The algebraic method for Constraint Satisfaction Problems

LAC 2018

Institute of Mathematics for Industry, Kyushu University and La Trobe University

Marcel Jackson

Constraints and satisfaction

Constraint

A tuple of variables, and a target relation on some domain

Constraint satisfaction problem

Given some constraints, can they be satisfied?

3SAT

Conjunction of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \wedge \neg x_2 \wedge \neg x_4) \wedge \ldots$$

Can the instance be satisfied?

3SAT

Conjunction of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \wedge \neg x_2 \wedge \neg x_4) \wedge \ldots$$

Can the instance be satisfied?

- ▷ The quintessential NP-complete classic

a classic catch by John Dyson 1981

3SAT

Conjunction of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \wedge \neg x_2 \wedge \neg x_4) \wedge \ldots$$

Can the instance be satisfied?

As a CSP

Each clause is a constraint:

- Clause $(\neg x_1 \vee x_2 \vee \neg x_3)$ means (x_1, x_2, x_3) must lie in

$$\{000, 001, 010, 011, 100, \sout{101}, 110, 111\}$$

Not-all-equal 3SAT

Conjunction of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \wedge \neg x_2 \wedge \neg x_4) \wedge \ldots$$

Can the instance be satisfied with each clause containing a true literal and a false literal?

Not-all-equal 3SAT

Conjunction of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \wedge \neg x_2 \wedge \neg x_4) \wedge \ldots$$

Can the instance be satisfied with each clause containing a true literal and a false literal?

- ▷ Another well-known NP-complete classic.

A Warrick Capper classic

Not-all-equal 3SAT

Conjunction of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \wedge \neg x_2 \wedge \neg x_4) \wedge \ldots$$

Can the instance be satisfied with each clause containing a true literal and a false literal?

As a CSP

Each clause is a constraint:

- Clause $(\neg x_1 \vee x_2 \vee \neg x_3)$ means (x_1, x_2, x_3) must lie in

$$\{000, 001, \cancel{010}, 011, 100, \cancel{101}, 110, 111\}$$

Solvability of linear equations

A system of equations over $\mathbb{Z}_2$:

$$
\begin{array}{ccccccccc}
x_1 & + & x_2 & & & +x_4 & = & 1 \\
 & & x_2 & + & x_3 & +x_4 & = & 1 \\
x_1 & & & + & x_3 & +x_4 & = & 0 \\
x_1 & & & & & +x_4 & = & 1
\end{array}
$$

Solvability of linear equations

A system of equations over $\mathbb{Z}_2$:

$$\begin{array}{ccccccccc} x_1 & + & x_2 & & & +x_4 & = & 1 \\ & & x_2 & + & x_3 & +x_4 & = & 1 \\ x_1 & & & + & x_3 & +x_4 & = & 0 \\ x_1 & & & & & +x_4 & = & 1 \end{array}$$

- Easily solved in polynomial time using Gaussian elimination. It has its own complexity class $\oplus\mathtt{L}$ (“parity $\mathtt{L}$”)

Solvability of linear equations

A system of equations over $\mathbb{Z}_2$:

$$\begin{array}{ccccccccc} x_1 & + & x_2 & & & +x_4 & = & 1 \\ & & x_2 & + & x_3 & +x_4 & = & 1 \\ x_1 & & & + & x_3 & +x_4 & = & 0 \\ x_1 & & & & & +x_4 & = & 1 \end{array}$$

As a CSP

Each equation is a constraint:

- Equation $x_2 + x_3 + x_4 = 1$ means (x_2, x_3, x_4) is constrained to be in

$$\{100, 010, 001, 111\}$$

Directed graph unreachability

A directed graph, a "start" vertex s and a "finish" vertex t. Is there *no* directed path from s to t?

Directed graph unreachability

A directed graph, a “start” vertex s and a “finish” vertex t. Is there *no* directed path from s to t?

- Easily solved in polynomial time (and nondeterministic logspace).
- A fundamental computational problem in computational complexity

Directed graph unreachability

A directed graph, a “start” vertex s and a “finish” vertex t. Is there *no* directed path from s to t?

