A tutorial on Algebra and CSP, Part 2

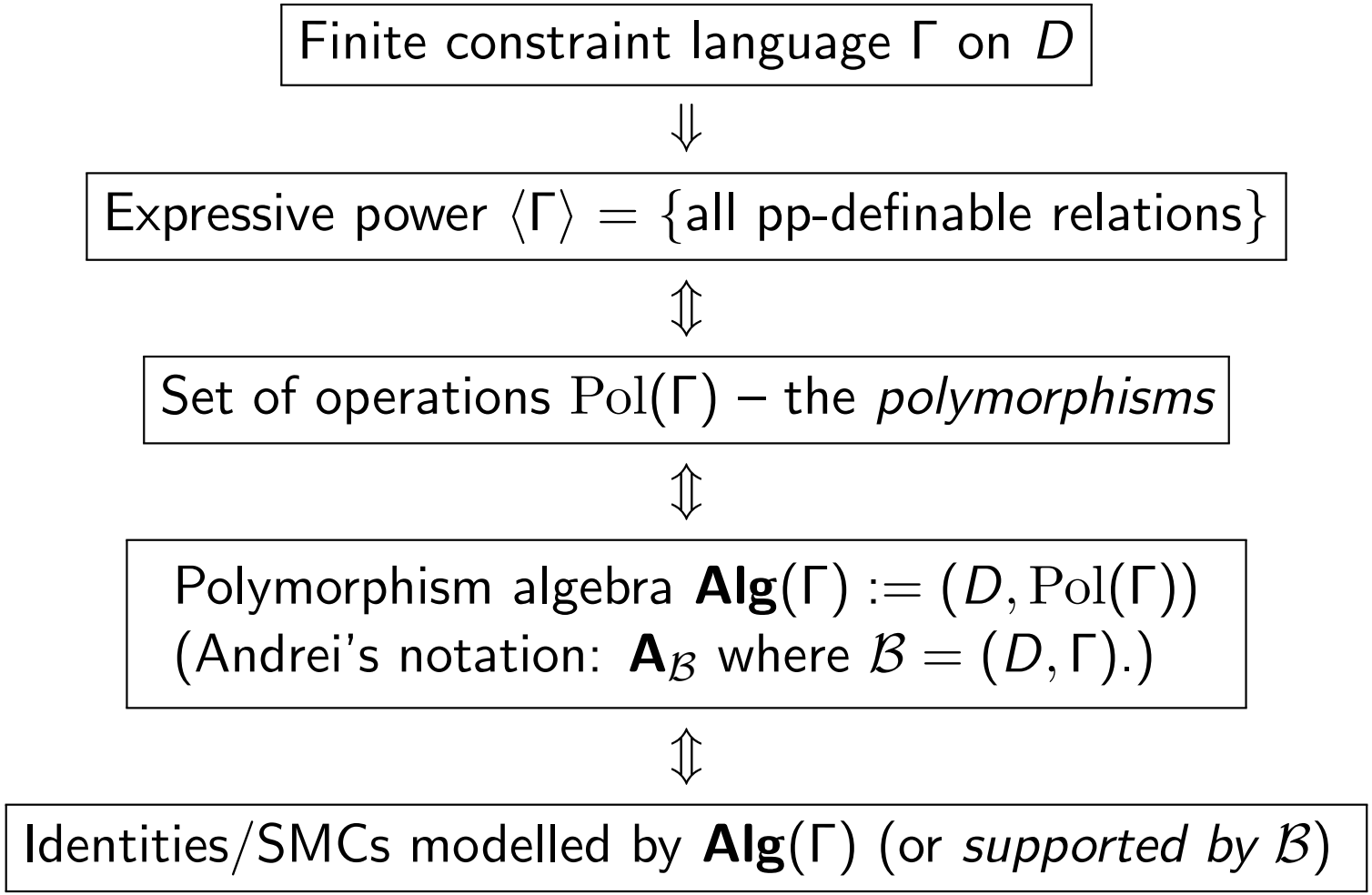
[With some corrections]

Ross Willard

Waterloo, Canada

CSP: Complexity and Approximability
Dagstuhl
Nov 6, 2012

Recall from Andrei's tutorial: what controls the complexity of $\mathrm{CSP}(\Gamma)$



But first a word about identities/Strong Mal'tsev Conditions (SMCs).

Suppose $\varepsilon_1, \ldots, \varepsilon_k$ are identities in formal function symbols f_1, f_2, ...

Let S be a set of operations on a domain D. The condition (on S)

$$\boxed{\begin{array}{c}\exists\ f_1, f_2, \ldots \in S \text{ (of the correct arities) such that} \\ (D, f_1, f_2, \ldots) \models \varepsilon_1\ \&\ \cdots\ \&\ \varepsilon_k\end{array}}$$

is a **Strong Mal'tsev condition**. (The SMC given by $\{\varepsilon_1, \ldots, \varepsilon_k\}$.)

Not to be confused with

- Mal'tsev condition
- Weak Mal'tsev condition
- Mal'tsev's condition
- Identities

Suggestive examples

D	Γ	**Alg**(Γ)	an interesting SMC
$\{0,1\}$	Γ_{3SAT}	$(\{0,1\}, \langle\varnothing\rangle)$	none
$\{0,1\}$	$\Gamma_{\mathrm{HornSAT}}$	$(\{0,1\}, \langle \mathit{min}\rangle)$	semilattice laws (ACI)
$\{0,1\}$	Γ_{2SAT}	$(\{0,1\}, \langle \mathit{majority}\rangle)$	majority laws
$\mathbb{Z}_p$	$\Gamma_{\mathrm{LinEq}/\mathbb{Z}_p}$	$(\mathbb{Z}_p, \langle x-y+z\rangle)$	Mal'tsev laws

Semilattice laws: $\mathrm{f}(x,\mathrm{f}(y,z)) = \mathrm{f}(\mathrm{f}(x,y),z)$, $\mathrm{f}(x,y) = \mathrm{f}(y,x)$, $\mathrm{f}(x,x) = x$.

