The proof of CSP Dichotomy Conjecture for 5-element domain

Dmitriy Zhuk
zhuk.dmitriy@gmail.com

Department of Mathematics and Mechanics
Moscow State University

Arbeitstagung Allgemeine Algebra
91th Workshop on General Algebra
Brno, February 5-7, 2016
Outline

1. What is CSP?
2. Transitive Closure
3. 1-Consistency
4. Absorbtion
5. Rosenberg Completeness Theorem
6. Central Relations
7. Partial Order Relations
8. All Functions
9. Linear Case
10. Algorithm
Definitions

Let $A$ be a finite set.
A mapping $A^n \rightarrow \{0, 1\}$ is called an $n$-ary predicate.
A subset $\rho \subseteq A^n$ is called an $n$-ary relation.

- We do not distinguish between predicates and relations.
Constraint Satisfaction Problem

Let $G$ be a finite set of predicates.

**CSP($G$)**

**Given:** a conjunction of predicates, i.e. a formula

$$\rho_1(x_{i_1,1}, \ldots, x_{i_1,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}),$$

where $\rho_1, \ldots, \rho_s \in G$.

**Decide:** whether the formula is satisfiable.
Constraint Satisfaction Problem

Let $G$ be a finite set of predicates.

**CSP($G$)**

Given: a conjunction of predicates, i.e. a formula

$$\rho_1(x_{i_1,1}, \ldots, x_{i_1,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}),$$

where $\rho_1, \ldots, \rho_s \in G$.

Decide: whether the formula is satisfiable.

**Example**

$A = \{0, 1, 2\}, G = \{x < y, x \leq y\}$.

CSP instances:

$x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4,$
Constraint Satisfaction Problem

Let $G$ be a finite set of predicates.

**CSP**($G$)

Given: a conjunction of predicates, i.e. a formula

$$\rho_1(x_{i_1,1}, \ldots, x_{i_1,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}),$$

where $\rho_1, \ldots, \rho_s \in G$.

Decide: whether the formula is satisfiable.

**Example**

$A = \{0, 1, 2\}, G = \{x < y, x \leq y\}$.

CSP instances:

$x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$, **No solutions**
Constraint Satisfaction Problem

Let $G$ be a finite set of predicates.

### CSP($G$)

Given: a conjunction of predicates, i.e. a formula

$$
\rho_1(x_{i_1,1}, \ldots, x_{i_1,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}),
$$

where $\rho_1, \ldots, \rho_s \in G$.

Decide: whether the formula is satisfiable.

### Example

$A = \{0, 1, 2\}, \ G = \{x < y, x \leq y\}$.

CSP instances:

- $x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$, No solutions
- $x_1 \leq x_2 \land x_2 \leq x_3 \land x_3 \leq x_1$,
Constraint Satisfaction Problem

Let $G$ be a finite set of predicates.

**CSP($G$)**

Given: a conjunction of predicates, i.e. a formula

$$\rho_1(x_{i_1,1}, \ldots, x_{i_1,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}),$$

where $\rho_1, \ldots, \rho_s \in G$.

Decide: whether the formula is satisfiable.

**Example**

$A = \{0, 1, 2\}$, $G = \{x < y, x \leq y\}$.

CSP instances:

$x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$, No solutions

$x_1 \leq x_2 \land x_2 \leq x_3 \land x_3 \leq x_1$, $x_1 = x_2 = x_3 = 0$. 
A weak near unanimity operation (WNU) is an operation $f$ satisfying
$f(x, x, \ldots, x) = x$ and
\[ f(x, \ldots, x, y) = f(x, \ldots, x, y, x) = \cdots = f(y, x, \ldots, x). \]
A weak near unanimity operation (WNU) is an operation $f$ satisfying:

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

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

Suppose $(x = c)$ belongs to $G$ for every $c \in A$. 