As a CSP

Each edge is a constraint:

- (u, v) means (u, v) is constrained to be in $\{00, 01, 11\}$ (that is, $\leq$ on $0, 1$)

Directed graph unreachability

A directed graph, a "start" vertex s and a "finish" vertex t. Is there *no* directed path from s to t?

As a CSP

Each edge is a constraint:

- (u, v) means (u, v) is constrained to be in $\{00, 01, 11\}$ (that is, $\leq$ on $0, 1$)

AND:

- s is constrained to be 1 while t is constrained to be 0

Schaefer's Theorem

So far, all these problems have domain $0, 1$.

Schaefer's Theorem (1979)

A Boolean satisfiability problem is either solvable in $\mathtt{P}$ or $\mathtt{NP}$-complete

Graph colouring

Given a graph $G = (V, E)$, can we colour the vertices V by $\{0, 1, 2\}$ so that adjacent vertices have different colours?

Graph colouring

Given a graph $G = (V, E)$, can we colour the vertices V by $\{0, 1, 2\}$ so that adjacent vertices have different colours?

- Yet another classic $\mathtt{NP}$-complete problem

Graph colouring

Given a graph $G = (V, E)$, can we colour the vertices V by $\{0, 1, 2\}$ so that adjacent vertices have different colours?

As a CSP

Each edge is a constraint:

- (u, v) means (u, v) is constrained to be in $\{01, 10, 02, 20, 12, 21\}$ (the $\neq$ relation on $\{0, 1, 2\}$)

Graph colouring

Given a graph $G = (V, E)$, can we colour the vertices V by $\{0, 1, 2\}$ so that adjacent vertices have different colours?

As a CSP

Each edge is a constraint:

- ▷ (u, v) means (u, v) is constrained to be in $\{01, 10, 02, 20, 12, 21\}$ (the $\neq$ relation on $\{0, 1, 2\}$)

The target domain and relations are fixed: “template”

Graph colouring

Given a graph $G = (V, E)$, can we colour the vertices V by $\{0, 1, 2\}$ so that adjacent vertices have different colours?

As a CSP

Each edge is a constraint:

- (u, v) means (u, v) is constrained to be in $\{01, 10, 02, 20, 12, 21\}$ (the $\neq$ relation on $\{0, 1, 2\}$)

The target domain and relations are fixed: "template"

Database example

Conjunctive database queries (the database is the template, the query the instance)

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \setminus \mathtt{P}$ that are not $\mathtt{NP}$-complete.

. . . but in practice there seem to be few natural problems that appear to have this intermediate status

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \backslash \mathtt{P}$ that are not $\mathtt{NP}$-complete.

Feder and Vardi (1994)

(Roughly) there is a largest logically definable subclass of $\mathtt{NP}$ in which Ladner's Theorem *might not* hold

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \setminus \mathtt{P}$ that are not $\mathtt{NP}$-complete.

Feder and Vardi (1994)

(Roughly) there is a largest logically definable subclass of $\mathtt{NP}$ in which Ladner's Theorem *might not* hold,

- ▷ it's the fixed finite template CSPs!

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \backslash \mathtt{P}$ that are not $\mathtt{NP}$-complete.

Feder and Vardi (1994)

(Roughly) there is a largest logically definable subclass of $\mathtt{NP}$ in which Ladner's Theorem *might not* hold,

- it's the fixed finite template CSPs!

(very roughly)

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \backslash \mathtt{P}$ that are not $\mathtt{NP}$-complete.

Feder and Vardi (1994)

(Roughly) there is a largest logically definable subclass of $\mathtt{NP}$ in which Ladner's Theorem *might not* hold,

- it's the fixed finite template CSPs!
- Conjecture: Ladner's Theorem fails for this class.

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \backslash \mathtt{P}$ that are not $\mathtt{NP}$-complete.

Feder and Vardi (1994)

(Roughly) there is a largest logically definable subclass of $\mathtt{NP}$ in which Ladner's Theorem *might not* hold,

- it's the fixed finite template CSPs!
- Conjecture: Ladner's Theorem fails for this class.

Bulatov/Zhuk (joint best paper award, FOCS 2017)

A fixed template CSP is either solvable in $\mathtt{P}$ or is $\mathtt{NP}$-complete.