Majority laws: $\mathrm{f}(x,x,y) = \mathrm{f}(x,y,x) = \mathrm{f}(y,x,x) = x$.

Mal'tsev laws $\mathrm{f}(x,x,y) = \mathrm{f}(y,x,x) = y$.

Reduction to the idempotent case:

- Can assume WLOG that $\{\{a\} : a \in D\} \subseteq \Gamma$.
- Then all $f \in Pol(\Gamma)$ are *idempotent*, i.e., satisfy $f(x, x, \ldots, x) = x$.

In this case, I'll write $\Gamma = \Gamma^c$.

Structural Dichotomy Conjecture (Bulatov, Jeavons, Krokhin 2005)

Assume $\Gamma = \Gamma^c$.

1. **Theorem**. If $(\{0, 1\}, \Gamma_{3\mathrm{SAT}})$ is pp-interpretable in (D, Γ), then $\mathrm{CSP}(\Gamma)$ is NP-complete.
2. **Conjecture**. Otherwise, $\mathrm{CSP}(\Gamma)$ is in P.

Goals of this lecture:

1. Describe two further reductions.
2. Describe two general P-time CSP algorithms.
   - Local consistency
   - "Few subpowers"
3. Explain where algebra plays a role
4. A few words about other conjectured dichotomies

Two further reductions

First reduction: to *binary* constraint languages (all relations are 1-ary or 2-ary).

Idea: Let Γ be a finite constraint language on D.

Choose $2n \geq$ max arity of relations in Γ.

$\exists$ a binary constraint language Γ_{bin} on D^n so that $\mathbf{Alg}(\Gamma_{\mathrm{bin}}) = (\mathbf{Alg}(\Gamma))^n$.

Thus each of Γ, Γ_{bin} is pp-interpretable in the other.

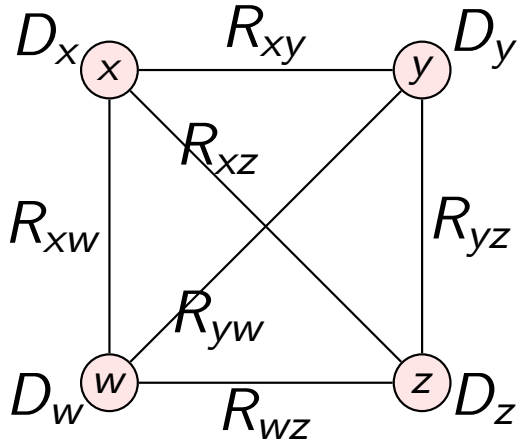
Note: for convenience, we assume our binary constraint language is *closed* (all 1-ary and 2-ary relations in $\langle\Gamma\rangle$ are already in Γ.)

Second reduction: to networks. Assume Γ is binary.

Definition

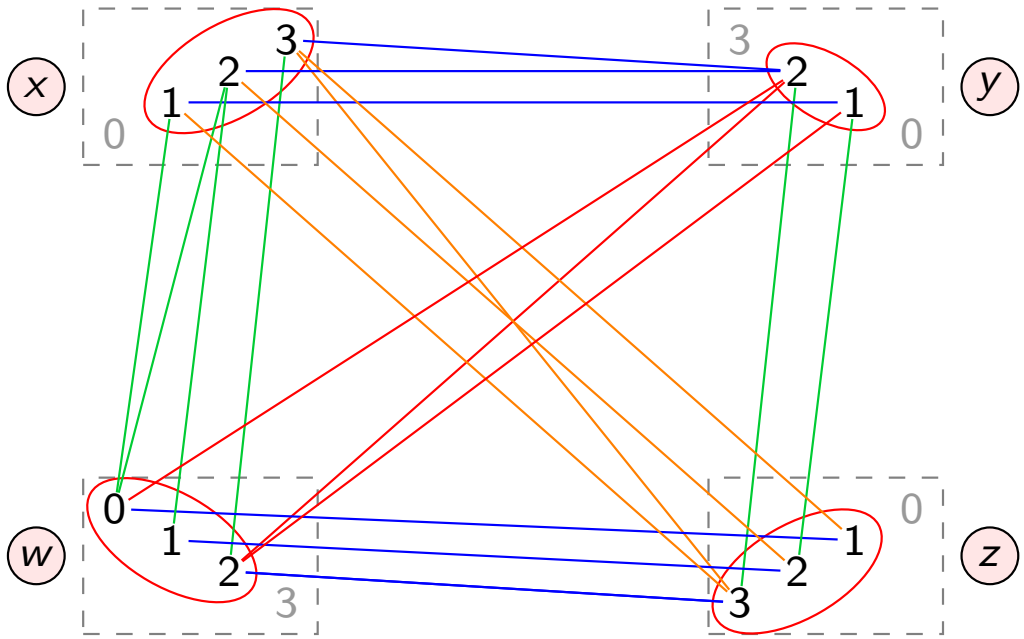
An instance $(V, \{\text{constraints}\})$ of $\mathrm{CSP}(\Gamma)$ is a *network* if:

1. Each $x \in V$ is the scope of exactly one constraint $(\{x\}, D_x)$.
2. Each pair $\{x, y\} \subseteq V$ with $x \neq y$ is the scope of exactly one constraint $(\{x, y\}, R_{xy})$. (Define $R_{yx} = R_{xy}^{-1}$ and $R_{xx} = \{(a, a) : a \in D_x\}$.)
3. $R_{xy} \subseteq D_x \times D_y$ for all x, y.



