Fine grained analysis of conservative Maltsev CSP

BLAST 2025

Andrew Moorhead
Joint work with Manuel Bodirsky

Institute für Algbera
TU Dresden

May 20, 2025

Funded by
the European Union

European Research Council
Established by the European Commission

Contents

Introduction

Introduction

The *(finite domain) constraint satisfaction problem*:

- Input: two similar structures $\mathfrak{I}$ and $\mathfrak{A}$ over a finite relational signature τ
- Output: 'Yes' if there exist a homomorphism $s : \mathfrak{I} \to \mathfrak{A}$, else 'No'

Easy facts:

- CSP belongs to the class NP,
- The CSP is NP-hard, since it can simulate NP-hard problems.

Fixed template CSP $(\mathrm{CSP}(\mathfrak{A}))$: fix the target structure $\mathfrak{A}$ to specify a *constraint language*

Introduction

Basic examples of fixed template finite domain CSP.

- $\mathrm{CSP}(\mathfrak{A})$ is NP-complete for templates $\mathfrak{A}$ such as
  - $K_3 = (\{a, b, c\}, \neq)$ (Graph 3-colorability)
  - $S = (\{0, 1\}, \{R_{(a,b,c)} = \{0, 1\}^3 \setminus \{(a, b, c)\} : a, b, c \in \{0, 1\}\})$ (3-SAT)
  - $H_k = ([k] = \{1, \dots, k\}, \mathrm{NAE} = [k]^3 \setminus \{(a, a, a) : a \in [k]\})$ (Generalized NAE-SAT)
- $\mathrm{CSP}(\mathfrak{A})$ is in P-TIME for templates $\mathfrak{A}$ such as
  - $K_2 = \{\{a, b\}, \neq)$ (Graph 2-colorability)
  - $\vec{C}_n = (\{0, \dots, n-1\}, \{(i, i+1 \mod n) : 0 \leq i \leq n-1\})$ (Existence of an n-cycle)
  - The reduct of S corresponding to Horn 3-SAT
  - $([n], \{G_\sigma \subseteq [n]^2 : \sigma$ a permutation of $[n]\}$
  - Systems of linear equations over a finite field
- In 1997, Feder and Vardi famously conjectured that a finite domain fixed template CSP is either NP-complete or in P-TIME. The conjecture was famously proven true independently by Bulatov and Zhuk in 2017.

Introduction

Definition

Let $\mathfrak{A}$ be a relational structure. An operation $f : A^n \to A$ is a *polymorphism* of $\mathfrak{A}$ if it is a homomorphism from $\mathfrak{A}^n \to \mathfrak{A}$. We denote by $\mathrm{Pol}(\mathfrak{A})$ the clone of polymorphisms of $\mathfrak{A}$. A *polymorphism algebra* for $\mathfrak{A}$ is an algebra $\mathbf{A}$ such that $\mathrm{Clo}(\mathbf{A}) = \mathrm{Pol}(\mathfrak{A})$.

Definition

An operation $f : A^4 \to A$ is *Kearnes-Markovic-McKenzie* if it satisfies the identity

$$f(xyzy) = f(yzxx)$$

Introduction

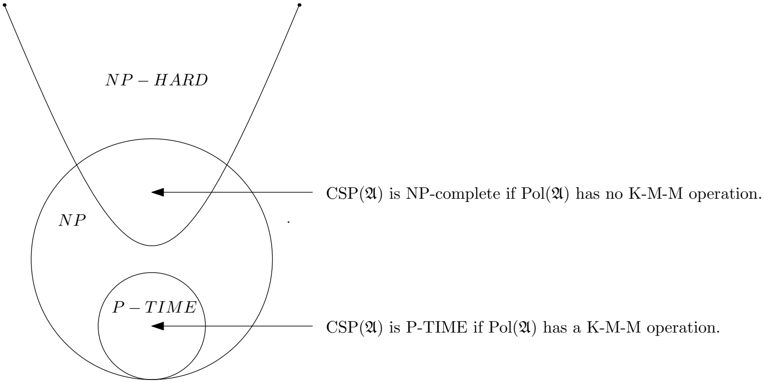


Figure: One of the many algebraic critera for CSP dichotomy

Introduction

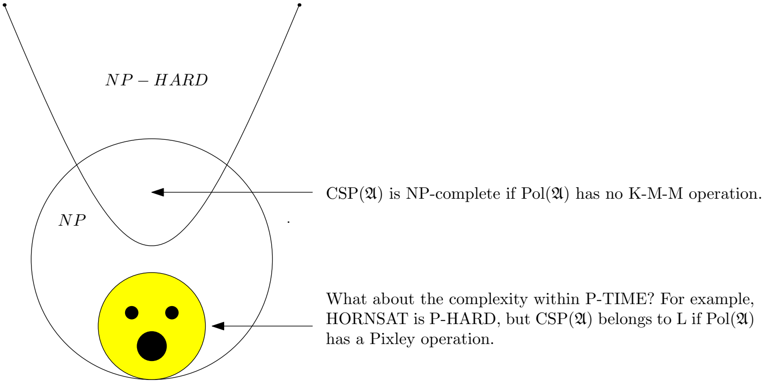


Figure: One of the many algebraic critera for CSP dichotomy

Introduction

Definition

- An operation $f : A^n \to A$ is *conservative* if

$$f(x_1, \ldots, x_n) \in \{x_1, \ldots, x_n\} \text{ for all } x_1, \ldots, x_n \in A.$$

- An ternary operation on A is *Maltsev* if

$$f(x, x, y) = f(y, x, x) = y \text{ for all } x, y \in A.$$

A CSP template $\mathfrak{A}$ is said to be conservative, or Maltsev, if all of its polymorphisms are conservative, or Maltsev.

Introduction

- Bulatov-Dalmau algorithm solves Maltsev CSPs in P-TIME by calculating a *compact representation* of the solution set of an input $\mathfrak{I}$.
- Bulatov also confirmed the dichotomy conjecture for all *conservative* templates (contains all unary constraints) early on.
- Problem: Bulatov-Dalmau algorithm does seem to admit an analysis which classifies conservative Maltsev CSP within P-TIME.