Ladner's Theorem versus CSPs

Ladner's Theorem

If $\mathtt{P} \neq \mathtt{NP}$ then there are problems in $\mathtt{NP} \backslash \mathtt{P}$ that are not $\mathtt{NP}$-complete.

Feder and Vardi (1994)

(Roughly) there is a largest logically definable subclass of $\mathtt{NP}$ in which Ladner's Theorem *might not* hold,

- ▷ it's the fixed finite template CSPs!
- ▷ Conjecture: Ladner's Theorem fails for this class.

Bulatov/Zhuk (joint best paper award, FOCS 2017)

A fixed template CSP is either solvable in $\mathtt{P}$ or is $\mathtt{NP}$-complete.

▷ *they give a structural characterisation of hardness for an enormous class of natural problems of interest...*

Goal

Goal

Goal

- Give some sort of appreciation to the background mathematics underlying the Bulatov/Zhuk result and proof

Goal

- Give some sort of appreciation to the background mathematics underlying the Bulatov/Zhuk result and proof
- as well as how this approach and result can be used to achieve other complexity-theoretic classifications

The CSP Dichotomy Theorem

Bulatov/Zhuk 2017

A fixed template CSP is solvable in P if it has a “cyclic polymorphism” and is NP-complete otherwise.

The CSP Dichotomy Theorem

Bulatov/Zhuk 2017

A fixed template CSP is solvable in $\mathtt{P}$ if it has a “cyclic polymorphism” and is $\mathtt{NP}$-complete otherwise.

- to be explained in due course

The CSP Dichotomy Theorem

Bulatov/Zhuk 2017

A fixed template CSP is solvable in P if it has a “cyclic polymorphism” and is NP-complete otherwise.

The CSP Dichotomy Theorem

Bulatov/Zhuk 2017

A fixed template CSP is solvable in P if it has a “cyclic polymorphism” and is NP-complete otherwise.

- the “easy” part

The CSP Dichotomy Theorem

Bulatov/Zhuk 2017

A fixed template CSP is solvable in $\mathtt{P}$ if it has a “cyclic polymorphism” and is $\mathtt{NP}$-complete otherwise.

- the hard part

The CSP Dichotomy Theorem

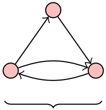
Bulatov/Zhuk 2017

A fixed template CSP is solvable in $\mathtt{P}$ if it has a "cyclic polymorphism" and is $\mathtt{NP}$-complete otherwise.

- the hard part

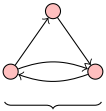
it's really hard

Polymorphism



template $\mathbb{D}$

Polymorphism

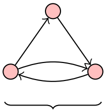


template $\mathbb{D}$

Automorphism

Automorphism: $f : \mathbb{D} \to \mathbb{D}$

Polymorphism

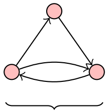

template $\mathbb{D}$

Automorphism

Automorphism: $f : \mathbb{D} \to \mathbb{D}$

- the set of automorphisms form a group action on D

Polymorphism

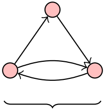


template $\mathbb{D}$

Polymorphism

Polymorphism: $f : \mathbb{D}^n \to \mathbb{D}$

Polymorphism



template $\mathbb{D}$

Polymorphism

Polymorphism: $f : \mathbb{D}^n \to \mathbb{D}$

- the set of all polymorphisms forms an exotic algebra on D

Cyclic

Bulatov/Zhuk 2017

A fixed template CSP is solvable in $\mathtt{P}$ if it has a cyclic polymorphism and is $\mathtt{NP}$-complete otherwise.

