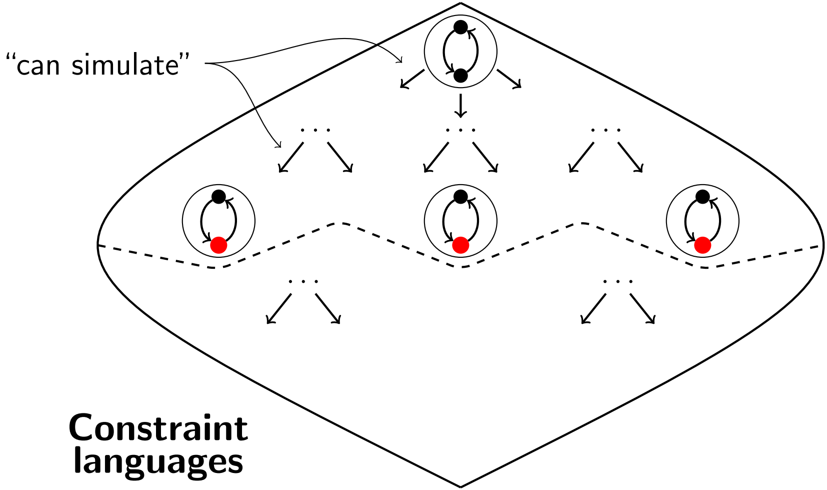
Conjecture (CSP Dichotomy Conjecture, Feder-Vardi'98)

Every $\mathrm{CSP}(\mathbb{A})$ is either in **P** *or* **NP***-complete.*

Theorem (Bulatov'17; Zhuk'17)

The above conjecture is true.

A theory of structured reductions for CSP (high-level view)



Simulation via gadgets

Three (increasingly more general) levels of simulation:

1. primitive positive (pp) definitions (= gadgets, same domain)

   Ex.: Let $\mathbb{A}_1 = (A; R)$, R ternary, and $\mathbb{A}_2 = (A; T, S)$ be s.t.

   $T(x) = \exists w\ R(x, w, x)$, $S(x, y) = \exists w\ R(x, y, w) \wedge R(w, y, x)$.

   Then an instance of $\mathrm{CSP}(\mathbb{A}_2)$, say

   $$T(y), S(x, y), S(z, x)$$

   can be re-written as an instance of $\mathrm{CSP}(\mathbb{A}_1)$

   $$R(y, w_1, y),\ R(x, y, w_2), R(w_2, y, x),\ R(z, x, w_3), R(w_3, x, z)$$

2. pp-interpretations (gadgets, possible change of domain)
3. pp-constructions (the above + use of constants)

Gadget reductions

- Take any $2\text{-}\textsc{Sat}$ instance $(x \vee \overline{y}) \wedge (y \vee \overline{z}) \wedge (y \vee \overline{u}) \wedge (x \vee u)$
- Let R be the set of solutions to it, projected to $\{y, z, u\}$, $R = \{(1, *, *), (0, 0, 0)\}$, and let $C_0 = \{(0)\}$, $C_1 = \{(1)\}$.
- If $\mathbb{A} = (\{0, 1\}; R, C_0, C_1)$ then $\mathrm{CSP}(\mathbb{A})$ reduces to $2\text{-}\textsc{Sat}$.
- If I is an instance of $\mathrm{CSP}(\mathbb{A})$,
  - replace each $R(y, z, u)$ constraint by the above 4 clauses (using fresh x each time)
  - replace each $C_0(w)$ by $(\overline{w} \vee \overline{w})$ and each $C_1(w)$ by $(w \vee w)$.

For the above reduction to work,

- we don't care how big the gadgets are or even what they are
- we only need to know that they exist.