Fact: if Γ is binary and closed, then $\mathrm{CSP}(\Gamma) \equiv_L \mathrm{CSP}(\Gamma)\restriction_{\text{networks}}$.

Networks can be visualized. For example, suppose $D = \{0, 1, 2, 3\}$ and Γ is the set of all 1-ary and 2-ary relations on D. Here is a network for Γ:

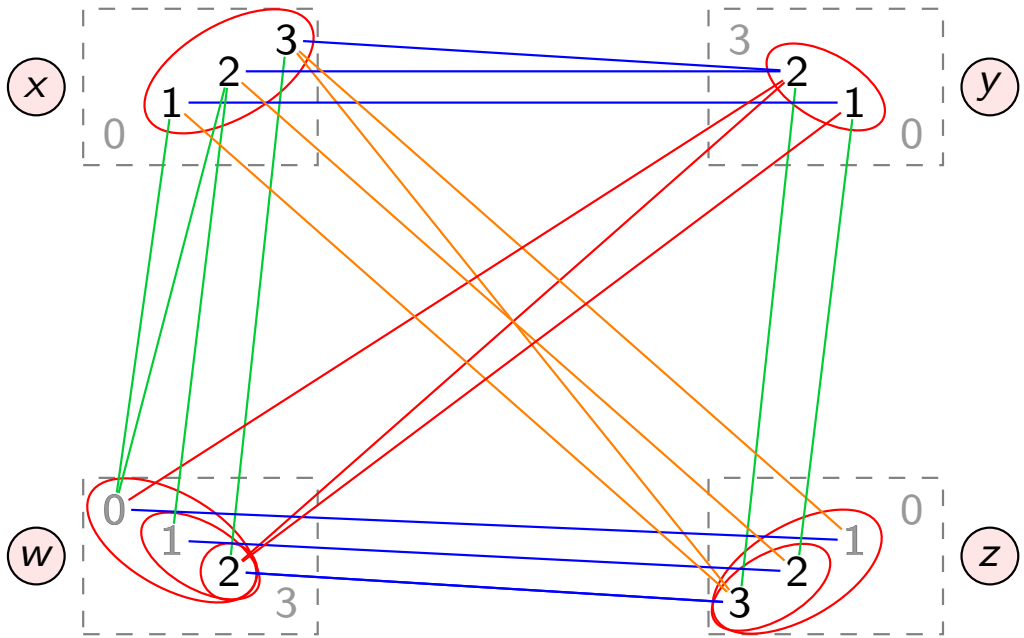


A solution is a *clique*.

Algorithm #1: Local consistency

Enforcing arc-consistency

Idea: Look for $x, y \in V$ and $a \in D_x$ having no edge (in R_{xy}) to D_y.



If found: replace D_x with $\mathrm{pr}_x(R_{xy})$.

This shrinks the network without changing its set of solutions.

Clearly: if some D_x becomes empty, the original network has no solution.

Definition

A binary network $\mathcal{N} = (V, (D_x)_{x \in V}, (R_{xy})_{x,y \in V})$ for Γ is *(1,2)-consistent*[a] if $\mathrm{pr}_x(R_{xy}) = D_x$ for all $x, y \in V$.

[a] Also called *(1,2)-minimal*, *1-minimal*, etc.

I.e., enforcing arc-consistency makes no changes.

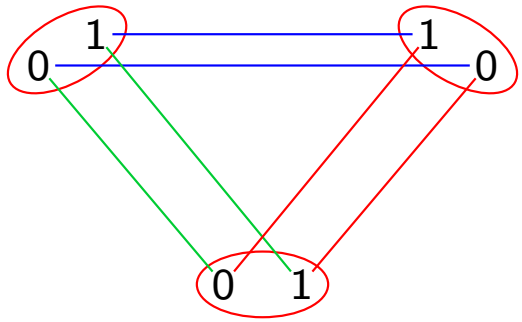
Fact (Montanari, 1974), or Exercise

$\exists$ a P-time algorithm which, given a binary network $\mathcal{N}$, either

1. Deduces an empty constraint by enforcing arc-consistency, or
2. produces an equivalent (1,2)-consistent subnetwork of $\mathcal{N}$.

This is the *enforcing arc-consistency* algorithm.

Note: (1,2)-consistent networks may have solutions, or may not. E.g.,



Definition

A binary, closed constraint language Γ *has width (1,2)* if every (1,2)-consistent network over Γ has a solution.

For such Γ, enforcing arc-consistency is a P-time algorithm solving $\mathrm{CSP}(\Gamma)$.

Question: Which Γ have width (1,2)?

Definition. $f : D^n \to D$ is a *set operation* if it satisfies $f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$ whenever $\{x_1, \ldots, x_n\} = \{y_1, \ldots, y_n\}$.

Example: if $\wedge$ is a semilattice operation on D (associative, commutative, and idempotent), then $\forall n$,

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

is a set operation.

Theorem (Dalmau & Pearson, 1999)

Suppose Γ *is a binary, closed constraint language on a domain* D *of size* d. *Let* $n = d^2$. *TFAE:*

1. Γ *has width (1,2).*
2. Γ *has an n-ary polymorphism* f *which is a set operation.*

E.g. $\Gamma_{\mathrm{HornSAT}}$ has width (1,2) (as *min* is a polymorphism).