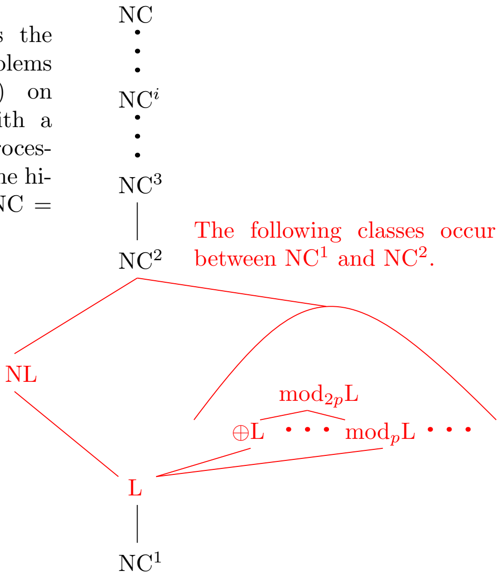
Our main result is the following.

Theorem (B+M)

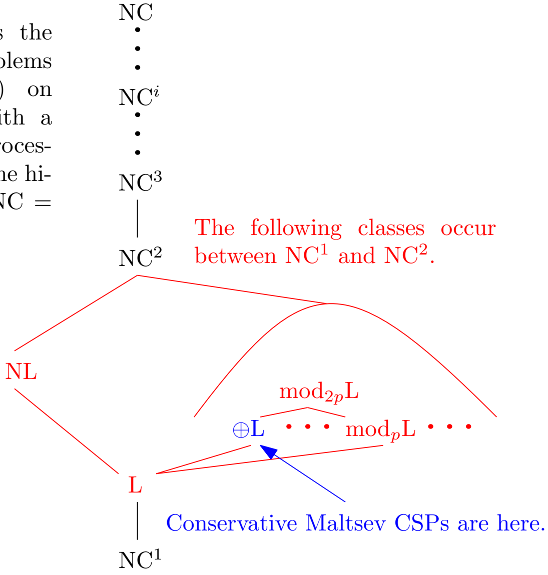
Let $\mathfrak{A}$ be a finite structure with a conservative Maltsev polymorphism. Then $\mathrm{CSP}(\mathfrak{A})$ is in $\oplus L$.

NC hierarchy: NC^i is the class of decision problems solvable in $O((\log n)^i)$ on a parallel computer with a polynomial number of processors (it is not known if the hierarchy collapses or if $\mathrm{NC} =$ P-TIME).

NC hierarchy: NC^i is the class of decision problems solvable in $O((\log n)^i)$ on a parallel computer with a polynomial number of processors (it is not known if the hierarchy collapses or if NC = P-TIME).



NC hierarchy: NC^i is the class of decision problems solvable in $O((\log n)^i)$ on a parallel computer with a polynomial number of processors (it is not known if the hierarchy collapses or if NC = P-TIME).



Introduction

- The classes mod_kL are related to *counting problems*, which ask for the number of accepting paths on an input to some nondeterministic Turing machine. Formally, we have the following.

Definition

Let $k > 1$. A language $\mathcal{L}$ belongs to the class mod_kL if there exists a nondeterministic logspace Turing machine T which halts on every computation path such that

$$\mathcal{L} = \{w : \text{number of accepting computation paths of } T \text{ on input } w \text{ is not divisible by } k\}$$

- All 'standard problems' of linear algebra problems for the rings $\mathbb{Z}_k$ are complete (up to logspace reductions) for the class mod_kL (Buntrock, Damm, Hertrampf, Meinel).

Introduction

We denote by $\oplus$L the class mod$_2$L.Luckily, we only need the following facts about $\oplus$L for our result:

1. L $\subseteq$ $\oplus$L (trivial, since L is deterministic).
2. The class of functions in FL$^{\oplus \mathsf{L}}$ with a fixed oracle is closed under composition (argue similarly to showing that logspace functions compose).
3. $\oplus$L is equal to the decision problems which belong to FL$^{\oplus \mathsf{L}}$ (Hertrampf, Reith, Vollmer).
4. Consistency of systems of $\mathbb{Z}_2$-linear equations is complete for $\oplus$L (Buntrock, Damm, Hertrampf, Meinel)

To show the result: compose logspace procedures with linear system solvability queries in a manner that solves $\mathrm{CSP}(\mathfrak{A})$ for a conservative Maltsev template $\mathfrak{A}$.

Overview

Minion homomorphisms and pp-constructability

- The algebraic approach to the CSP exploits the Galois connection between function clones and relational clones on a finite set. In particular, if $\mathrm{Pol}(\mathfrak{A})$ has a clone homomorphism to $\mathrm{Pol}(\mathfrak{B})$, then $\mathrm{CSP}(\mathfrak{B})$ is logspace reducible to $\mathrm{CSP}(\mathfrak{A})$.
- This perspective fails to account for homomorphic equivalence and adding all singleton unary relations to a core structure.
- Barto, Opršal, and Pinsker discovered that the notion of a clone homomorphism can be relaxed to what is called a *minion* homomorphism and that this has a tight connection to so-called pp-constructions. Pp-constructability covers all of the mentioned (logspace) reductions and has become an important perspective from which to organize finite domain CSP.
- We construct an algorithm for a specific class of structures with a conservative Maltsev polymoprhism and then show that this class pp-constructs all other structures with a conservative Maltsev polymorphism.

Definition

Let A be a nonempty set. A set of operations $\mathcal{M} \subseteq \mathrm{Op}(A)$ is called a (function) *minion* if it is closed under variable permutation, variable, identification, and adding dummy variables. The result of applying any composition of these operations to an operation $f \in \mathcal{M}$ is called a *minor* of f. A mapping $\psi : \mathcal{M} \to \mathcal{N}$ is a *minion homomorphism* if it commutes with taking minors.