Polymorphisms by example

- Take any 2-SAT instance $(x \vee \overline{y}) \wedge (y \vee \overline{z}) \wedge (y \vee \overline{u}) \wedge (x \vee u)$
- Take any three solutions $\mathbf{a}, \mathbf{b}, \mathbf{c}$ to this instance
- Apply the ternary *majority* operation m to $\mathbf{a}, \mathbf{b}, \mathbf{c}$ coordinate-wise (variables ordered here as x, y, z, u)

$$
\begin{array}{rcccccc}
 & & m & m & m & m & & \\
 & & \downarrow & \downarrow & \downarrow & \downarrow & & \\
\mathbf{a} = & (& 1 & 1 & 1 & 0 &) & \text{sat} \\
\mathbf{b} = & (& 1 & 1 & 0 & 1 &) & \text{sat} \\
\mathbf{c} = & (& 1 & 0 & 0 & 0 &) & \text{sat} \\
\hline
m(\mathbf{a}, \mathbf{b}, \mathbf{c}) = & (& 1 & 1 & 0 & 0 &) & \text{sat}
\end{array}
$$

Polymorphisms

An operation $f : A^m \to A$ is called a polymorphism of a k-ary relation $R \subseteq A^k$ if, for any $m \times k$ matrix with rows in R,

$$
\begin{array}{ccccccc}
 & f & & f & & f & \\
 & \downarrow & & \downarrow & & \downarrow & \\
(& a_{11} & , & \ldots & , & a_{1k} &) \in R \\
 & \vdots & & \vdots & & \vdots & \vdots \\
(& a_{m1} & , & \ldots & , & a_{mk} &) \in R \\
\hline
 & & & & & & \Downarrow \\
(& f(a_{11},\ldots,a_{m1}) & , & \ldots & , & f(a_{1k},\ldots,a_{mk}) &) \in R
\end{array}
$$

Call f a polymorphism of $\mathbb{A}$ if it is such for all R in $\mathbb{A}$.
Notation: $\mathrm{Pol}(\mathbb{A})$.

More examples of polymorphisms

1. Consider $R(x, y, z) = \bar{x} \vee \bar{y} \vee z$ and binary ops max and min

$$\begin{array}{ccc|l} 1 & 0 & 0 & \in R \\ 0 & 1 & 0 & \in R \\ \hline 1 & 1 & 0 & \notin R \end{array} \qquad \begin{array}{ccc|l} ? & ? & ? & \in R \\ ? & ? & ? & \in R \\ \hline 1 & 1 & 0 & \notin R \end{array}$$

2. Every polymorphism of 3-SAT is a projection (aka dictator), i.e. $f(x_1, \ldots, x_n) = x_i$ for some i.
3. Every polymorphism of 3-Col is of the form $f(x_1, \ldots, x_n) = \pi(x_i)$ for some $i \le n$ and permutation π
4. If $\mathbb{A} = (A, E)$ is a digraph then f is a polymorphism of $\mathbb{A}$ if $(a_1, b_1), \ldots, (a_n, b_n) \in E \Rightarrow (f(a_1, \ldots, a_n), f(b_1, \ldots, b_n)) \in E$. In other words, f is simply a homomorphism from $\mathbb{A}^n$ to $\mathbb{A}$.

A Galois connection

Notation:

- Let $\langle \mathbb{A} \rangle_{pp}$ be the set of all relations pp-definable in $\mathbb{A}$ (and $=$).
- $\mathrm{Pol}(\mathbb{A})$ is the set of all polymorphisms of $\mathbb{A}$.
- For a set C of operations on A, $\mathrm{Inv}(C) = \{R \mid C \subseteq \mathrm{Pol}(R)\}$

Theorem (Geiger'68; Bodnarchuk et al.'69)

For every $\mathbb{A}$, we have $\langle \mathbb{A} \rangle_{pp} = \mathrm{Inv}(\mathrm{Pol}(\mathbb{A}))$.

In words, "pp-definable in $\mathbb{A}$" $=$ "breaks no polymorphisms of $\mathbb{A}$".

- Polymorphisms of $\mathbb{A}$ precisely control what $\mathbb{A}$ can pp-define.

Clones

For any $\mathbb{A}$, $\mathrm{Pol}(\mathbb{A})$ is a clone.

Clone = set C of multivariate functions on a set A such that

1. C is closed under composition, and
2. C contains all projections/dictators ($f(x_1, \ldots, x_n) = x_i$)

Examples of clones:

- *trivial* clone $\mathcal{T}$, consisting of all projections.
- all linear functions (wrt some fixed ring)
- all monotone functions (wrt some fixed partial order)

- When $|A| = 2$, all clones have been described [Post'1921].
- For $|A| > 2$, there is no hope to get a complete description.

How to find gadgets (even though you don't have to)

Natural questions: Given a structure $\mathbb{A}$ and a relation R_0:

1. How do you check whether $R_0 \in \langle \mathbb{A} \rangle_{pp}$?
2. If this holds, how do you find an actual gadget?