Conjecture CSP($G$) is solvable in polynomial time if there exists a WNU preserving $G$. CSP($G$) is NP-complete otherwise.

Theorem [Ralph McKenzie and Miklós Maróti]

CSP($G$) is NP-complete if no WNU preserving $G$. 

Challenge Given a finite set of predicates $G$ and a WNU $w$ that preserves $G$. Find an algorithm that solves CSP($G$) in polynomial time.

Dmitriy Zhuk zhuk.dmitriy@gmail.com (Moscow State University)
A weak near unanimity operation (WNU) is an operation $f$ satisfying $f(x, x, \ldots, x) = x$ and $f(x, \ldots, x, y, x) = \cdots = f(y, x, \ldots, x)$.

Suppose $(x = c)$ belongs to $G$ for every $c \in A$.

Conjecture

CSP($G$) is solvable in polynomial time if there exists a WNU preserving $G$, CSP($G$) is NP-complete otherwise.
A weak near unanimity operation (WNU) is an operation $f$ satisfying
\[ f(x, x, \ldots, x) = x \text{ and } f(x, \ldots, x, y) = f(x, \ldots, x, y, x) = \cdots = f(y, x, \ldots, x). \]

Suppose $(x = c)$ belongs to $G$ for every $c \in A$.

**Conjecture**

$\text{CSP}(G)$ is solvable in polynomial time if there exists a WNU preserving $G$, $\text{CSP}(G)$ is NP-complete otherwise.

**Theorem** [Ralph McKenzie and Miklós Maróti]

$\text{CSP}(G)$ is NP-complete if no WNU preserving $G$. 
A weak near unanimity operation (WNU) is an operation $f$ satisfying
$f(x, x, \ldots, x) = x$ and
$f(x, \ldots, x, y) = f(x, \ldots, x, y, x) = \cdots = f(y, x, \ldots, x)$.

Suppose $(x = c)$ belongs to $G$ for every $c \in A$.

Conjecture

CSP($G$) is solvable in polynomial time if there exists a WNU preserving $G$, CSP($G$) is NP-complete otherwise.

Theorem [Ralph McKenzie and Miklós Maróti]

CSP($G$) is NP-complete if no WNU preserving $G$.

Challenge

Given a finite set of predicates $G$ and a WNU $w$ that preserves $G$. Find an algorithm that solves CSP($G$) in polynomial time.
Given a CSP instance

\[ \rho_1(x_{i,1}, \ldots, x_{i,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}), \]

where \( \rho_1, \ldots, \rho_s \in G. \)
Given a CSP instance

\[ \rho_1(x_{i_1,1}, \ldots, x_{i_1,n_1}) \land \cdots \land \rho_s(x_{i_s,1}, \ldots, x_{i_s,n_s}), \]

where \( \rho_1, \ldots, \rho_s \in G. \)

**Step 1: Generate all binary constraints**

For every constraint \( \rho(x_1, \ldots, x_n) \) and \( i, j \in \{1, 2, \ldots, n\} \) we add a binary constraint \( \sigma_{i,j}(x_i, x_j) \), where

\[ \sigma_{i,j}(y_i, y_j) = \exists y_1 \ldots \exists y_{i-1} \exists y_{i+1} \ldots \exists y_{j-1} \exists y_{j+1} \ldots \exists y_n \rho(y_1, \ldots, y_n). \]
Step 2: Transitive closure.

For every 2 binary constraints $\rho_1(x_i, x_j)$ and $\rho_2(x_j, x_k)$ we add the constraint $\rho_3(x_i, x_k)$, where $\rho_3(y_1, y_2) = \exists z \rho_1(y_1, z) \land \rho_2(z, y_2)$. 
Transitive Closure

Step 2: Transitive closure.

For every 2 binary constraints $\rho_1(x_i, x_j)$ and $\rho_2(x_j, x_k)$ we add the constraint $\rho_3(x_i, x_k)$, where $\rho_3(y_1, y_2) = \exists z \rho_1(y_1, z) \land \rho_2(z, y_2)$.