Proof (2) $\Rightarrow$ (1). Assume Γ has an n-ary polymorphism satisfying $f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$ whenever $\{x_1, \ldots, x_n\} = \{y_1, \ldots, y_n\}$.

Let $\mathcal{N} = (V, (D_x)_x, (R_{xy})_{x,y})$ be a (1,2)-consistent network for Γ.

For each pair $x, y \in V$, list the edges in R_{xy} (padding the list to length n). Apply f to the list (coordinatewise).

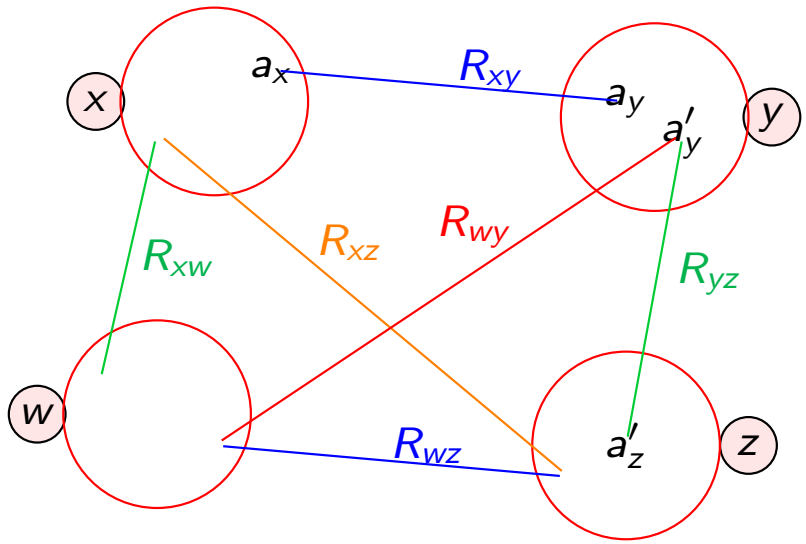
$$\left.\begin{array}{c} \begin{pmatrix} b_1 & , & c_1 \end{pmatrix} \\ \begin{pmatrix} b_2 & , & c_2 \end{pmatrix} \\ \vdots \quad\quad \vdots \\ \begin{pmatrix} b_n & , & c_n \end{pmatrix} \\ \hline \begin{pmatrix} f(\mathbf{b}) & , & f(\mathbf{c}) \end{pmatrix} \end{array}\right\} = R_{xy}$$

$$(a_x, a_y) \quad := \quad \in R_{xy}$$

(Last "$\in R_{xy}$" because R_{xy} is invariant under f.)

This chooses a **special edge** in R_{xy} for each pair $x, y \in V$.

Edges chosen by f



Focus on $(a_x, a_y) \in R_{xy}$ and $(a'_y, a'_z) \in R_{yz}$. **Claim**: $a_y = a'_y$.

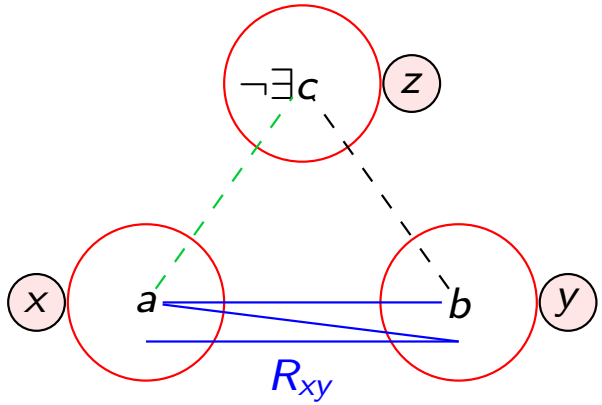
$$R_{xy} = \left\{ \begin{array}{cc} (\ b_1\ ,\ c_1\ ) & (\ c'_1\ ,\ d'_1\ ) \\ \vdots \quad \vdots & \vdots \quad \vdots \\ (\ b_n\ ,\ c_n\ ) & (\ c'_n\ ,\ d'_n\ ) \end{array} \right\} = R_{yz}$$

$$(a_x, a_y) := (\ f(\mathbf{b})\ ,\ f(\mathbf{c})\ ) \qquad (\ f(\mathbf{c}')\ ,\ f(\mathbf{d}')\ ) =: (a'_y, a'_z)$$

We're in a (1,2)-consistent network, so both $\{c_1, \ldots, c_n\}$ and $\{c'_1, \ldots, c'_n\}$ are enumerations of D_y. Hence $f(\mathbf{c}) = f(\mathbf{c}')$, so we have a clique. □

(2,3)-consistency

Idea: Look for $x, y, z \in V$ and $(a, b) \in R_{xy}$ which doesn't extend to a 3-clique on x, y, z.



If found: replace R_{xy} with $proj_{xy}(\{\text{3-cliques on } x, y, z\})$.

(And enforce (1,2)-consistency.)

As before: if a constraint becomes empty, the original network has no solution.

Definition

A binary network $\mathcal{N} = (V, (D_x), (R_{xy}))$ is *(2,3)-consistent* if it is (1,2)-consistent and

- Every edge can be extended to a triangle (at any $z \in V$).

Equivalently, $\mathcal{N}$ is (2,3)-consistent if enforcing (2,3)-consistency yields no changes.