Cyclic

Bulatov/Zhuk 2017

A fixed template CSP is solvable in P if it has a cyclic polymorphism and is NP-complete otherwise.

n-ary cyclic polymorphism

$$\forall x_1 \ldots \forall x_n \qquad c(x_1, x_2, \ldots, x_n) = c(x_2, \ldots, x_n, x_1)$$

Cyclic

Bulatov/Zhuk 2017

A fixed template CSP is solvable in $\mathtt{P}$ if it has a cyclic polymorphism and is $\mathtt{NP}$-complete otherwise.

n-ary cyclic polymorphism

$$\forall x_1 \ldots \forall x_n \qquad c(x_1, x_2, \ldots, x_n) = c(x_2, \ldots, x_n, x_1)$$

Behind the scenes (Barto, Kozik 2012 after Taylor 1977)

A finite algebra has a cyclic term if and only if its variety contains no algebras with essentially trivial term operations (projections)

Cyclic

Bulatov/Zhuk 2017

A fixed template CSP is solvable in P if it has a cyclic polymorphism and is NP-complete otherwise.

n-ary cyclic polymorphism

$$\forall x_1 \ldots \forall x_n \qquad c(x_1, x_2, \ldots, x_n) = c(x_2, \ldots, x_n, x_1)$$

Behind the scenes (Barto, Kozik 2012 after Taylor 1977)

A finite algebra has a cyclic term if and only if its variety contains no algebras with essentially trivial term operations (projections)

- iff (roughly) the template relations can logically define the 3SAT ternary relations by way of primitive positive formulæ

Cyclic

Bulatov/Zhuk 2017

A fixed template CSP is solvable in $\mathtt{P}$ if it has a cyclic polymorphism and is $\mathtt{NP}$-complete otherwise.

n-ary cyclic polymorphism

$$\forall x_1 \ldots \forall x_n \qquad c(x_1, x_2, \ldots, x_n) = c(x_2, \ldots, x_n, x_1)$$

Behind the scenes (Barto, Kozik 2012 after Taylor 1977)

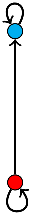
A finite algebra has a cyclic term if and only if its variety contains no algebras with essentially trivial term operations (projections)

- iff it has cyclic terms of all prime arities greater than the size of the algebra

Example: Directed graph unreachability

$$\mathbb{R} = \langle \{0, 1\}; \leq, \textcolor{red}{\blacksquare}, \textcolor{cyan}{\blacksquare} \rangle$$

Example: Directed graph unreachability



$\mathbb{R} = \langle \{0, 1\}; \leq, \blacksquare, \blacksquare \rangle$

Cyclic polymorphism

The operations of meet $\wedge$ and join $\vee$ on $0, 1$ are cyclic polymorphisms.

Example: Directed graph unreachability

$\mathbb{R} = \langle \{0, 1\}; \leq, \blacksquare, \blacksquare \rangle$

Cyclic polymorphism

The operations of meet $\wedge$ and join $\vee$ on $0, 1$ are cyclic polymorphisms.

if...

$$\begin{array}{ccc} a & & b \\ \rotatebox{90}{$\leq$} & \text{and} & \rotatebox{90}{$\leq$} \\ a' & & b' \end{array}$$

Example: Directed graph unreachability

$\mathbb{R} = \langle \{0, 1\}; \leq, \text{■}, \text{■} \rangle$

Cyclic polymorphism

The operations of meet $\wedge$ and join $\vee$ on $0, 1$ are cyclic polymorphisms.

and

$$\begin{array}{ccccc} a & \wedge & b & = & c \\ \rotatebox{90}{$\leq$} & & \rotatebox{90}{$\leq$} & & \\ a' & \wedge & b' & = & c' \end{array}$$

Example: Directed graph unreachability

$\mathbb{R} = \langle\{0, 1\}; \leq, \blacksquare, \blacksquare\rangle$

Cyclic polymorphism

The operations of meet $\wedge$ and join $\vee$ on $0, 1$ are cyclic polymorphisms.

then...

$$\begin{array}{ccccc} a & \wedge & b & = & c \\ \rotatebox{90}{$\leq$} & & \rotatebox{90}{$\leq$} & & \rotatebox{90}{$\leq$} \\ a' & \wedge & b' & = & c' \end{array}$$

Example: Directed graph unreachability

$\mathbb{R} = \langle \{0, 1\}; \leq,$ ■, ■ $\rangle$

Cyclic polymorphism

The operations of meet $\wedge$ and join $\vee$ on $0, 1$ are cyclic polymorphisms.