Example

We have constraints $(x_1 \leq x_2)$ and $(x_2 \leq x_3)$. We add $(x_1 \leq x_3)$.
1-Consistency

Let $D_i$ be the domain of $x_i$. A CSP instance is called **1-consistent** if $x_i$ in any constraint takes all values from $D_i$. 

Step 3: Constraint propagation.

We can provide 1-consistency: if a variable $x_i$ takes only values from $D'_i \subset D_i$ in a constraint then we reduce the domain of $x_i$ to $D'_i$ and restrict all other constraints.

Example CSP instance on $A = \{0, 1, 2\}$, $x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$.

$x_1 < x_2 \Rightarrow$ the domain of $x_2$ can be reduced to $\{1, 2\}$,

$x_2 < x_3 \Rightarrow$ the domain of $x_3$ can be reduced to $\{2\}$,

$x_3 < x_4 \Rightarrow$ no solution for $x_4$. We get a contradiction.

We cannot reduce forever, hence either we get 1-consistency, or we get a contradiction.
1-Consistency

Let $D_i$ be the domain of $x_i$. A CSP instance is called 1-consistent if $x_i$ in any constraint takes all values from $D_i$.

**Step 3: Constraint propagation.**

We can provide 1-consistency:

if a variable $x_i$ takes only values from $D'_i \subset D_i$ in a constraint then we reduce the domain of $x_i$ to $D'_i$ and restrict all other constraints.
1-Consistency

Let $D_i$ be the domain of $x_i$. A CSP instance is called 1-consistent if $x_i$ in any constraint takes all values from $D_i$.

**Step 3: Constraint propagation.**

We can provide 1-consistency:

if a variable $x_i$ takes only values from $D'_i \subseteq D_i$ in a constraint then we reduce the domain of $x_i$ to $D'_i$ and restrict all other constraints.

**Example**

CSP instance on $A = \{0, 1, 2\}$, $x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$. 

Dmitriy Zhuk zhuk.dmitriy@gmail.com (Moscow State University)

CSP for small domain

AAA91 8 / 19
1-Consistency

Let $D_i$ be the domain of $x_i$. A CSP instance is called 1-consistent if $x_i$ in any constraint takes all values from $D_i$.

**Step 3: Constraint propagation.**

We can provide 1-consistency:
if a variable $x_i$ takes only values from $D'_i \subset D_i$ in a constraint then we reduce the domain of $x_i$ to $D'_i$ and restrict all other constraints.

**Example**

CSP instance on $A = \{0, 1, 2\}$, $x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$.
$x_1 < x_2 \Rightarrow$ the domain of $x_2$ can be reduced to $\{1, 2\}$.
1-Consistency

Let $D_i$ be the domain of $x_i$. A CSP instance is called 1-consistent if $x_i$ in any constraint takes all values from $D_i$.

Step 3: Constraint propagation.

We can provide 1-consistency:
if a variable $x_i$ takes only values from $D'_i \subset D_i$ in a constraint then we reduce the domain of $x_i$ to $D'_i$ and restrict all other constraints.

Example

CSP instance on $A = \{0, 1, 2\}$, $x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$.

$x_1 < x_2 \Rightarrow$ the domain of $x_2$ can be reduced to $\{1, 2\}$,

$x_2 < x_3 \Rightarrow$ the domain of $x_3$ can be reduced to $\{2\}$,
1-Consistency

Let $D_i$ be the domain of $x_i$. A CSP instance is called 1-consistent if $x_i$ in any constraint takes all values from $D_i$.

**Step 3: Constraint propagation.**

We can provide 1-consistency:
if a variable $x_i$ takes only values from $D'_i \subset D_i$ in a constraint then we reduce the domain of $x_i$ to $D'_i$ and restrict all other constraints.