More generally: (j, k)-consistency = "all $\leq j$-cliques extend to k-cliques."[1]

[1] Oops, I must be more careful. Let $\Gamma^{(j)}$ denote the expansion of Γ to all $\leq j$-ary relations in $\langle\Gamma\rangle$. Define a *j-network* to be like a network except that every at-most j-element subset $J \subseteq V$ of variables is the scope of exactly one constraint with constraint relation R_J; and for all such J and $\varnothing \neq I \subset J$ we have $\mathrm{pr}_I(R_J) \subseteq R_I$. Every network $\mathcal{N}$ over Γ easily gives rise to a j-network $\mathcal{N}^{(j)}$ over $\Gamma^{(j)}$: for R_J simply take the set of all cliques on J determined by $\mathcal{N}$. However, we will later need to consider j-networks not arising from networks in this way. Generally, a j-network is defined to be (j, k)-*consistent* if $\forall J \subseteq K \subseteq V$ with $|J| \leq j$ and $|K| \leq k$, every tuple in R_J can be extended to a tuple in D^K whose projection to every at-most j-element $L \subseteq K$ belongs to R_L. This agrees with the footnoted "definition" for j-networks of the form $\mathcal{N}^{(j)}$.

Fact

Fix $j < k$. $\exists$ a P-time algorithm which, given a binary network $\mathcal{N}$, either

1. Deduces an empty constraint by enforcing[a] (j,k)-consistency, or
2. Produces an equivalent (j,k)-consistent subnetwork of $\mathcal{N}^{(j)}$.

[a] Again I must be more careful. Given a network $\mathcal{N}$, we must form the j-network $\mathcal{N}^{(j)}$ as explained in the previous footnote and then enforce (j,k)-consistency starting from $\mathcal{N}^{(j)}$. During this enforcement the j-network will likely evolve and no longer be of the form $\mathcal{M}^{(j)}$ for any network $\mathcal{M}$.

Definition

A binary, closed constraint language Γ *has width* (j,k) if every (j,k)-consistent j-network over $\Gamma^{(j)}$ has a solution.

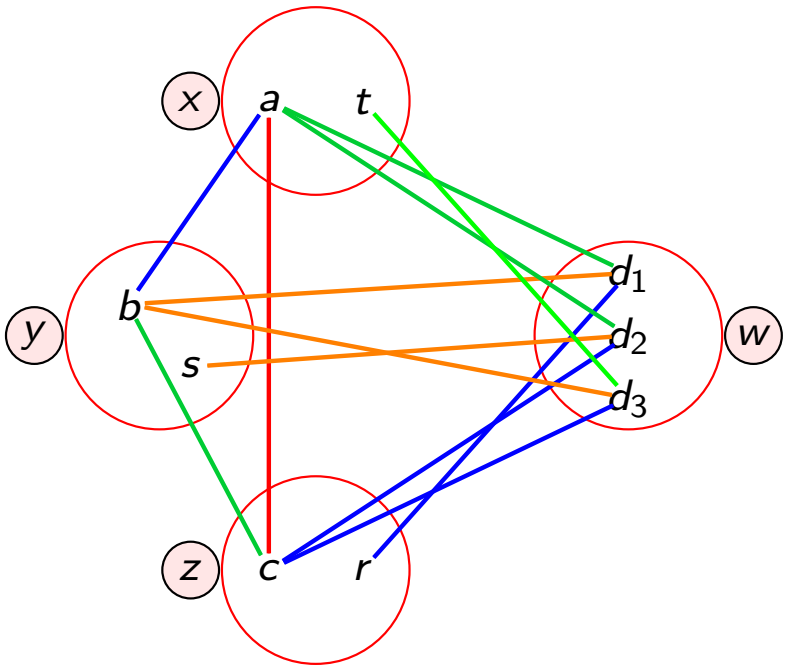
For such Γ, enforcing (j,k)-consistency solves $\mathrm{CSP}(\Gamma)$.

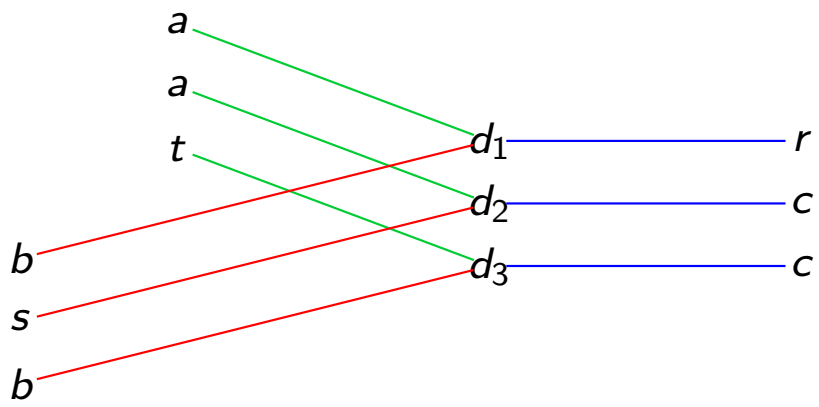
Theorem (Jeavons & Cohen, 1997)

If Γ is binary, closed, and has a majority polymorphism, i.e., a 3-ary f satisfying $f(x,x,y) = f(x,y,x) = f(y,x,x) = x$, then Γ has width (2,3).