and obviously $x_1 \wedge x_2 = x_2 \wedge x_1$ (for all $x_1, x_2 \in \{0, 1\}$)

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is NP-complete otherwise

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is NP-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is NP-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is NP-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)
- now assume there is an odd circuit. $\overbrace{u_1 - u_2 - u_3 - \cdots - u_p - u_1}^{\text{an odd length circuit}}$

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is $\mathtt{NP}$-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)
- now assume there is an odd circuit. $\overbrace{u_1 - u_2 - u_3 - \cdots - u_p - u_1}^{\text{an odd length circuit}}$
  Claim: if there is a cyclic polymorphism c, there is a loop

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is $\mathtt{NP}$-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)
- now assume there is an odd circuit. $\overbrace{u_1 - u_2 - u_3 - \cdots - u_p - u_1}^{\text{an odd length circuit}}$
  Claim: if there is a cyclic polymorphism c, there is a loop

$$c(\quad u_1, \quad u_2, \quad \ldots \quad u_{p-1}, \quad u_p)$$

$$c(\quad u_2, \quad u_3, \quad \ldots \quad u_p, \quad u_1)$$

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is NP-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)
- now assume there is an odd circuit. $\overbrace{u_1 - u_2 - u_3 - \cdots - u_p - u_1}^{\text{an odd length circuit}}$
  Claim: if there is a cyclic polymorphism c, there is a loop

$$c(\ u_1,\ u_2,\ \ldots\ u_{p-1},\ u_p) = v$$

$$c(\ u_2,\ u_3,\ \ldots\ u_p,\ u_1) = v$$

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is NP-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)
- now assume there is an odd circuit. $\overbrace{u_1 - u_2 - u_3 - \cdots - u_p - u_1}^{\text{an odd length circuit}}$
  Claim: if there is a cyclic polymorphism c, there is a loop

$$
\begin{array}{lccccccc}
c(& u_1, & u_2, & \dots & u_{p-1}, & u_p) & = & v \\
 & | & | & \dots & | & | & & \\
c(& u_2, & u_3, & \dots & u_p, & u_1) & = & v
\end{array}
$$

Hell-Nešetřil Dichotomy

Hell and Nešetřil (1990)

A finite graph has tractable CSP if it has a loop or is bipartite and is $\mathtt{NP}$-complete otherwise

Modern proof by cyclic polymorphism

- loop: trivial
- bipartite: easy (logspace)
- now assume there is an odd circuit. $\overbrace{u_1 - u_2 - u_3 - \cdots - u_p - u_1}^{\text{an odd length circuit}}$
  Claim: if there is a cyclic polymorphism c, there is a loop

$$\begin{array}{cccccccc} c(& u_1, & u_2, & \ldots & u_{p-1}, & u_p) & = & v \\ & | & | & \ldots & | & | & & | \\ c(& u_2, & u_3, & \ldots & u_p, & u_1) & = & v \end{array}$$

Beyond satisfaction

Many other results follow from polymorphism analysis

- Counting CSPs (complexity of counting solutions)

Beyond satisfaction

Many other results follow from polymorphism analysis

- Counting CSPs (complexity of counting solutions)
- valued CSPs (constraints have a cost. Minimise it.)

Beyond satisfaction

Many other results follow from polymorphism analysis

- Counting CSPs (complexity of counting solutions)
- valued CSPs (constraints have a cost. Minimise it.)
- approximation of CSPs (complexity of solving asymptotically most constraints)

Beyond satisfaction

Many other results follow from polymorphism analysis

- Counting CSPs (complexity of counting solutions)
- valued CSPs (constraints have a cost. Minimise it.)
- approximation of CSPs (complexity of solving asymptotically most constraints)
- **Universal Horn class membership...**

CSPs as homomorphism problems

Fixed template $\mathbb{A}$

CSP($\mathbb{A}$) is just the homomorphism problem for homomorphisms into $\mathbb{A}$

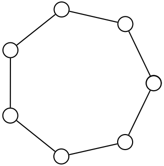
$$\overbrace{\mathbb{B}}^{\text{instance}} \xrightarrow{?} \overbrace{\mathbb{A}}^{\text{template}}$$