**Example**

CSP instance on $A = \{0, 1, 2\}$, $x_1 < x_2 \land x_2 < x_3 \land x_3 < x_4$.
$x_1 < x_2 \Rightarrow$ the domain of $x_2$ can be reduced to $\{1, 2\}$,
$x_2 < x_3 \Rightarrow$ the domain of $x_3$ can be reduced to $\{2\}$,
$x_3 < x_4 \Rightarrow$ no solution for $x_4$. We get a contradiction.

- We cannot reduce forever, hence either we get 1-consistency, or we get a contradiction.
Libor Barto said something about absorption...said it is very important...
Libor Barto said something about absorption...said it is very important... But it was complicated... I consider only binary absorption!
Libor Barto said something about absorption... said it is very important... But it was complicated... I consider only binary absorption!

**Definition**

A subuniverse $B$ absorbs $A$ if there exists a binary operation $f \in \text{Clo}(w)$ such that $f(B, A) \subseteq B$ and $f(A, B) \subseteq B$.

- $\text{Clo}(w)$ is the clone generated by a WNU $w$. 
Libor Barto said something about absorption... said it is very important... But it was complicated... I consider only binary absorption!

**Definition**

A subuniverse $B$ absorbs $A$ if there exists a binary operation $f \in \text{Clo}(w)$ such that $f(B, A) \subseteq B$ and $f(A, B) \subseteq B$.

- $\text{Clo}(w)$ is the clone generated by a WNU $w$.

**Step 4: Absorbing restriction.**

If $B_i$ absorbs $D_i$, we reduce the domain $D_i$ to $B_i$. Then we go to Step 3 and provide 1-consistency!

- Constraint propagation cannot give a contradiction in this case!
That was all I knew about CSP...
But I know a bit about Clone Theory...Let try to apply it!
That was all I knew about CSP...
But I know a bit about Clone Theory...Let try to apply it!

Main Results in Clone Theory

1. The description of all clones on 2 elements (Post's Lattice).
2. Rosenb erg's description of all maximal clones on $k$ elements
   Rosenb erg Completeness Theorem
   There are only following maximal clones on $k$ elements.
   1. Maximal clone of monotone functions;
   2. Maximal clone of auto dual functions;
   3. Maximal clone dened by an equivalence relation;
   4. Maximal clone of quasi-linear functions;
   5. Maximal clone dened by a unary relation;
   6. Maximal clone dened by a central relation;
   7. Maximal clone dened by an $h$-universal relation.
That was all I knew about CSP... But I know a bit about Clone Theory...Let try to apply it!

Main Results in Clone Theory

1. The description of all clones on 2 elements (Post’s Lattice).
That was all I knew about CSP... But I know a bit about Clone Theory... Let try to apply it!

Main Results in Clone Theory

1. The description of all clones on 2 elements (Post’s Lattice).
That was all I knew about CSP...
But I know a bit about Clone Theory...Let try to apply it!

### Main Results in Clone Theory

1. The description of all clones on 2 elements (Post’s Lattice).
2. Rosenberg’s description of all maximal clones on $k$ elements.
That was all I knew about CSP...
But I know a bit about Clone Theory...Let try to apply it!

### Main Results in Clone Theory

1. The description of all clones on 2 elements (Post’s Lattice).
2. Rosenberg’s description of all maximal clones on $k$ elements.

### Rosenberg Completeness Theorem

There are only following maximal clones on $k$ elements.

1. Maximal clone defined by a unary relation;
2. Maximal clone of monotone functions;
3. Maximal clone of autodual functions;
4. Maximal clone defined by an equivalence relation;
5. Maximal clone of quasi-linear functions;
6. Maximal clone defined by a central relation;
7. Maximal clone defined by an $h$-universal relation.
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$. 

1. $C$ is the clone of all functions on $D_i$, or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants.

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU.

4. Maximal clone of autodual functions; cannot happen because of constants.

5. Maximal clone defined by an equivalence relation; factorize WNU, generate a new clone, apply Rosenb erg Theorem again.