Answer: There is a generic way, via polymorphisms.

- $R_0 \in \langle \mathbb{A} \rangle_{pp}$ iff R_0 is preserved by all $f \in \mathrm{Pol}(\mathbb{A})$ of arity $|R_0|$.
- There is an algorithm that solves both (1) and (2) [CJG'99]
  - it puts problem (1) in complexity class **coNEXPTIME**.
  - For Boolean CSPs, both (1) and (2) are **P** [Dalmau'00]
  - For some $d > 1$, (1) is **coNEXPTIME**-complete for $\mathbb{A}$ with a d-element domain [Willard'10].
  - The previous claim is open, if one fixes $\mathbb{A}$ (not just its domain).

The algorithm, by example

$\mathbb{A} = (\{0,1,2\}, \neq, C_0, C_1, C_2)$ and $R_0 = \{(0,1),(0,2),(1,1),(2,2)\}$.

Idea: represent polymorphisms $f \in \mathbb{A}$ of arity 4 as a 3^4-ary relation

$S = \{(f(0000), f(0001), f(0002), \ldots, f(2222)) \mid$ 4-ary $f \in \mathrm{Pol}(\mathbb{A})\}$.

We have $S \in \langle \mathbb{A} \rangle_{pp}$: one can define $S(x_{0000}, \ldots, x_{2222})$ as

$$\Big(\bigwedge_{i_1 \neq i_2, j_1 \neq j_2, k_1 \neq k_2, l_1 \neq l_2} x_{i_1 j_1 k_1 l_1} \neq x_{i_2 j_2 k_2 l_2} \Big) \wedge \Big(\bigwedge_i x_{iiii} = i \Big)$$

Now $\exists$-quantify all variables in S except x_{0012} and x_{1212}.
Call the obtained binary relation R'.

It is easy to show that $R_0 \subseteq R'$ and that $R' = R_0$ iff $R_0 \in \langle \mathbb{A} \rangle_{pp}$.

Simulation vs. polymorphisms

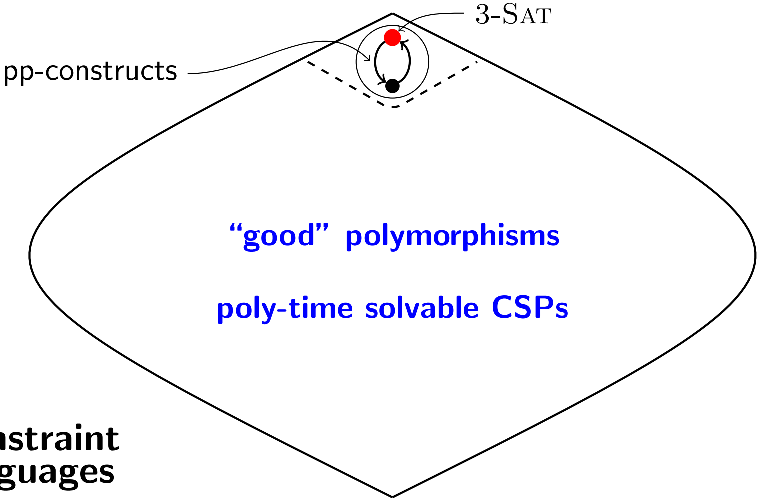
Theorem (Birkhoff'35; Geiger'68; Bodnarchuk et al.'69; Bodirsky; Willard; Barto, Opršal,Pinsker'18)

- $\mathbb{A}$ *pp-defines* $\mathbb{B}$ *iff* $\mathrm{Pol}(\mathbb{A}) \subseteq \mathrm{Pol}(\mathbb{B})$.
- $\mathbb{A}$ *pp-interprets* $\mathbb{B}$ *iff* $\mathrm{Pol}(\mathbb{A}) \to \mathrm{Pol}(\mathbb{B})$ *(homomorphism).*
- $\mathbb{A}$ *pp-constructs* $\mathbb{B}$ *iff* $\mathrm{Pol}(\mathbb{A}) \dashrightarrow \mathrm{Pol}(\mathbb{B})$ *(height-1 homo).*

Remarks:

- Proof constructive $\Rightarrow$ generic reduction $\mathrm{CSP}(\mathbb{B}) \rightsquigarrow \mathrm{CSP}(\mathbb{A})$
- $\xi : \mathrm{Pol}(\mathbb{A}) \to \mathrm{Pol}(\mathbb{B})$ iff it "preserves equations/identities"
  - This allows applications of deep structural universal algebra
- $\xi : \mathrm{Pol}(\mathbb{A}) \dashrightarrow \mathrm{Pol}(\mathbb{B})$ iff it "preserves ... of height 1"
  - Not used in resolving Dichotomy Conj, but very important

Algebraic dichotomy (picture not to scale)



Constraint languages

Negative and positive descriptions

Theorem

For any $\mathbb{A}$, TFAE:

1. $\mathbb{A}$ *does not pp-construct* 3-SAT *(or, equivalently,* 3-COL*)*
2. $\mathbb{A}$ *has a weak near-unanimity polym'm [Mároti,McKenzie'08]*

$$f(y,x,\ldots,x,x) = f(x,y,\ldots,x,x) = \ldots = f(x,x,\ldots,x,y)$$

3. $\mathbb{A}$ *has a cyclic polymorphism [Barto,Kozik'12]*

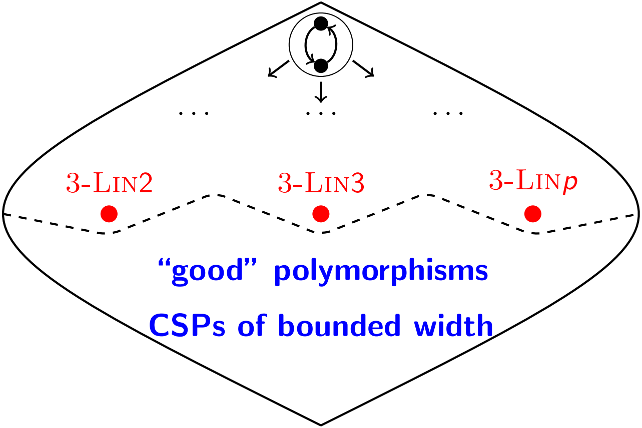
$$f(x_1,x_2,x_3,\ldots,x_n) = f(x_2,x_3,\ldots,x_n,x_1)$$

4. $\mathbb{A}$ *has a Siggers polymorphism [Siggers'09,KMM'14]*

$$f(r,a,r,e) = f(a,r,e,a)$$

Another picture for CSPs

When problems of the form 3-LINp are the key hard problems.



How to use "good" polymorphisms

"Good" polymorphisms imply "useful" structure in a CSP.

- Q: How to extract this structure and use it algorithmically?
- A: This varies from case to case.
  - — sometimes it's DIY, sometimes you need to call a specialist.

A very simple example: For each n, $\mathrm{Pol}(\mathbb{A})$ contains f_n (of arity n) such that $f(a_1, a_2, \ldots, a_n)$ depends only on $\{a_1, a_2, \ldots, a_n\}$.

$$
\begin{array}{cccccccc}
 & f_n & & & & f_n & & \\
 & \downarrow & & \downarrow & & \downarrow & & \\
(& a_{11} & , & \ldots & , & a_{1k} &) & \in R \\
 & \vdots & & \vdots & & \vdots & & \vdots \\
(& a_{n1} & , & \ldots & , & a_{nk} &) & \in R \\
\hline
 & & & & & & & \Downarrow \\
(& f_n(a_{11}, \ldots, a_{n1}) & , & \ldots & , & f_n(a_{1k}, \ldots, a_{nk}) &) & \in R
\end{array}
$$

CSPs and polymorphisms

1. **Decision CSP**: Can all constraints be satisfied?
2. **Counting CSP**: Count the number of solutions
3. **Max CSP**: Find a map satisfying max number of constraints
4. **Approx Max CSP**: Satisfy $c \times \mathrm{Opt}$ number of constraints
5. **Approx Min CSP**: assuming $1-\epsilon$ fraction of constraints can be satisfied, find a map satisfying $\geq 1-g(\epsilon)$ fraction.
6. **Promise CSP**: given a 3-col graph, find a 6-colouring for it

Each of the above has an appropriate notion of polymorphism

- lack of good polymorphisms $\Rightarrow$ hardness
- good polymorphisms $\Rightarrow$ efficient algorithms

A theory of structured reductions for CSP (high-level view)