CSPs as homomorphism problems

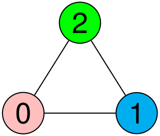
Fixed template $\mathbb{A}$

CSP($\mathbb{A}$) is just the homomorphism problem for homomorphisms into $\mathbb{A}$

$$\overbrace{\mathbb{B}}^{\text{instance}} \xrightarrow{?} \overbrace{\mathbb{A}}^{\text{template}}$$



a graph



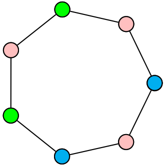
$\mathbb{K}_3$

CSPs as homomorphism problems

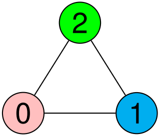
Fixed template $\mathbb{A}$

CSP($\mathbb{A}$) is just the homomorphism problem for homomorphisms into $\mathbb{A}$

$$\overbrace{\mathbb{B}}^{\text{instance}} \xrightarrow{?} \overbrace{\mathbb{A}}^{\text{template}}$$



a graph



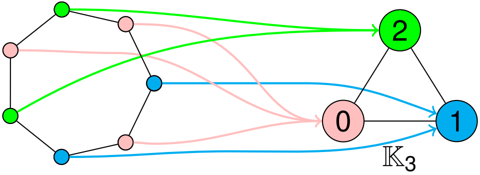
$\mathbb{K}_3$

CSPs as homomorphism problems

Fixed template $\mathbb{A}$

CSP($\mathbb{A}$) is just the homomorphism problem for homomorphisms into $\mathbb{A}$

$$\overbrace{\mathbb{B}}^{\text{instance}} \xrightarrow{?} \overbrace{\mathbb{A}}^{\text{template}}$$



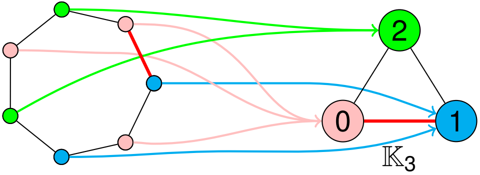
a graph

CSPs as homomorphism problems

Fixed template $\mathbb{A}$

CSP($\mathbb{A}$) is just the homomorphism problem for homomorphisms into $\mathbb{A}$

$$\overbrace{\mathbb{B}}^{\text{instance}} \xrightarrow{?} \overbrace{\mathbb{A}}^{\text{template}}$$



a graph

(3-colouring is an edge-preserving function into $\mathbb{K}_3$)

Universal Horn

A non-constraint is *implied* if every solution maps it inside the target relation.

CSP to universal Horn

- CSP($\mathbb{A}$): is there a homomorphism from $\mathbb{B}$ into $\mathbb{A}$?
- UHorn($\mathbb{A}$): are there no implied constraints?

Universal Horn

A non-constraint is *implied* if every solution maps it inside the target relation.

CSP to universal Horn

- CSP($\mathbb{A}$): is there a homomorphism from $\mathbb{B}$ into $\mathbb{A}$?
- UHorn($\mathbb{A}$): are there no implied constraints?

Fix a finite structure $\mathbb{A}$ in signature $\mathcal{R}$

- $\mathbb{B}$ is an induced substructure of a direct power of $\mathbb{A}$

Universal Horn

A non-constraint is *implied* if every solution maps it inside the target relation.

CSP to universal Horn

- ▷ CSP($\mathbb{A}$): is there a homomorphism from $\mathbb{B}$ into $\mathbb{A}$?
- ▷ UHorn($\mathbb{A}$): are there no implied constraints?

Fix a finite structure $\mathbb{A}$ in signature $\mathcal{R}$

- $\mathbb{B}$ is an induced substructure of a direct power of $\mathbb{A}$
- $\mathbb{B}$ satisfies the universal Horn sentences of $\mathbb{A}$

Universal Horn

A non-constraint is *implied* if every solution maps it inside the target relation.

CSP to universal Horn

- CSP($\mathbb{A}$): is there a homomorphism from $\mathbb{B}$ into $\mathbb{A}$?
- UHorn($\mathbb{A}$): are there no implied constraints?