Definition

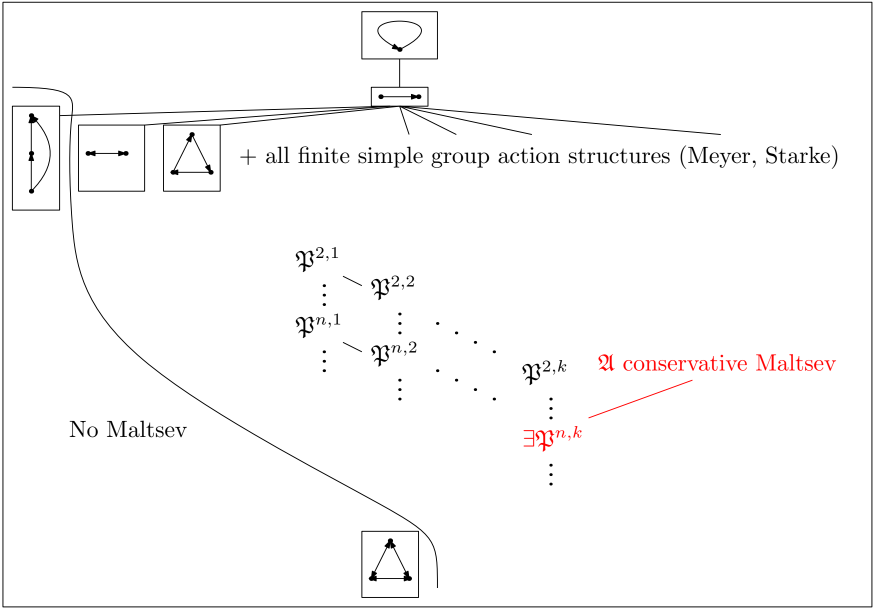
- A structure $\mathfrak{B}$ is said to be a *pp-power* of a structure $\mathfrak{A}$ if it is isomorphic to a structure with domain A^n, where $n \geq 1$, whose relations are pp-definable from $\mathfrak{A}$ (a k-ary relation of the pp-power is a kn-ary pp-definable relation in $\mathfrak{A}$).
- A (finite) structure $\mathfrak{A}$ is said to *pp-construct* a (finite) structure $\mathfrak{B}$ if $\mathfrak{B}$ is homomorphically equivalent to a pp-power of $\mathfrak{A}$.

Theorem (Barto, Pinkser, Opršal)

Let $\mathfrak{A}$ and $\mathfrak{B}$ be finite structures. The following are equivalent:

- $\mathfrak{A}$ *pp-constructs* $\mathfrak{B}$.
- $\mathrm{Pol}(\mathfrak{A})$ *has a minion homomorphism to* $\mathrm{Pol}(\mathfrak{B})$.

Moreover, $\mathfrak{A}$ *has no Siggers polymorphism if and only if* $\mathrm{Pol}(\mathfrak{A})$ *has a minion homomorphism to the clone of projections.*



Pp-constructability preorder (rectangles are equivalence classes)

Basic simplifying pp-construction

- A *minority* operation is a ternary operation m that satisfies

$$m(xxy) = m(xyx) = m(yxx) = y$$

for all inputs x, y.

- A *conservative minority algebra* is an algebra $\mathbf{A} = (A; m)$ with one basic operation m, which is a conservative minority operation on A.

Lemma (Carbonnel)

If a clone contains a conservative Maltsev operation, then it contains a conservative minority operation.

- Hence, every structure $\mathfrak{A}$ with a conservative Maltsev operation is pp-constructed by the invariant relations of some conservative minority algebra.

Structure and invariant relations of conservative minority algebras

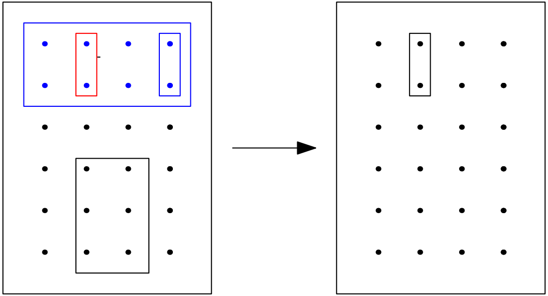
Conservative minority algebras have *very* well behaved congruences.

Lemma

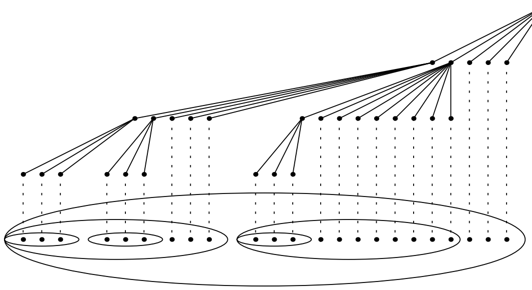
Let $\mathbf{A} = (A, m)$ *be a conservative minority algebra. Let* θ *be a congruence of* $\mathbf{A}$*, let* $B = [b]_\theta$ *be a* θ*-class, and let* $\mathbf{B} \leq \mathbf{A}$ *be the subalgebra with domain* B*. Let* α *be a congruence of* $\mathbf{B}$*. For every* $c \in B$*, the relation*

$$\alpha_c = \{(x, y) \in A^2 : x, y \in [c]_\alpha \text{ or } x = y\}$$

is a congruence of $\mathbf{A}$*.*



We call a congruence θ of a conservative minority algebra $\mathbf{A}$ a *(single) block congruence* if θ has at most one nontrivial class. Since $\mathbf{A}$ is congruence permutable, it follows that the blocks $[a]_\alpha, [b]_\beta$ of block congruences α, β are either disjoint or one is contained in the other. Therefore, every conservative minority algebra has a unique maximal congruence. Hence, there is a tree structure encoding the congruences of $\mathbf{A}$.



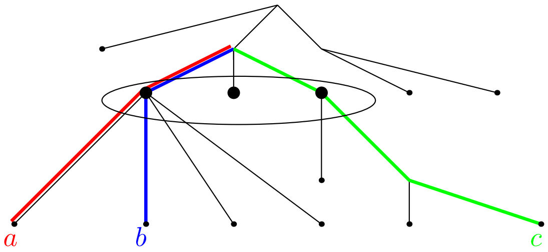
Definition

A *conservative minority tree* is a triple $\mathcal{T} = (T, \leq_{\mathcal{T}}, M_{\mathcal{T}})$ where

- T is a finite set,
- $(T; \leq_{\mathcal{T}})$ is a tree ordered so that the root is the maximal element, and
- $M_{\mathcal{T}} = (\mathbf{C}_{v,\mathcal{T}})_{v \in T \setminus L_{\mathcal{T}}}$ where
  - $L_{\mathcal{T}}$ is the set of leaves of $(T; \leq_{\mathcal{T}})$,
  - $C_{v,\mathcal{T}}$ is the set of children of v in $(T; \leq_{\mathcal{T}})$, for every $v \in T$,
  - $\mathbf{C}_{v,\mathcal{T}} = (C_{v,\mathcal{T}}; m_v)$ for some conservative minority operation $m_{v,\mathcal{T}} \colon C_{v,\mathcal{T}}^3 \to C_{v,\mathcal{T}}$.