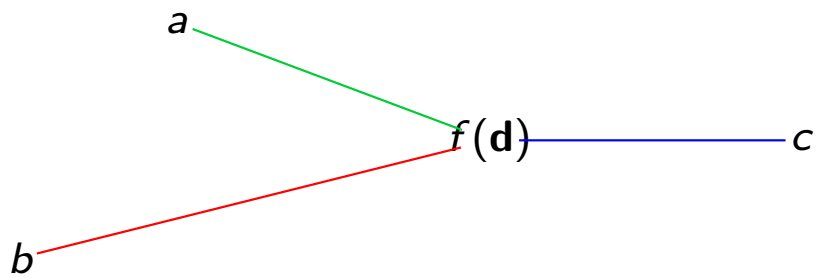
E.g. Γ_{2SAT} has width (2,3).

Proof idea. Let $\mathcal{N}$ be a (2,3)-consistent network. Show inductively that it is $(k-1,k)$-consistent $\forall k$. E.g., let $k = 4$, assume (a,b,c) is a triangle on x, y, z and w is a another variable.





$\Downarrow$ apply the majority polymorphism



A similar argument shows that if binary Γ has a k-ary near-unanimity (NU) polymorphism, then Γ has width $(2, k)$.

Definition

Γ has *bounded width* if it has width (j, k) for some $1 \leq j < k$.

Constraint languages having bounded width are precisely those Γ for which local consistency checking gives a P-time algorithm for $\mathrm{CSP}(\Gamma)$.

The class of bounded with constraint languages is robust: characterized by

- $\mathrm{CSP}(\Gamma)$ having bounded treewidth duality.
- Definability of $\neg\mathrm{CSP}(\Gamma)$ in Datalog.
- Winning strategy for a natural pebble game.

Question: which Γ have bounded width?

The "obvious" obstruction to bounded width is linear equations.

- $\Gamma_{\mathrm{LinEq}/\mathbb{Z}_m}$ does not have bounded width, for any m.
- Neither does $\Gamma_{\mathrm{Coset}/M}$ (M any finite R-module).

Larose & Zadori proved that if $(\mathbb{Z}_m, \Gamma_{\mathrm{LinEq}/\mathbb{Z}_m})$ (or more generally, $(M, \Gamma_{\mathrm{Coset}/M})$) is pp-interpretable in (D, Γ), then Γ also does not have bounded width.

Bounded Width Conjecture (Larose, Zadori, 2007). There are no other obstacles. That is, assume $\Gamma = \Gamma^c$. The following are equivalent:

1. Γ has bounded width.
2. No "coset structure" $(M, \Gamma_{\mathrm{Coset}/M})$ is pp-interpretable in (D, Γ).

Define

$$\mathcal{K} = \{\Gamma : \Gamma = \Gamma^c, \text{ no } (M, \Gamma_{\mathrm{Coset}/M}) \text{ is pp-interpretable in } (D, \Gamma)\}.$$

$\{\mathbf{Alg}(\Gamma) : \Gamma \in \mathcal{K}\}$ is a well-studied class of finite algebras.

Theorem (Hobby, McKenzie, Szendrei). For $\mathbf{A} = \mathbf{Alg}(\Gamma^c)$, TFAE:

- $\Gamma^c \in \mathcal{K}$
- $\mathrm{var}(\mathbf{A})$ "omits types 1,2"
- $\mathrm{var}(\mathbf{A})$ is "congruence SD($\wedge$)"
- $\mathbf{A}$ satisfies an explicit (though messy) Mal'tsev condition

} "tame congruence theory"

- $\forall$ sufficiently large n, $\mathbf{A}$ has an n-ary "weak NU" operation f_n (Maróti, McKenzie, 2008).

$$f_n(y, x, \ldots, x) = f_n(x, y, \ldots, x) = \cdots = f_n(x, \ldots, x, y).$$

  - Can assume $f_m(\underbrace{x, \ldots, x}_{m-1}, y) = f_n(\underbrace{x, \ldots, x}_{n-1}, y)$ for all m, n. (BK, 2009).

Remark: really need deep algebra here.

Theorem (Barto, Kozik, 2009)

The Bounded Width Conjecture is true. In fact, if $\Gamma = \Gamma^c$ *is binary and* Γ *fails to interpret any coset structure, then* Γ *has width (2,3).*

Remarks on the proof.

1. Fix a (2,3)-consistent network $\mathcal{N}$ over Γ. As in the proofs for set operation and majority polymorphisms, the idea is to "shrink" $\mathcal{N}$ to a clique, using available polymorphisms.

2. The proof is devishly complicated and marvelously clever.

3. Like the proof in the majority case, polymorphisms are applied repeatedly.

4. But the applications are MUCH more complicated.

5. Coordinated WNUs (of very high arity) are just enough to work.

Algorithm #2: Few subpowers

Let $\Gamma = \Gamma^c$ be a constraint language (binary if you like).

Let $\mathcal{N} = (V, (C_t)_{t=1}^m)$ be an instance of $\mathrm{CSP}(\Gamma)$, (A network if you like.)

Let $n = |V|$, and linearly order $V = \{x_1, \ldots, x_n\}$.

Thus assignments $\alpha : V \to$ may be identified with elements of D^n.
In this framework, define the following subsets of D^n:

$$\begin{aligned} S_0 &= D^n \\ S_1 &= \{\text{solutions to } (V, \{C_1\})\} \\ S_2 &= \{\text{solutions to } (V, \{C_1, C_2\})\} \\ &\vdots \\ S_m &= \{\text{solutions to } (V, \{C_1, \ldots, C_m\}) = \mathcal{N}\} \end{aligned}$$

Then $D^n = S_0 \supseteq S_1 \supseteq S_2 \supseteq \cdots \supseteq S_m$ and want to know whether $S_m = \varnothing$.

$$D^n = S_0 \supseteq S_1 \supseteq S_2 \supseteq \cdots \supseteq S_m = \text{Solutions}(\mathcal{N})$$

The *few subpowers algorithm* (BD + IMMVW):

- is *not* based on reasoning with constraints.
- instead, it successively computes "*nice generating sets*" for each S_t, considered as a subalgebra of $\mathbf{Alg}(\Gamma)^n$.
- At the start, a nice generating set is easily provided for S_0.
- In the end, $\mathcal{N}$ has a solution $\Leftrightarrow$ the last generating set is $\neq \varnothing$.

