Sparsification and Running Time Aspects of CSPs

a.k.a. "Parameterizing by *n*"

Magnus Wahlström

Royal Holloway, University of London

PACS 2024

Based on joint work with Victor Lagerkvist

The overall topic

CSP(Γ)

Fixed: Constraint language Γ over domain D
Input: Formula F over Γ (conjunction of constraints from Γ) on variable set V
Question: Is there an assignment $\phi\colon V \to D$ that satisfies F?

- How does the fine-grained/exact complexity of CSP(Γ) depend on the language Γ?
  - 3-SAT $O(1.3308^n)$, k-SAT $O((2-\Theta(1/k))^n)$, 1-in-k SAT $O(1.1730^n)$ ...
- What can the algebraic methods tell us about this question?

Outline

Outline

The algebraic method

- If relations of Γ_1 can be implemented in language Γ_2 then CSP(Γ_2) is at least as hard as CSP(Γ_1)
  - 1-in-3 SAT (relation $R_{1/3} = \{(0,0,1),(0,1,0),(1,0,0)\}$) is not harder than 3-SAT:

$$R_{1/3}(x,y,z) \equiv (x \vee y \vee z) \wedge (\neg x \vee \neg y) \wedge (\neg x \vee \neg z) \wedge (\neg y \vee \neg z)$$

- If not, then Γ_2 enjoys an algebraic invariant that proves that it is structurally more restricted
  - $R_{1/3}$ is preserved by the partial Mal'tsev partial polymorphism

The algebraic method

- If relations of Γ_1 can be implemented in language Γ_2 then CSP(Γ_2) is at least as hard as CSP(Γ_1)
  - 1-in-3 SAT (relation $R_{1/3} = \{(0,0,1),(0,1,0),(1,0,0)\}$) is not harder than 3-SAT:

$$R_{1/3}(x,y,z) \equiv (x \vee y \vee z) \wedge (\neg x \vee \neg y) \wedge (\neg x \vee \neg z) \wedge (\neg y \vee \neg z)$$

- If not, then Γ_2 enjoys an algebraic invariant that proves that it is structurally more restricted
  - $R_{1/3}$ is preserved by the partial Mal'tsev partial polymorphism

(Partial/total) polymorphisms

Let $R \subseteq D^n$ be a relation over domain D

Polymorphisms

Polymorphism of R: Operation $D^r \to D$ such that if $t_1, \ldots, t_r \in R$ then $p(t_1, \ldots, t_r) \in R$, e.g. 3-way XOR:

t_1	0	0	1	$\in R$
t_2	0	1	0	$\in R$
t_3	1	0	0	$\in R$
$p(t_1, t_2, t_3)$	1	1	1	$\in R$

Partial polymorphisms

Partial polymorphism of R: Partial operation $D^r \to D$ such that for $t_1, \ldots, t_R \in R$, if $p(t_1, \ldots, t_r)$ is defined then $p(t_1, \ldots, t_r) \in R$, e.g.

t_1	0	0	1	$\in R$
t_2	0	1	0	$\in R$
t_3	1	0	0	$\in R$
$p(t_1, t_2, t_3)$	1	$\bot$	1	???

...does not give any conclusions.

(Partial/total) polymorphisms

Let $R \subseteq D^n$ be a relation over domain D

Polymorphisms

Polymorphism of R: Operation $D^r \to D$ such that if $t_1, \ldots, t_r \in R$ then $p(t_1, \ldots, t_r) \in R$, e.g. 3-way XOR:

t_1	0	0	1	$\in R$
t_2	0	1	0	$\in R$
t_3	1	0	0	$\in R$
$p(t_1, t_2, t_3)$	1	1	1	$\in R$

Partial polymorphisms

Partial polymorphism of R: Partial operation $D^r \to D$ such that for $t_1, \ldots, t_R \in R$, if $p(t_1, \ldots, t_r)$ is defined then $p(t_1, \ldots, t_r) \in R$, e.g.

t_1	0	0	1	$\in R$
t_2	0	1	0	$\in R$
t_3	1	0	0	$\in R$
$p(t_1, t_2, t_3)$	1	$\perp$	1	???