Fix a finite structure $\mathbb{A}$ in signature $\mathcal{R}$

- $\mathbb{B}$ is an induced substructure of a direct power of $\mathbb{A}$
- $\mathbb{B}$ satisfies the universal Horn sentences of $\mathbb{A}$
- if $r \in \mathcal{R}$ is a relation of arity n and $(b_1, \ldots, b_n) \notin r^B$ then there is a homomorphism ϕ from $\mathbb{B}$ to $\mathbb{A}$ with $(\phi(b_1), \ldots, \phi(b_n)) \notin r^A$

Universal Horn

A non-constraint is *implied* if every solution maps it inside the target relation.

CSP to universal Horn

- CSP($\mathbb{A}$): is there a homomorphism from $\mathbb{B}$ into $\mathbb{A}$?
- UHorn($\mathbb{A}$): are there no implied constraints?

Fix a finite structure $\mathbb{A}$ in signature $\mathcal{R}$

- $\mathbb{B}$ is an induced substructure of a direct power of $\mathbb{A}$
- $\mathbb{B}$ satisfies the universal Horn sentences of $\mathbb{A}$
- if $r \in \mathcal{R}$ is a relation of arity n and $(b_1, \ldots, b_n) \notin r^B$ then there is a homomorphism ϕ from $\mathbb{B}$ to $\mathbb{A}$ with $(\phi(b_1), \ldots, \phi(b_n)) \notin r^A$
- $\mathbb{B}$ has no implied constraints relative to $\mathbb{A}$

CSP and UHorn are different

CSP and UHorn are different

Example

$$\mathbb{R} = \langle \{0, 1\}; \leq, \blacksquare, \blacksquare \rangle$$

CSP and UHorn are different

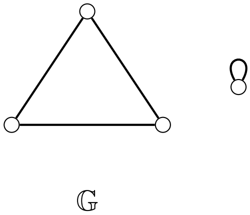
Example

CSP($\mathbb{R}$) is $\mathtt{NL}$-complete, but UHorn($\mathbb{R}$) is first order definable

$\mathbb{R} = \langle \{0, 1\}; \leq, \blacksquare, \blacksquare \rangle$

CSP and UHorn are different

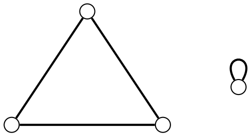
Example



CSP and UHorn are different

Example

CSP($\mathbb{G}$) is first order definable, but UHorn($\mathbb{G}$) is $\mathtt{NP}$-complete



$\mathbb{G}$

Main Result

Barto, Jackson, Ham (2017)

UHorn($\mathbb{A}$) is solvable in $\mathtt{P}$ if $\mathbb{A}$ has an idempotent cyclic polymorphism and otherwise is $\mathtt{NP}$-complete

Main Result

Barto, Jackson, Ham (2017)

UHorn($\mathbb{A}$) is solvable in $\mathtt{P}$ if $\mathbb{A}$ has an idempotent cyclic polymorphism and otherwise is $\mathtt{NP}$-complete


(roughly)

The easy part is the easiness part

Let $\mathbb{A}_{\mathrm{const}}$ be the result of adding singleton unary relations to $\mathbb{A}$

The easy part is the easiness part

Let $\mathbb{A}_{\text{const}}$ be the result of adding singleton unary relations to $\mathbb{A}$

If $\mathbb{A}$ has an idempotent cyclic polymorphism

The easy part is the easiness part

Let $\mathbb{A}_{\mathrm{const}}$ be the result of adding singleton unary relations to $\mathbb{A}$

If $\mathbb{A}$ has an idempotent cyclic polymorphism

1. then by the Bulatov/Zhuk Dichotomy Theorem, CSP($\mathbb{A}_{\mathrm{const}}$) is tractable (the very hard part of their result)

The easy part is the easiness part

Let $\mathbb{A}_{\text{const}}$ be the result of adding singleton unary relations to $\mathbb{A}$

If $\mathbb{A}$ has an idempotent cyclic polymorphism

1. then by the Bulatov/Zhuk Dichotomy Theorem, CSP($\mathbb{A}_{\text{const}}$) is tractable (the very hard part of their result)
2. make multiple calls to this to solve membership in UHorn($\mathbb{A}$)