This is **very** loosely analogous to Gaussian elimination.

More accurately, it is based on algorithms for computing in permutation groups.

Special case: when Γ has a Mal'tsev polymorphism

Bulatov & Dalmau, A simple algorithm for Mal'tsev constraints, 2006. [2]

Recall: the Mal'tsev laws are $\mathrm{f}(x,x,y) = \mathrm{f}(y,x,x) = y$. (Think $x - y + z$.)

Definition

Suppose $S \subseteq D^n$.

$$\mathrm{Fork}(S) = \{(i,b,c) \in [n] \times D \times D : \exists \mathbf{u}, \mathbf{v} \in S \text{ with } u_j = v_j \text{ for all } 1 \leq j < i, \text{ and } (u_i, v_i) = (b,c)\}.$$

A subset $T \subseteq S$ is called a **compact representation of** S if $\mathrm{Fork}(T) = \mathrm{Fork}(S)$ and T is minimal with respect to this property.

Exercise: T a compact rep. for $S \subseteq D^n \Rightarrow |T| \leq n|D|^2$.

[2] See also Dyer & Richerby, An effective dichotomy for counting CSP.

Some algebraic housekeeping

Let $\mathbf{A} = (D, \{\text{operations}\})$ be an algebra. (For example, $\mathbf{A} = \mathbf{Alg}(\Gamma)$.)
Let $S, T \subseteq D$.

- S is a *subalgebra* of $\mathbf{A}$ if S is closed under all the operations of $\mathbf{A}$.
  - (In the example: $\Leftrightarrow$ S is a pp-definable 1-ary relation from Γ.)
- The *subalgebra generated by* T is the iterated closure of T under the operations of $\mathbf{A}$. Denote it by $\langle\langle T \rangle\rangle_{\mathbf{A}}$.
  - (In the example: $\langle\langle T \rangle\rangle_{\mathbf{A}}$ is the smallest pp-definable (from Γ) 1-ary relation containing T.)
- We say $\mathbf{A}$ *has a Mal'tsev operation* if $\langle\{\text{operations}\}\rangle$ contains one.
  - (Example: $= \Gamma$ has a Mal'tsev polymorphism.)

Key Fact (Bulatov, Dalmau)

Suppose $\mathbf{A}$ *is an algebra having a Mal'tsev operation,* $n \geq 1$*, and* S *is a subalgebra of* $\mathbf{A}^n$*. If* T *is a compact representation of* S*, then* $\langle\langle T \rangle\rangle_{\mathbf{A}^n} = S$*.*

Proof idea

Clearly $\langle\langle T \rangle\rangle_{\mathbf{A}^n} \subseteq S$. Suppose $\mathrm{pr}_{1,\ldots,i-1}(\langle\langle T \rangle\rangle_{\mathbf{A}^n}) = \mathrm{pr}_{1,\ldots,i-1}(S)$.
We will show $\mathrm{pr}_{1,\ldots,i}(\langle\langle T \rangle\rangle_{\mathbf{A}^n}) = \mathrm{pr}_{1,\ldots,i}(S)$.
Pick $\mathbf{a} = (a_1, \ldots, a_{i-1}, a_i, \ldots) \in S$.
So $\exists \mathbf{a}' = (a_1, \ldots, a_{i-1}, b, \ldots) \in \langle\langle T \rangle\rangle_{\mathbf{A}^n}$. (Thus also $\mathbf{a}' \in S$.)
Thus $(i, a_i, b) \in \mathrm{Fork}(S) = \mathrm{Fork}(T)$.
Pick $\mathbf{u}, \mathbf{v} \in T$ witnessing this.
We have

$$\begin{aligned} \mathbf{u} &= (u_1, \ldots, u_{i-1}, a_i, \ldots) \in T \\ \mathbf{v} &= (u_1, \ldots, u_{i-1}, b, \ldots) \in T \\ \mathbf{a}' &= (a_1, \ldots, a_{i-1}, b, \ldots) \in \langle\langle T \rangle\rangle_{\mathbf{A}^n}. \end{aligned}$$

Proof idea (continued).

We have

$$\begin{aligned} \mathbf{u} &= (u_1, \ldots, u_{i-1}, a_i, \ldots) \in T \\ \mathbf{v} &= (u_1, \ldots, u_{i-1}, b, \ldots) \in T \\ \mathbf{a}' &= (a_1, \ldots, a_{i-1}, b, \ldots) \in \langle\langle T \rangle\rangle_{\mathbf{A}^n}. \end{aligned}$$

Applying the Maltsev operation, we get

$$f(\mathbf{u}, \mathbf{v}, \mathbf{a}') = (a_1, \ldots, a_{i-1}, a_i, \ldots) \in \langle\langle T \rangle\rangle_{\mathbf{A}^n}$$

as desired. □

The BD Algorithm:

Recall the $\mathrm{CSP}(\Gamma)$ instance $\mathcal{N} = (V, (C_t)_{t=1}^m)$ with $V = \{x_1, \ldots, x_n\}$.