...does not give any conclusions.

Algebraic invariant notions

Notions of implementations and invariants over a language Γ:

Invariant	**Implementations**	**Implementation operations**
Polymorphisms	pp-definitions	$\exists Y : R_1(X_1, Y_1) \wedge \ldots \wedge R_m(X_m, Y_m)$ $(X_i \subseteq X,\ Y_i \subseteq Y,\ R_i \in \Gamma)$
Classes of polymorphisms (e.g. p such that $p(x,x,y) = p(y,x,x) = y$)	pp-constructions	Implementations across domains
Partial polymorphisms	qfpp-definitions	$R_1(X_1) \wedge \ldots \wedge R_m(X_m)$, $(X_i \subseteq X,\ R_i \in \Gamma)$

E.g.:

A language Γ can implement a relation R using a qfpp-definition if and only if every partial polymorphism of Γ also preserves R

Algebraic invariant notions

Notions of implementations and invariants over a language Γ:

Invariant	**Implementations**	**Implementation operations**
Polymorphisms	pp-definitions	$\exists Y : R_1(X_1, Y_1) \wedge \ldots \wedge R_m(X_m, Y_m)$ $(X_i \subseteq X,\ Y_i \subseteq Y,\ R_i \in \Gamma)$
Classes of polymorphisms (e.g. p such that $p(x,x,y) = p(y,x,x) = y$)	pp-constructions	Implementations across domains
Partial polymorphisms	qfpp-definitions	$R_1(X_1) \wedge \ldots \wedge R_m(X_m)$, $(X_i \subseteq X,\ R_i \in \Gamma)$

E.g.:

A language Γ can implement a relation R using a qfpp-definition if and only if every partial polymorphism of Γ also preserves R

Algebraic invariant notions

Notions of implementations and invariants over a language Γ:

Invariant	**Implementations**	**Implementation operations**
Polymorphisms	pp-definitions	$\exists Y: R_1(X_1, Y_1) \wedge \ldots \wedge R_m(X_m, Y_m)$ $(X_i \subseteq X,\ Y_i \subseteq Y,\ R_i \in \Gamma)$
Classes of polymorphisms (e.g. p such that $p(x,x,y) = p(y,x,x) = y$)	pp-constructions	Implementations across domains
Partial polymorphisms	qfpp-definitions	$R_1(X_1) \wedge \ldots \wedge R_m(X_m)$, $(X_i \subseteq X,\ R_i \in \Gamma)$

E.g.:

A language Γ can implement a relation R using a qfpp-definition if and only if every partial polymorphism of Γ also preserves R

Algebraic invariant notions

Notions of implementations and invariants over a language Γ:

Invariant	**Implementations**	**Implementation operations**
Polymorphisms	pp-definitions	$\exists Y: R_1(X_1, Y_1) \wedge \ldots \wedge R_m(X_m, Y_m)$ $(X_i \subseteq X,\ Y_i \subseteq Y,\ R_i \in \Gamma)$
Classes of polymorphisms (e.g. p such that $p(x,x,y) = p(y,x,x) = y$)	pp-constructions	Implementations across domains
Partial polymorphisms	qfpp-definitions	$R_1(X_1) \wedge \ldots \wedge R_m(X_m)$, $(X_i \subseteq X,\ R_i \in \Gamma)$

E.g.:

A language Γ can implement a relation R using a qfpp-definition if and only if every partial polymorphism of Γ also preserves R

Outline

CSP(Γ)

Fixed: Constraint language Γ over domain D
Input: Formula F over Γ (conjunction of constraints from Γ) on variable set V
Question: Is there an assignment $\phi\colon V \to D$ that satisfies F?

- Assume CSP(Γ) is NP-hard. How does the running time depend on Γ?
  - ETH: Time $O^*(c^n)$ for some $c = c(\Gamma)$ – but which c?
- Example **3-SAT**:

$$O(1.84^n) \Rightarrow O(1.62^n) \Rightarrow \ldots \Rightarrow O((4/3)^n) \Rightarrow \ldots \Rightarrow O(1.308^n) \ldots$$

- Questions:
  1. How does the value $c(\Gamma)$ depend on Γ?
  2. For which languages is $c(\Gamma)$ non-trivial, i.e. $c(\Gamma) < |D|$?