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

1. Then $\mathbb{A}_{\text{const}}$ has no cyclic polymorphism

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

1. Then $\mathbb{A}_{\mathrm{const}}$ has no cyclic polymorphism
2. Apply the ANT to $\mathbb{A}_{\mathrm{const}}$

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

1. Then $\mathbb{A}_{\mathrm{const}}$ has no cyclic polymorphism
2. Apply the ANT to $\mathbb{A}_{\mathrm{const}}$

The All or Nothing Theorem (ANT); Ham, J, 2016

(Nothing is easy part)

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

1. Then $\mathbb{A}_{\text{const}}$ has no cyclic polymorphism
2. Apply the ANT to $\mathbb{A}_{\text{const}}$

The All or Nothing Theorem (ANT); Ham, J, 2016

(Nothing is easy part)

- if $\mathbb{D}$ has no cyclic polymorphism then $\forall k\ \exists \ell$ s.t. it is $\mathtt{NP}$-hard to distinguish (i) from (ii):
  - (i) $\mathbb{B}$ is every reasonable partial homomorphism on k elements extends to a solution
  - (ii) $\mathbb{B}$ has no homomorphism into $\mathbb{A}$

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

1. Then $\mathbb{A}_{\mathrm{const}}$ has no cyclic polymorphism
2. Apply the ANT to $\mathbb{A}_{\mathrm{const}}$

The All or Nothing Theorem (ANT); Ham, J, 2016

(Nothing is easy part)

- if $\mathbb{D}$ has no cyclic polymorphism then $\forall k\ \exists \ell$ s.t. it is $\mathtt{NP}$-hard to distinguish (i) from (ii):
  - (i) $\mathbb{B}$ is every reasonable partial homomorphism on k elements extends to a solution
  - (ii) $\mathbb{B}$ has no homomorphism into $\mathbb{A}$

reasonable: can be extended to any further ℓ elements

The hard part is the hardness part

If $\mathbb{A}$ has no idempotent cyclic polymorphism

1. Then $\mathbb{A}_{\text{const}}$ has no cyclic polymorphism
2. Apply the ANT to $\mathbb{A}_{\text{const}}$

The All or Nothing Theorem (ANT); Ham, J, 2016

(Nothing is easy part)

- if $\mathbb{D}$ has no cyclic polymorphism then $\forall k\ \exists \ell$ s.t. it is $\mathtt{NP}$-hard to distinguish (i) from (ii):
  - (i) $\mathbb{B}$ is every reasonable partial homomorphism on k elements extends to a solution
  - (ii) $\mathbb{B}$ has no homomorphism into $\mathbb{A}$

reasonable: can be extended to any further ℓ elements

- (ii) means No for UHorn($\mathbb{A}$). Argue (nontrivially) how (i) implies YES for UHorn($\mathbb{A}$).

References

- L. Barto, L. Ham and M. Jackson, Flexible satisfaction, in progress. arXiv1611.00886
- L. Barto and M. Kozik, Absorbing subalgebras, cyclic terms and the constraint satisfaction problem, Logical Methods in Computer Science, 8/1:07 (2012), 1–26.
- A. Bulatov, A dichotomy theorem for nonuniform CSPs, FOCS 2017, pp 319–330. arXiv:1703.03021
- T. Feder and M. Vardi, The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory, SIAM J. Computing, 28 (1), 1998, 57–104.
- L. Ham and M. Jackson, All or Nothing: toward a promise problem dichotomy for constraint satisfaction problems, in Principles and practice of Constraint Programming, J.C. Beck (Ed.): CP 2017, LNCS 10416, pp. 139–156, 2017.
- P. Hell and J. Nešetřil, On the complexity of H-coloring, J. Combin. Theory Ser. B 48(1) (1990), 92–110.
- T. J. Schaefer, The Complexity of Satisfiability Problems, STOC (1978), pp. 216–226.
- W. Taylor, Varieties obeying homotopy laws, Can J. Math. 29 (1977), 498–527.
- D. Zhuk, A Proof of CSP Dichotomy Conjecture, FOCS 2017, pp. 331–342. arXiv:704.01914