A conservative minority tree is *simple* if every 'local' algebra $\mathbf{C}_{v,\mathcal{T}}$ is a simple algebra.
We say that a tree is *reduced* if every nonleaf has at least two children.

We define a minority operation on the leaves of a conservative minority tree $\mathcal{T}$ in the obvious way to obtain the algebra $\mathbf{A}_{\mathcal{T}}$ which is *represented* by $\mathcal{T}$.



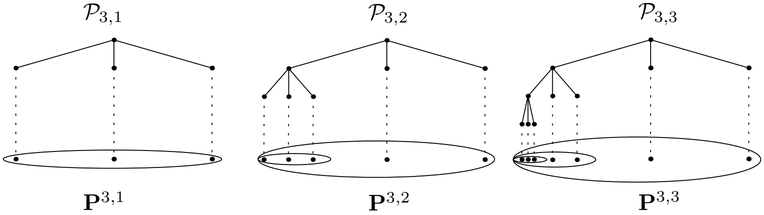
Lemma

Every conservative minority algebra $\mathbf{A}$ *is isomorphic to* $\mathbf{A}_{\mathcal{T}}$ *for some simple reduced conservative minority tree* $\mathcal{T}$.

Examples

Definition

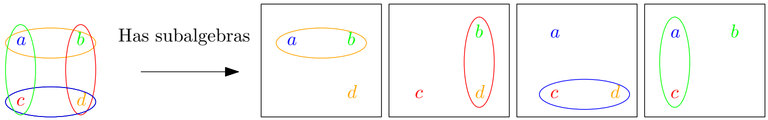
Let $\mathbf{P}_n = (\{0, \ldots, n-1\}, p_n)$ be the algebra with conservative minority operation defined on injective triples (a, b, c) as the first projection. For example, $p_n(4, 4, 5) = 5$ and $p_n(6, 3, 7) = 6$. We call $\mathbf{P}_n$ the n-element *projection minority algebra*.



Recursively define a sequence of projection minority trees $\mathcal{P}_{n,k}$ for $k \geq 1$ as above, where each 'local' algebra is isomorphic to $\mathbf{P}_n$. Denote each respectively represented algebra as $\mathbf{P}^{n,k}$. The corresponding structures pp-construct all others!

Examples

The tree structure of a conservative minority tree only stores the information about global congruences. There exist increasingly complex examples of conservative minority algebras which are simple, but not *hereditarily simple*.
Let $\mathbf{S}_4 = (\{0,1,2,3\}, m)$ be the conservative minority algebra specified as follows.



We can then build a simple conservative minority tree with 'local' algebras which are isomorphic to $\mathbf{S}_4$. In general, any conservative minority algebra is a subalgebra of a simple conservative minority algebra, so we can recursively build increasingly complicated examples.

Invariant relaions

Kearnes and Szendrei studied the invariant relations of algebras with a *parallelogram term* (Maltsev operations are the most basic examples) and characterized all *critical relations* of sufficiently high arity using commutator theory. We can specialize their theory of critical relations to our setting to obtain a nice set of relations which pp-defines $\mathrm{Inv}(\mathbf{A})$ for any conservative minority $\mathbf{A}$.

Definition (Kearnes, Szendrei)

Let $\mathbf{A}$ be a finite algebra and let $R \leq \mathbf{A}^n$ be an element of $\mathrm{Inv}(\mathbf{A})$. Then R is called *critical* if it has no dummy coordinates and is meet irreducible in the lattice of subalgebras of $\mathbf{A}^n$.

Lemma

A conservative minority algebra is subdirectly irreducible (unique atomic congruence called the monolith) if and only if its conrguences are linearly ordered by inclusion if and only if every congruence is a block congruence.

Theorem

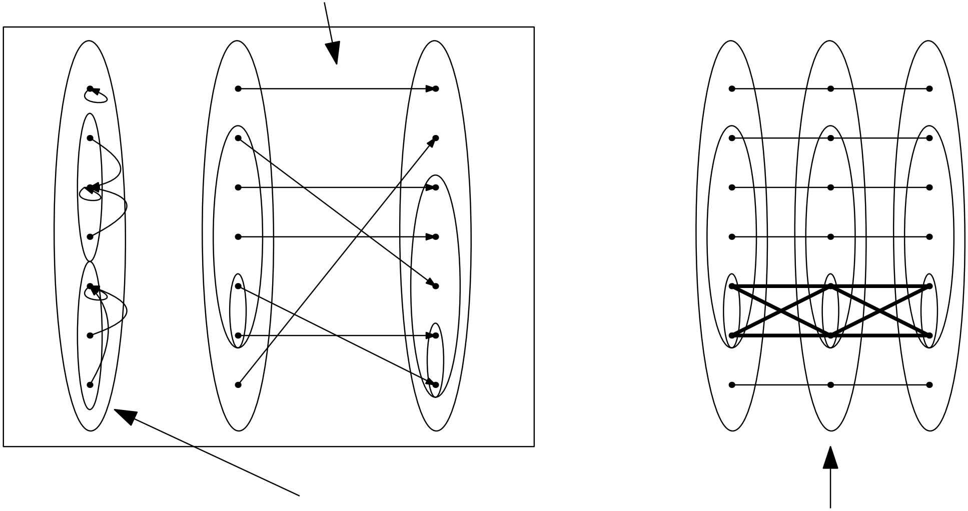
Let $\mathbf{A} = (A; f)$ be a finite conservative minority algebra. Then for any $R \subseteq A^k$ the following are equivalent.