6. Maximal clone defined by a central relation.

7. Maximal clone of monotone functions.

Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$,

or $C$ belongs to one of the maximal clones.
Let $\mathcal{C}$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $\mathcal{C}$ is the clone of all functions on $D_i$,
2. or $\mathcal{C}$ belongs to one of the maximal clones.
3. Maximal clone defined by a unary relation;
4. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU
5. Maximal clone of autodual functions; cannot happen because of constants
6. Maximal clone defined by an equivalence relation; factorize WNU, generate a new clone, apply Rosenberg Theorem again
7. Maximal clone defined by a central relation;
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$, or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$,

or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an $h$-universal relation;
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$, or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants.

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU.
Let $\mathcal{C}$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $\mathcal{C}$ is the clone of all functions on $D_i$,

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions;
Let \( C \) be the clone generated by the WNU \( w \) on \( D_i \) and all constants from \( D_i \).

Apply Rosenberg theorem. Then

1. \( C \) is the clone of all functions on \( D_i \),

or \( C \) belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an \( h \)-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions; cannot happen because of constants
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.
Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$,

or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions; cannot happen because of constants

5. Maximal clone defined by an equivalence relation;
Let \( C \) be the clone generated by the WNU \( w \) on \( D_i \) and all constants from \( D_i \).

Apply Rosenberg theorem. Then

1. \( C \) is the clone of all functions on \( D_i \),

or \( C \) belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an \( h \)-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions; cannot happen because of constants

5. Maximal clone defined by an equivalence relation; factorize WNU, generate a new clone, apply Rosenberg Theorem again
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$,

or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions; cannot happen because of constants

5. Maximal clone defined by an equivalence relation; factorize WNU, generate a new clone, apply Rosenberg Theorem again

6. Maximal clone defined by a central relation;
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.
Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$,

or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions; cannot happen because of constants

5. Maximal clone defined by an equivalence relation; factorize WNU, generate a new clone, apply Rosenberg Theorem again

6. Maximal clone defined by a central relation;

7. Maximal clone of monotone functions;
Let $C$ be the clone generated by the WNU $w$ on $D_i$ and all constants from $D_i$.

Apply Rosenberg theorem. Then

1. $C$ is the clone of all functions on $D_i$,

or $C$ belongs to one of the maximal clones.

2. Maximal clone defined by a unary relation; cannot happen because of constants

3. Maximal clone defined by an $h$-universal relation; cannot happen because of WNU

4. Maximal clone of autodual functions; cannot happen because of constants

5. Maximal clone defined by an equivalence relation; factorize WNU, generate a new clone, apply Rosenberg Theorem again

6. Maximal clone defined by a central relation;

7. Maximal clone of monotone functions;

8. Maximal clone of quasi-linear functions;
Central Relations

To simplify we consider only binary central relations.

A relation \( \rho \subseteq A \times A \) is called central if it is reflexive, symmetric, and there exists \( c \) such that \( \{ c \} \times A \subseteq \rho \).

- the set of all elements \( c \) such that \( \{ c \} \times A \subseteq \rho \) is called center.

Step 5: Central restriction. If \( C_i \) is a center in \( D_i \), we reduce \( D_i \) to \( C_i \). Then we go to Step 3 and provide 1-consistency!

If we don't have binary absorption, then constraint propagation cannot give a contradiction in this case!
To simplify we consider only binary central relations.

A relation $\rho \subseteq A \times A$ is called central if it is reflexive, symmetric, and there exists $c$ such that $\{c\} \times A \subseteq \rho$.

- the set of all elements $c$ such that $\{c\} \times A \subseteq \rho$ is called center.

**Step 5: Central restriction.**

If $C_i$ is a center in $D_i$, we reduce $D_i$ to $C_i$.