CSP(Γ)

Fixed: Constraint language Γ over domain D
Input: Formula F over Γ (conjunction of constraints from Γ) on variable set V
Question: Is there an assignment $\phi\colon V \to D$ that satisfies F?

- Assume CSP(Γ) is NP-hard. How does the running time depend on Γ?
  - ETH: Time $O^*(c^n)$ for some $c = c(\Gamma)$ – but which c?
- Example **3-SAT**:

$$O(1.84^n) \Rightarrow O(1.62^n) \Rightarrow \ldots \Rightarrow O((4/3)^n) \Rightarrow \ldots \Rightarrow O(1.308^n) \ldots$$

- Questions:
  1. How does the value $c(\Gamma)$ depend on Γ?
  2. For which languages is $c(\Gamma)$ non-trivial, i.e. $c(\Gamma) < |D|$?

Problems with partial polymorphisms

The Question

Can we use algebraic methods to study the running time of CSP(Γ)?

- Good news: $c(\Gamma)$ depends only on the partial polymorphisms of Γ (pPol(Γ))
- Problem: pPol(Γ) has awful structure
  - Uncountably many classes even for $D = \{0, 1\}$
  - A finite NP-hard language Γ needs* an infinite number of partial polymorphisms to define it
  - Any* finite set of partial polymorphisms preserves $2^{2^{\Theta(n)}}$ relations of arity n
- Study 3-SAT via pPol(3-CNF): not so much
- To make progress, we study languages defined via their partial polymorphisms

Problems with partial polymorphisms

The Question

Can we use algebraic methods to study the running time of CSP(Γ)?

- Good news: $c(\Gamma)$ depends only on the partial polymorphisms of Γ (pPol(Γ))
- Problem: pPol(Γ) has awful structure
  - Uncountably many classes even for $D = \{0, 1\}$
  - A finite NP-hard language Γ needs* an infinite number of partial polymorphisms to define it
  - Any* finite set of partial polymorphisms preserves $2^{2^{\Theta(n)}}$ relations of arity n
- Study 3-SAT via pPol(3-CNF): not so much
- To make progress, we study languages defined via their partial polymorphisms

Problems with partial polymorphisms

The Question

Can we use algebraic methods to study the running time of CSP(Γ)?

- Good news: $c(\Gamma)$ depends only on the partial polymorphisms of Γ (pPol(Γ))
- Problem: pPol(Γ) has awful structure
  - Uncountably many classes even for $D = \{0, 1\}$
  - A finite NP-hard language Γ needs* an infinite number of partial polymorphisms to define it
  - Any* finite set of partial polymorphisms preserves $2^{2^{\Theta(n)}}$ relations of arity n
- Study 3-SAT via pPol(3-CNF): not so much
- To make progress, we study languages defined via their partial polymorphisms

Pattern partial polymorphisms

- Recall classes of polymorphisms, e.g.:
  1. Majority operation: $m\colon D^3 \to D$ such that

$$m(x,x,y) = m(x,y,x) = m(y,x,x) = x \quad \text{for all } x,y \in D$$

  2. Mal'tsev operation: $m\colon D^3 \to D$ such that

$$m(x,x,y) = m(y,x,x) = y \quad \text{for all } x,y$$

  e.g. $m(x,y,z) = x - y + z$

- Partial Mal'tsev operation: $m\colon D^3 \to D$ such that

$$m(x,y,z) = \begin{cases} z & x = y \\ x & y = z \\ \bot & \text{otherwise} \end{cases}$$

- Partial majority, partial k-NU, ...