We have

$$D^n \supseteq S_0 \supseteq S_1 \supseteq S_2 \supseteq \cdots \geq S_m = \{\text{solutions to } \mathcal{N}\}. \cap S_0. \qquad (\dagger)$$

Observe that the S_t are subalgebras of $\mathbf{Alg}(\Gamma)^n$.

For "nice generating sets" (of the S_t) we will use compact representations.

[Relaxation: S_0 can be any subalgebra of $\mathbf{Alg}(\Gamma)^n$; require a compact representation for S_0 as additional input.]

"FixValues" Lemma

Compact rep's for all S_t can be found (in P-time) in the special case of the relaxation ($D^n \supseteq S_0$) where

- $m < n$, and
- For all $t \geq 1$, the constraint defining S_t relative to S_{t-1} has the form "$x_t = a_t$." (OK since $\Gamma = \Gamma^c$.)

(Proof idea: each S_t is "rectangular.")

Recall

$$D^n = S_0 \supseteq S_1 \supseteq S_2 \supseteq \cdots \geq S_m = \{\text{solutions to } \mathcal{N}\} \qquad (\dagger)$$

(no longer need the relaxation).

At stage t, we wish to compute a compact rep. for S_t, given a compact rep. T for S_{t-1} and the constraint C that defined S_t relative to S_{t-1}.

Say $S_t = S_{t-1} \cap \{\text{“}(x_j, x_k) \in R\text{”}\}$.

Key task: For each $(i, a, b) \in [n] \times D \times D$, we need to decide whether $(i, a, b) \in \mathrm{Fork}(S_t)$ and, if “yes,” we must find a witnessing pair $\mathbf{u}, \mathbf{v} \in S_t$.

Because of the rectangularity of S_t, it suffices to first search for **any** $\mathbf{u} \in S_t$ satisfying $(u_j, u_k) \in R$ and $u_i = a$.

If one is found, then search for $\mathbf{v} \in S_t$ so that $\mathbf{u}, \mathbf{v}$ witness (i, a, b). This can be done by applying the “FixValues” Lemma to the special case of the relaxation of $(\dagger)$ starting at S_{t-1}, letting $m = i$, and applying the constraints “$x_1 = u_1$,” “$x_2 = u_2$,” …, “$x_{i-1} = u_{i-1}$.”

Generalizing the Bulatov-Dalmau algorithm

A key ingredient in the BD algorithm is the following obviously necessary feature of Γ. Let $\mathbf{A} = \mathbf{Alg}(\Gamma)$.

Required Property

Every subalgebra of $\mathbf{A}^n$ that arises (i.e., as the set of solutions of a CSP over Γ) has a generating set whose size is bounded by a polynomial in n.

Consider the following more stringent property:

Few subpowers

An algebra $\mathbf{A}$ has **few subpowers** if **every** subalgebra of $\mathbf{A}^n$ has a generating set whose size is bounded by a polynomial in n.

IMMVW (2010) + BIMMVW (2010) + Dalmau (2005) adapted the BD algorithm to work for all Γ for which $\mathbf{Alg}(\Gamma)$ has few subpowers.

Question: Does $\exists\ \Gamma$ satisfying the Required Property yet not having few subpowers?

Two related conjectures

There are such things called "bounded pathwidth duality" $\Leftrightarrow$ "definable in linear Datalog," both implying $\mathrm{CSP}(\Gamma) \in \mathrm{NL}$.

Obvious obstructions:

- $(M, \Gamma_{\mathrm{Coset}/M})$ (M a finite simple R-module).
- Horn-SAT.

Speculation #1 (Larose, Tesson): these are the only obstructions.

Algebra characterizes those $\Gamma = \Gamma^c$ which do not pp-interpret any $(M, \Gamma_{\mathrm{Coset}/M})$ or $(\{0,1\}, \Gamma_{\mathrm{HornSAT}})$.

- "Omit types 1,2,5."
- "Congruence SD($\vee$)."
- A Mal'tsev condition.

Marcin Kozik has something to say on this.

There is such a thing called "definable in symmetric Datalog," implying $\mathrm{CSP}(\Gamma) \in \mathrm{L}$.

Obvious obstructions:

- $(M, \Gamma_{\mathrm{Coset}/M})$ (M a finite simple R-module).
- Horn-SAT.
- Directed *st*-connectivity ($= \mathrm{CSP}(\leq, \{0\}, \{1\})$).

Speculation #2 (Larose, Tesson): these are the only obstructions.

Algebra characterizes those $\Gamma = \Gamma^c$ which do not pp-interpret any $(M, \Gamma_{\mathrm{Coset}/M})$ or $(\{0,1\}, \Gamma_{\mathrm{HornSAT}})$ or $\mathit{PATH} = (\{0,1\}, \leq, \{0\}, \{1\})$.

- "Omit types 1,2,4,5."
- A Mal'tsev condition.

No one has anything new to say.

And of course the Holy Grail

Structural Dichotomy Conjecture

Assume $\Gamma = \Gamma^c$. If $(\{0,1\}, \Gamma_{3\mathrm{SAT}})$ is not pp-interpretable in (D, Γ), then $\mathrm{CSP}(\Gamma)$ is in P.

Algebra characterizes these $\Gamma = \Gamma^c$ too, providing Mal'tsev conditions. (Existence of "cyclic operations," due to Barto & Kozik, is particularly deep.)

Wide open question: Characterize (combinatorially) the finite graphs (V, E) which have polymorphisms characterizing:

- Structural Dichotomy Conjecture
- Linear Datalog Conjecture
- Symmetric Datalog Conjecture

Thank you!