1. *R is preserved by f.*
2. *R has a primitive positive definition from the following finite set of at most ternary relations:*
   - 2.1 *All unary relations $X \subseteq A$,*
   - 2.2 *All subdirect transversal endomorphism graphs $S \leq \mathbf{B} \times \mathbf{C}$ for $\mathbf{C} \leq \mathbf{B} \leq \mathbf{A}$ and $\mathbf{C}$ subdirectly irreducible,*
   - 2.3 *All isomorphism graphs $G \leq \mathbf{A}_1 \times \mathbf{A}_2$ where $\mathbf{A}_1, \mathbf{A}_2 \leq \mathbf{A}$ are subdirectly irreducible,*
   - 2.4 *For each subdirectly irreducible $\mathbf{C} \leq \mathbf{A}$ with a monolith whose nontrivial class has exactly two elements, the ternary relation*

$$\mathrm{Lin}_{\mathbf{C}} := \big\{(x, y, z) \in \{0, 1\}^3 \mid (x, y, z) \text{ is a solution to the } \mathbb{Z}_2\text{-linear equation}\big\}$$
$$x + y + z = 1 \cup \big\{(c, c, c) : c \in C \setminus \{0, 1\}\big\},$$

where $\{0, 1\}$ is some labeling of the nontrivial class of the monolith of $\mathbf{C}$.

Isomorphism between two SI subalgebras



Transversal endomorphism graph of subalgebra

Linear relations

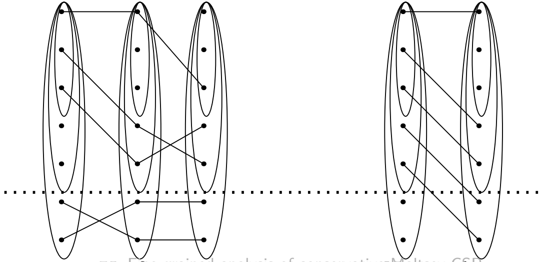
Solving the CSP for the subdirectly irreducible $\mathbb{P}^{n,k}$

The structures $\mathfrak{P}^{n,k}$

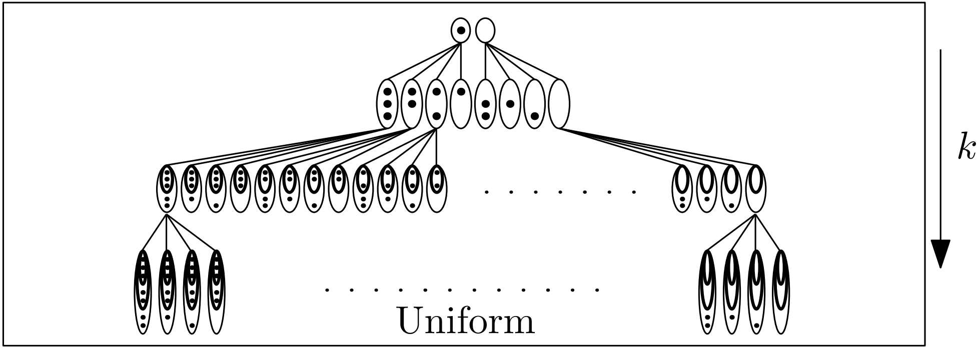
Fix $n \geq 2$ and recursively build structures $\mathfrak{P}^{n,k}$ so that $\mathrm{Inv}(\mathbf{P}^{n,k}) = \mathrm{Inv}(\mathrm{Pol}(\mathfrak{P}^{n,k}))$. We distinguish between *uniform* and *nonuniform* relations.

Definition

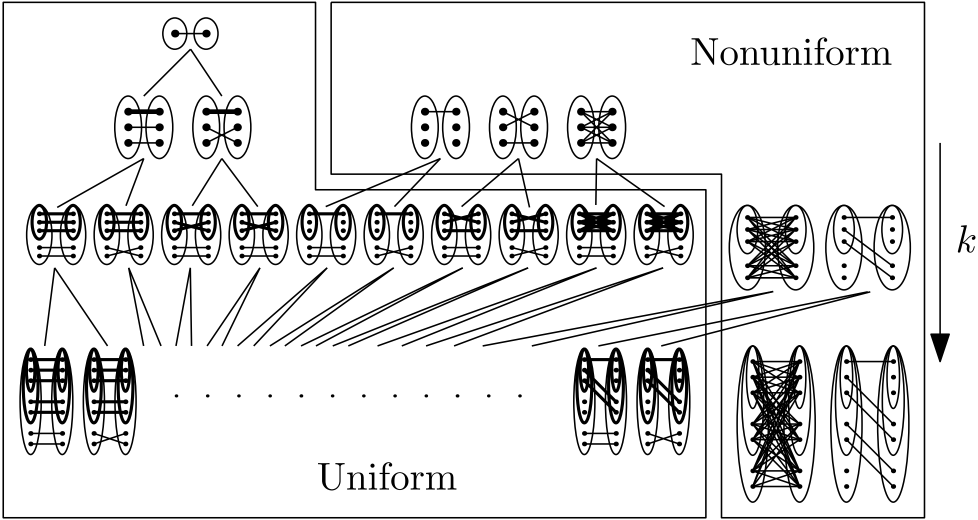
Let $n \geq 2$ and $k \geq 1$. We say that $R \leq \mathbf{P}^{n,k}$ is *uniform* if no tuple in R contains entries which both belong to the maximal congruence of $\mathbf{P}^{n,k}$ and its complement. If $k = 1$, we say R is uniform if it is subdirect and the only tuples in R which contain 0 are the constant $(0, \dots, 0)$ tuple.



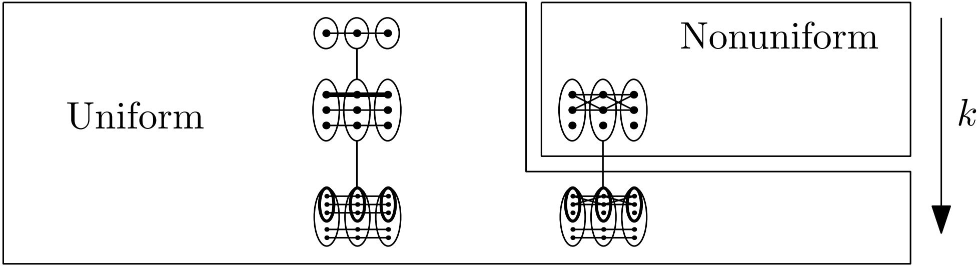
Starting with the trivial structure, we recursively generate relations for larger values of k.