Pattern partial polymorphisms

- Recall classes of polymorphisms, e.g.:
  1. Majority operation: $m\colon D^3 \to D$ such that
  $$m(x,x,y) = m(x,y,x) = m(y,x,x) = x \quad \text{for all } x,y \in D$$
  2. Mal'tsev operation: $m\colon D^3 \to D$ such that
  $$m(x,x,y) = m(y,x,x) = y \quad \text{for all } x,y$$
  e.g. $m(x,y,z) = x - y + z$
- Partial Mal'tsev operation: $m\colon D^3 \to D$ such that
$$m(x,y,z) = \begin{cases} z & x = y \\ x & y = z \\ \bot & \text{otherwise} \end{cases}$$
- Partial majority, partial k-NU, ...

Results sample – upper bounds

Algorithms from partial polymorphisms

- Γ has partial Mal'tsev ppol: CSP(Γ) solved by meet-in-the-middle, $O(|D|^{n/2})$
- Γ has partial majority ppol: CSP(Γ) solved by fast matrix multiplication, $O(|D|^{(\omega/3)n})$
- Boolean domian, partial k-NU ppol $\Rightarrow$ local search algorithms (conjecturally)

- Total Mal'tsev polymorphism: "behaves like linear equations"
- Total majority polymorphism: tractable binary language (e.g. 2-SAT)
- Results 1–2 apply in appropriate oracle model

Question

Does any non-trivial pattern partial polymorphism imply that CSP(Γ) is solvable in $O^*(c^n)$ time for some $c < |D|$? (Non-uniform algorithms are fine.)

Results sample – upper bounds

Algorithms from partial polymorphisms

- Γ has partial Mal'tsev ppol: CSP(Γ) solved by meet-in-the-middle, $O(|D|^{n/2})$
- Γ has partial majority ppol: CSP(Γ) solved by fast matrix multiplication, $O(|D|^{(\omega/3)n})$
- Boolean domian, partial k-NU ppol $\Rightarrow$ local search algorithms (conjecturally)

- Total Mal'tsev polymorphism: "behaves like linear equations"
- Total majority polymorphism: tractable binary language (e.g. 2-SAT)
- Results 1–2 apply in appropriate oracle model

Question

Does any non-trivial pattern partial polymorphism imply that CSP(Γ) is solvable in $O^*(c^n)$ time for some $c < |D|$? (Non-uniform algorithms are fine.)

Results sample – lower bounds

Complexity of inv(p)-SAT

Let p be a "purely partial" polymorphism (which does not imply any total polymorphism).

Let inv(p)-SAT be the SAT problem where the constraints can use any relation R that is preserved by p (of unbounded arity) given in some suitable white-box representation.

Then there is a constant $c_p > 1$ such that under SETH, no algorithm can solve inv(p)-SAT in time $O((c_p - \varepsilon)^n)$ for any $\varepsilon > 0$.

- We can pad any relation $R(X) \subseteq \{0,1\}^n$ to a relation

$$R'(X, X' = f(X)) \subseteq \{0,1\}^{\alpha n}$$

  such that R' is preserved by p, for some $\alpha > 1$.
- So SAT on n variables reduces to inv(p)-SAT on αn variables.

Results sample – lower bounds

Complexity of inv(p)-SAT

Let p be a "purely partial" polymorphism (which does not imply any total polymorphism).

Let inv(p)-SAT be the SAT problem where the constraints can use any relation R that is preserved by p (of unbounded arity) given in some suitable white-box representation.

Then there is a constant $c_p > 1$ such that under SETH, no algorithm can solve inv(p)-SAT in time $O((c_p - \varepsilon)^n)$ for any $\varepsilon > 0$.

- We can pad any relation $R(X) \subseteq \{0,1\}^n$ to a relation

$$R'(X, X' = f(X)) \subseteq \{0,1\}^{\alpha n}$$

  such that R' is preserved by p, for some $\alpha > 1$.
- So SAT on n variables reduces to inv(p)-SAT on αn variables.

inv(p)-SAT – summary

Let's stay with $D = \{0, 1\}$.