Starting with the trivial structure, we recursively generate relations for larger values of k.



Starting with the trivial structure, we recursively generate relations for larger values of k.

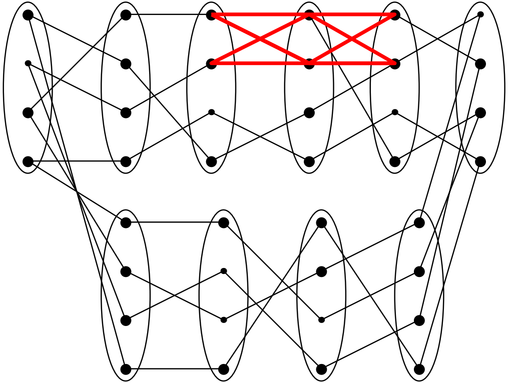


Solving $\mathrm{CSP}(\mathfrak{P}^{n,k})$

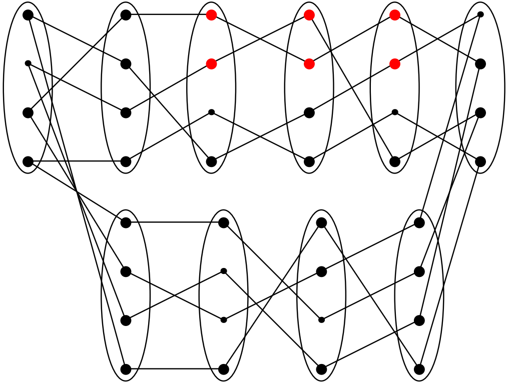
- To solve the CSP for the structures $\mathfrak{P}^{n,k}$, we recursively specify a sequence of reductions

$$\mathrm{Reduce}_{n,1}, \mathrm{Reduce}_{n,2}, \ldots, \mathrm{Reduce}_{n,k}, \ldots$$

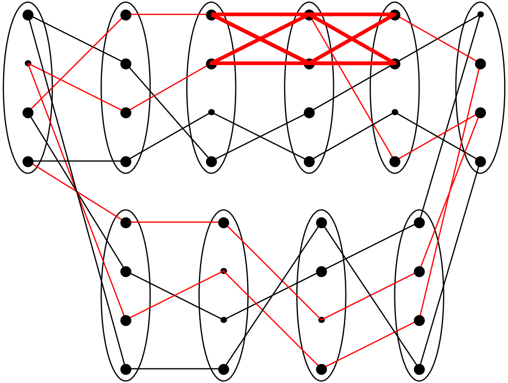
  where for an instance $\mathfrak{I} \in \mathrm{CSP}(\mathfrak{P}^{n,k})$, the instance output by $\mathrm{Reduce}_{n,k}(\mathfrak{I})$ is an instance of $\mathrm{CSP}(\mathfrak{P}^{n,k-1})$.
- For the basis, we first specify how to solve $\mathrm{CSP}(\mathfrak{P}^{n,1})$. The relations of $\mathfrak{P}^{n,1}$ are pp-interdefinable with the following relations:
  - $\{R_X : X \subseteq \{0, \ldots, n-1\}\}$ (all possible unary relations).
  - $\{R_\sigma : \ \sigma$ is a permutation of $\{0, \ldots, n-1\}$ $\}$ (bijection graphs).
  - $L(x, y, z)$ (solutions to the $\mathbb{Z}_2$-linear equation $x + y + z = 1$).



Here is a typical (connected) input instance $\mathfrak{I}$.



If we replace the linear relation L with its unary projections, then we obtain an instance with only bijection graphs and unary relations. These relations are also preserved by a majority operation, so the CSP is solvable with Datalog. By Dalmau + Larose, it is solvable in linear symmetric Datalog, which can be implemented in L.

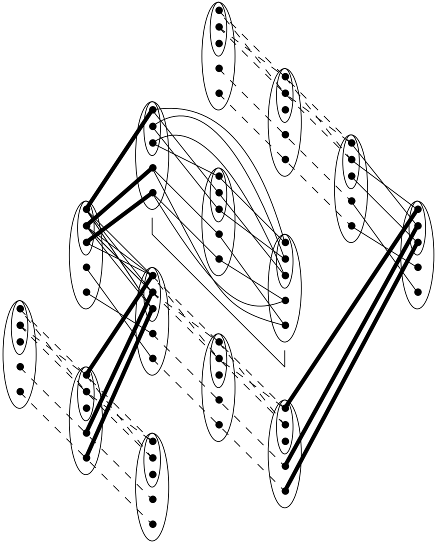


If the instance passes this first test, then we can find a system of linear equations by propagating the set $\{0, 1\}$ along bijection paths. If this system is consistent, then the instance has a solution. It is possible to find this system with linear symmetric Datalog rules, hence this system can be computed in L. Since checking if the system is consistent is in $\oplus$L, we obtain a procedure in $\oplus$L.

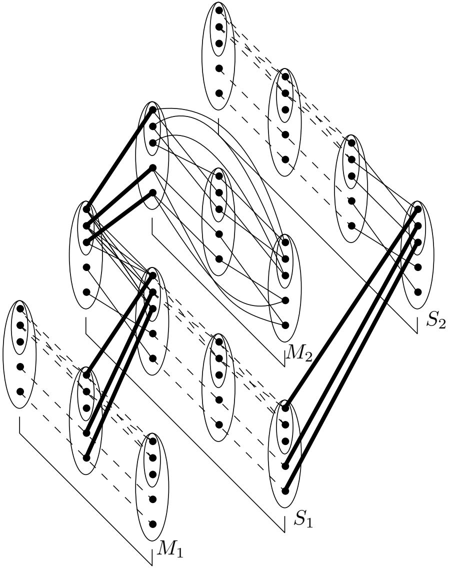
We denote the previous procedure by $\mathrm{Solve}_{n,1}$. This is called as a subprocedure in a procedure we call $\mathrm{Reduce}_{n,1}$, which does the following on an instance $\mathfrak{I}$ of $\mathrm{CSP}(\mathfrak{P}^{n,k})$:

- Find the connected components $C_1, \ldots, C_v$ of $\mathfrak{I}$ and form induced subinstances $\mathfrak{C}_1, \ldots, \mathfrak{C}_v$.
- On each connected component instance $\mathfrak{C}_i$, run $\mathrm{Solve}_{n,1}(\mathfrak{C}_i)$.
  - If the instance is satisfiable, return an instance $\overline{\mathfrak{C}_i}$ of the trivial structure which has all binary constraints and ternary constraints replaced by respectively binary and ternary $=$, and a unary constraint $R_{\{0\}}(x)$ for every variable $x \in C_i$ (this instance is trivially satisfiable).
  - If the instance is unsatisfiable, return an instance $\overline{\mathfrak{C}_i}$ of the trivial structure which has all binary constraints and ternary constraints replaced by respectively binary and ternary $=$, and a unary constraint $R_{\emptyset}(x)$ for every variable $x \in C_i$ (this instance is trivially unsatisfiable).
- Define $\overline{\mathfrak{I}}$ as the union of $\overline{\mathfrak{C}_1}, \ldots, \overline{\mathfrak{C}_v}$ and output $\overline{\mathfrak{I}}$.

Notice that $\mathrm{Reduce}_{n,1}$ can be implemented as a composition of procedures in $FL^{\oplus \mathsf{L}}$.

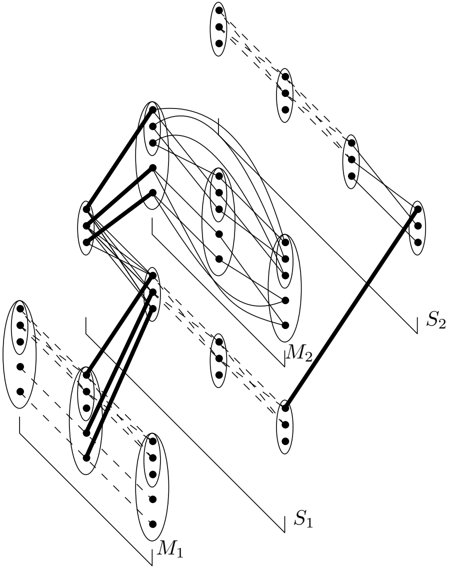


Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.



Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.

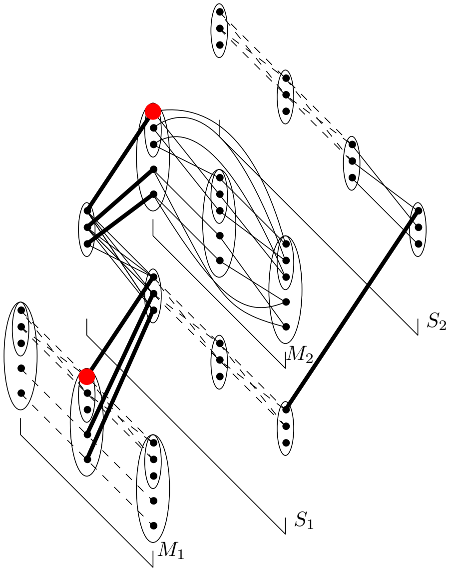
We find the connected components with respect to the uniform relations first. Then we label which are *source* components and which are not source components.



Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.

We find the connected components with respect to the uniform relations first. Then we label which are *source* components and which are not source components.

The source components can already be viewed as instances of CSP($\mathfrak{P}^{3,1}$).

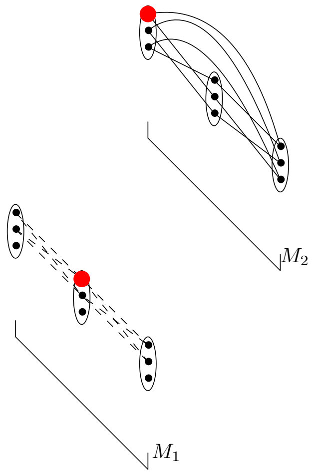


Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.

We find the connected components with respect to the uniform relations first. Then we label which are *source* components and which are not source components.

The source components can already be viewed as instances of CSP($\mathfrak{P}^{3,1}$).

For the nonsource components, we form restricted instances and run $\text{Reduce}_{n,1}$, then combine the output instance with the remaining bijection graphs below. Beforehand, we must add some unary relations which will ensure that the reduction is sound.

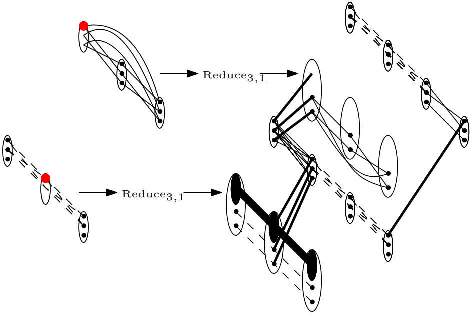


Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.

We find the connected components with respect to the uniform relations first. Then we label which are *source* components and which are not source components.

The source components can already be viewed as instances of CSP($\mathfrak{P}^{3,1}$).

For the nonsource components, we form restricted instances and run Reduce$_{n,1}$, then combine the output instance with the remaining bijection graphs below. Beforehand, we must add some unary relations which will ensure that the reduction is sound.

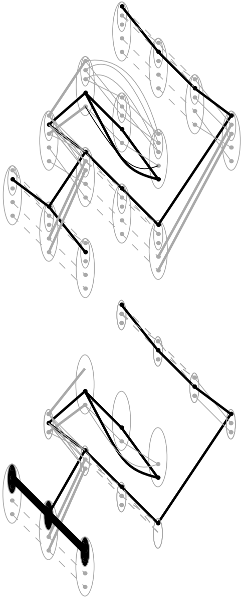


Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.

We find the connected components with respect to the uniform relations first. Then we label which are *source* components and which are not source components.

The source components can already be viewed as instances of CSP($\mathfrak{P}^{3,1}$).

For the nonsource components, we form restricted instances and run Reduce$_{n,1}$, then combine the output instance with the remaining bijection graphs below. Beforehand, we must add some unary relations which will ensure that the reduction is sound.



Now suppose we have an instance of CSP($\mathfrak{P}^{n,2}$) (here n=3). Here we have a single important nonuniform relation which is an isomorphism graph between two subdirectly irreducible subalgebras.