- For one p we have both upper and lower bounds:
  - Partial Mal'tsev solved in $O^*(2^{n/2})$ time, not in $O(2^{n/7.29})$ time under SETH
- We can also study "inv(p) analogues" of problems we care about
  - Partial k-NU operation ($k \geq 4$) contains $(k-1)$-SAT, not k-SAT
  - inv(nu_k) has a SETH lower bound of $(2 - \Theta(\log k/k))^n$
  - inv(nu_4) has a SETH lower bound of $2^{n/5.9} \approx 1.125^n$

Questions

1. Can the gap for partial Mal'tsev be closed?
2. Can the lower bound for inv(nu_k) be lifted to $2 - \Theta(1/k)$ to match the conjectured k-SAT behaviour?

inv(p)-SAT – summary

Let's stay with $D = \{0, 1\}$.

- For one p we have both upper and lower bounds:
  - Partial Mal'tsev solved in $O^*(2^{n/2})$ time, not in $O(2^{n/7.29})$ time under SETH
- We can also study "inv(p) analogues" of problems we care about
  - Partial k-NU operation ($k \geq 4$) contains $(k-1)$-SAT, not k-SAT
  - inv(nu_k) has a SETH lower bound of $(2 - \Theta(\log k/k))^n$
  - inv(nu_4) has a SETH lower bound of $2^{n/5.9} \approx 1.125^n$

Questions

1. Can the gap for partial Mal'tsev be closed?
2. Can the lower bound for inv(nu_k) be lifted to $2 - \Theta(1/k)$ to match the conjectured k-SAT behaviour?

Interlude - fgpp-definitions

Clement Carbonnel 2022

A functionally guarded pp-definition (fgpp-definition) is a definition

$$R(x_1, \ldots, x_n) \equiv \exists (y_1 = f_1(x_{i_1})), \ldots, (y_t = f_t(x_{i_t})) : R_1(X_1, Y_1) \wedge \ldots \wedge R_m(X_m, Y_m)$$

where $f_i \colon D \to D$ are arbitrary functions.

Pattern partial polymorphisms precisely characterize the expressive power under fgpp-definitions

- This appears to work between domains too (cf. pp-constructions)
- Does this give a vehicle to study fine-grained ($O(n^c)$) problem complexity?
  - Zero-weight triangle, min-weight triangle, Orthogonal Vectors, (k, ℓ)-hyperclique . . . . . .

Outline

Sparsification and non-redundancy

Sparsification

A sparsification for CSP(Γ) is a kernelization with parameter n:

- A polynomial-time reduction that maps an instance I on n variables to an instance I' of total size $f(n)$ that is a yes-instance if and only if I is a yes-instance

Non-redundancy

A language Γ has non-redundancy $f(n)$ if every formula F over Γ with n variables has a subformula $F' \subseteq F$ such that

1. F and F' have identical solution spaces
2. F' contains at most $f(n)$ constraints

- The questions turn out to be practically the same (for NP-hard problems)
- What can we say about sparsification/non-redundancy bounds?

Sparsification and non-redundancy

Sparsification

A sparsification for CSP(Γ) is a kernelization with parameter n:

- A polynomial-time reduction that maps an instance I on n variables to an instance I' of total size $f(n)$ that is a yes-instance if and only if I is a yes-instance

Non-redundancy

A language Γ has non-redundancy $f(n)$ if every formula F over Γ with n variables has a subformula $F' \subseteq F$ such that

1. F and F' have identical solution spaces
2. F' contains at most $f(n)$ constraints

- The questions turn out to be practically the same (for NP-hard problems)
- What can we say about sparsification/non-redundancy bounds?

Sparsification and non-redundancy

Sparsification

A sparsification for CSP(Γ) is a kernelization with parameter n:

- A polynomial-time reduction that maps an instance I on n variables to an instance I' of total size $f(n)$ that is a yes-instance if and only if I is a yes-instance

Non-redundancy

A language Γ has non-redundancy $f(n)$ if every formula F over Γ with n variables has a subformula $F' \subseteq F$ such that

1. F and F' have identical solution spaces
2. F' contains at most $f(n)$ constraints

- The questions turn out to be practically the same (for NP-hard problems)
- What can we say about sparsification/non-redundancy bounds?