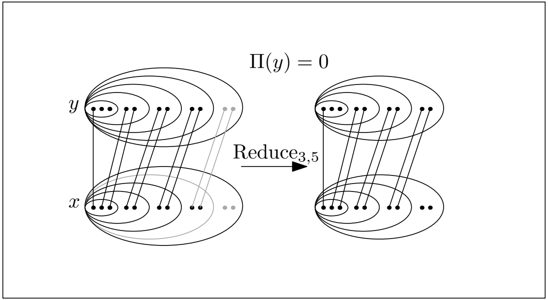
We find the connected components with respect to the uniform relations first. Then we label which are *source* components and which are not source components.

The source components can already be viewed as instances of CSP($\mathfrak{P}^{3,1}$).

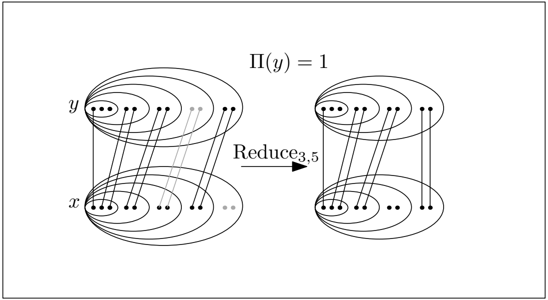
For the nonsource components, we form restricted instances and run $\text{Reduce}_{n,1}$, then combine the output instance with the remaining bijection graphs below. Beforehand, we must add some unary relations which will ensure that the reduction is sound.

The obtained instance is indeed a reduction.

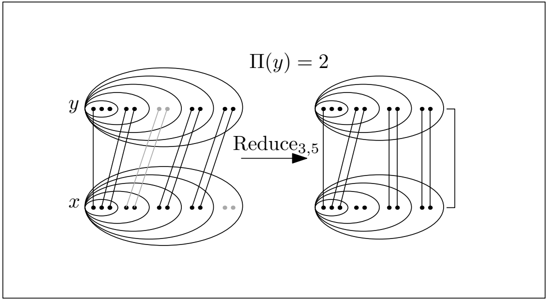
- For higher values of k:
  - Find ‘uniform‘ connected components, distinguish between source and nonsource components.
  - Source components are reduced by restricting relations to the maximal congruence block.
  - Nonsource components are reduced by recursively calling $\mathrm{Reduce}_{n,k-1}$ on the instance restricted to the maximal congruence and combining the output with the original.
  - Information provided by a function $\Pi : I \to \{0, \ldots, k-1\}$ shows how to reduce the isomorphism graph between SI subalgebras.



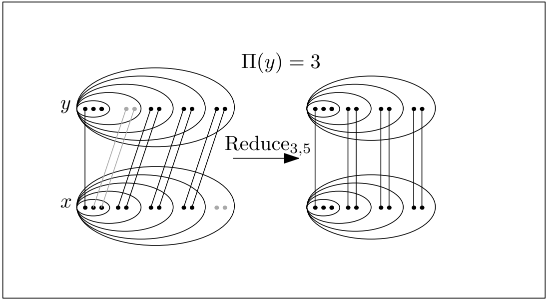
- For higher values of k:
  - Find ‘uniform‘ connected components, distinguish between source and nonsource components.
  - Source components are reduced by restricting relations to the maximal congruence block.
  - Nonsource components are reduced by recursively calling $\mathrm{Reduce}_{n,k-1}$ on the instance restricted to the maximal congruence and combining the output with the original.
  - Information provided by a function $\Pi : I \to \{0, \ldots, k-1\}$ shows how to reduce the isomorphism graph between SI subalgebras.



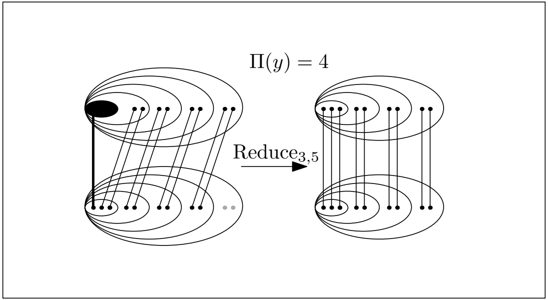
- For higher values of k:
  - Find ‘uniform‘ connected components, distinguish between source and nonsource components.
  - Source components are reduced by restricting relations to the maximal congruence block.
  - Nonsource components are reduced by recursively calling $\mathrm{Reduce}_{n,k-1}$ on the instance restricted to the maximal congruence and combining the output with the original.
  - Information provided by a function $\Pi : I \to \{0, \ldots, k-1\}$ shows how to reduce the isomorphism graph between SI subalgebras.

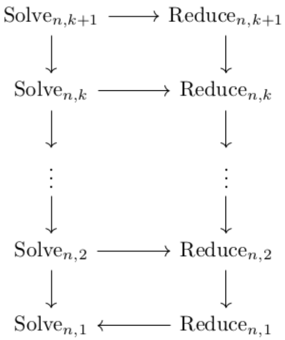


- For higher values of k:
  - Find 'uniform' connected components, distinguish between source and nonsource components.
  - Source components are reduced by restricting relations to the maximal congruence block.
  - Nonsource components are reduced by recursively calling $\mathrm{Reduce}_{n,k-1}$ on the instance restricted to the maximal congruence and combining the output with the original.
  - Information provided by a function $\Pi : I \to \{0, \ldots, k-1\}$ shows how to reduce the isomorphism graph between SI subalgebras.



- For higher values of k:
  - Find ‘uniform‘ connected components, distinguish between source and nonsource components.
  - Source components are reduced by restricting relations to the maximal congruence block.
  - Nonsource components are reduced by recursively calling $\mathrm{Reduce}_{n,k-1}$ on the instance restricted to the maximal congruence and combining the output with the original.
  - Information provided by a function $\Pi : I \to \{0, \ldots, k-1\}$ (called a *reduction atlas*) shows how to reduce the isomorphism graph between SI subalgebras.





- All of the procedures $\text{Reduce}_{n,k}$ can be implemented in an extension of (symmetric linear) Datalog which we call $\mathbb{Z}_2$-Datalog.

Thank you!