Sparsification – fun facts

- k-SAT has no sparsification to size $O(n^{k-\varepsilon})$, $\varepsilon > 0$ unless PH collapses (Dell, van Melkebeek 2010)
- But every other Boolean language of arity k has sparsification/non-redundancy of $O(n^{k-1})$ or better, using algebraic encodings (Chen, Jansen, Pieterse 2020)
- Some languages, e.g. 1-in-k-SAT, reduce to $O(n)$ size
- For every rational number $p/q \geq 1$, there is a language Γ (over some domain D) such that (Jansen, unpublished?)
  1. CSP(Γ) can be sparsified to bitsize $O(n^{p/q})$
  2. CSP(Γ) cannot be sparsified to bitsize $O(n^{p/q-\varepsilon})$, for any $\varepsilon > 0$ unless PH collapses

Question [LW 2017; CJP 2020; Carbonnel 2022]

Can we (finally!) characterise the languages Γ which allow for $O(n)$-size sparsification? Even in the Boolean domain?

Sparsification – fun facts

- k-SAT has no sparsification to size $O(n^{k-\varepsilon})$, $\varepsilon > 0$ unless PH collapses (Dell, van Melkebeek 2010)
- But every other Boolean language of arity k has sparsification/non-redundancy of $O(n^{k-1})$ or better, using algebraic encodings (Chen, Jansen, Pieterse 2020)
- Some languages, e.g. 1-in-k-SAT, reduce to $O(n)$ size
- For every rational number $p/q \geq 1$, there is a language Γ (over some domain D) such that (Jansen, unpublished?)
  1. CSP(Γ) can be sparsified to bitsize $O(n^{p/q})$
  2. CSP(Γ) cannot be sparsified to bitsize $O(n^{p/q-\varepsilon})$, for any $\varepsilon > 0$ unless PH collapses

Question [LW 2017; CJP 2020; Carbonnel 2022]

Can we (finally!) characterise the languages Γ which allow for $O(n)$-size sparsification? Even in the Boolean domain?

Sparsification – fun facts

- k-SAT has no sparsification to size $O(n^{k-\varepsilon})$, $\varepsilon > 0$ unless PH collapses (Dell, van Melkebeek 2010)
- But every other Boolean language of arity k has sparsification/non-redundancy of $O(n^{k-1})$ or better, using algebraic encodings (Chen, Jansen, Pieterse 2020)
- Some languages, e.g. 1-in-k-SAT, reduce to $O(n)$ size
- For every rational number $p/q \geq 1$, there is a language Γ (over some domain D) such that (Jansen, unpublished?)
  1. CSP(Γ) can be sparsified to bitsize $O(n^{p/q})$
  2. CSP(Γ) cannot be sparsified to bitsize $O(n^{p/q-\varepsilon})$, for any $\varepsilon > 0$ unless PH collapses

Question [LW 2017; CJP 2020; Carbonnel 2022]

Can we (finally!) characterise the languages Γ which allow for $O(n)$-size sparsification? Even in the Boolean domain?

Non-trivial sparsification, non-Boolean domain

Carbonnel 2022

For every language Γ with constraints of arity r, either

1. Γ fgpp-defines r-clauses and has only the trivial non-redundancy bound $O(n^r)$, or
2. Γ has a non-trivial pattern partial polymorphism and has sparsification and non-redundancy to size $O(n^{r-\varepsilon})$ where $\varepsilon = 2^{1-r}$

1. Observation: fgpp-definitions preserve non-redundancy bounds up to a constant factor
2. In the presence of a pattern partial polymorphism, use results from extremal hypergraph theory to reduce a formula to a non-trivial "basis"

Non-trivial sparsification, non-Boolean domain

Carbonnel 2022

For every language Γ with constraints of arity r, either

1. Γ fgpp-defines r-clauses and has only the trivial non-redundancy bound $O(n^r)$, or
2. Γ has a non-trivial pattern partial polymorphism and has sparsification and non-redundancy to size $O(n^{r-\varepsilon})$ where $\varepsilon = 2^{1-r}$

1. Observation: fgpp-definitions preserve non-redundancy bounds up to a constant factor
2. In the presence of a pattern partial polymorphism, use results from extremal hypergraph theory to reduce a formula to a non-trivial "basis"