Then we go to Step 3 and provide 1-consistency!
Central Relations

To simplify we consider only binary central relations.

A relation $\rho \subseteq A \times A$ is called central if it is reflexive, symmetric, and there exists $c$ such that $\{c\} \times A \subseteq \rho$.

- the set of all elements $c$ such that $\{c\} \times A \subseteq \rho$ is called center.

Step 5: Central restriction.

If $C_i$ is a center in $D_i$, we reduce $D_i$ to $C_i$. Then we go to Step 3 and provide 1-consistency!

- if we don’t have binary absorbtion, then constraint propagation cannot give a contradiction in this case!
Every maximal clone of monotone functions is defined by a partial order relation with a greatest and a least element.

- the least element can be viewed as a center.

![Diagram with nodes 0, 1, 2, 3, 4 showing a partial order relation and the least element as a center.]

Dmitriy Zhuk
zhuk.dmitriy@gmail.com
(Moscow State University)
Every maximal clone of monotone functions is defined by a partial order relation with a greatest and a least element.

- the least element can be viewed as a center.

**Step 5: Central restriction.**

If we have a partial order on $D_i$, we reduce $D_i$ to $\{g\}$ where $g$ is the least element.

Then we go to Step 3 and provide 1-consistency!
Every maximal clone of monotone functions is defined by a partial order relation with a greatest and a least element.

- the least element can be viewed as a center.

**Step 5: Central restriction.**

If we have a partial order on $D_i$, we reduce $D_i$ to $\{g\}$ where $g$ is the least element.

Then we go to Step 3 and provide 1-consistency!

- if we don’t have binary absorption, then constraint propagation cannot give a contradiction in this case!
For a congruence $\sigma$ the clone generated by $w/\sigma$ and constants is the clone of all functions.
For a congruence $\sigma$ the clone generated by $w/\sigma$ and constants is the clone of all functions.

Step 6: “All Functions” restriction.
Choose any equivalence class $E$ in $\sigma$ and reduce the domain $D_i$ to $E$. Then we go to Step 3 and provide 1-consistency!
For a congruence $\sigma$ the clone generated by $w/\sigma$ and constants is the clone of all functions.

**Step 6: “All Functions” restriction.**

Choose any equivalence class $E$ in $\sigma$ and reduce the domain $D_i$ to $E$. Then we go to Step 3 and provide 1-consistency!

- if we don’t have a binary absorption and a center, then constraint propagation cannot give a contradiction in this case!
Maximal clone of quasi-linear functions

- If a WNU is a quasi-linear function then it can be represented as
  \( t \cdot (x_1 + x_2 + \ldots + x_n) \) for an integer \( t \) and an operation + from an abelian group.
If a WNU is a quasi-linear function then it can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$ for an integer $t$ and an operation $+$ from an abelian group.

### Step 7: Linear restriction

1. For every $i$ choose the minimal congruence $\sigma_i$ on $D_i$ such that the WNU $w/\sigma_i$ can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$. 
Maximal clone of quasi-linear functions

- If a WNU is a quasi-linear function then it can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$ for an integer $t$ and an operation $+$ from an abelian group.

Step 7: Linear restriction

1. For every $i$ choose the minimal congruence $\sigma_i$ on $D_i$ such that the WNU $w/\sigma_i$ can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$.
2. Factorize all the constraints, i.e. replace every predicate $\rho$ by $\rho'(x_1, \ldots, x_n) = \exists y_1 \ldots \exists y_n \rho(y_1, \ldots, y_n) \land (x_1, y_1) \in \sigma_i \land \ldots \land (x_n, y_n) \in \sigma_n$

The obtained CSP instance we denote by $\Theta$
If a WNU is a quasi-linear function then it can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$ for an integer $t$ and an operation $+$ from an abelian group.

**Step 7: Linear restriction**

1. For every $i$ choose the minimal congruence $\sigma_i$ on $D_i$ such that the WNU $w/\sigma_i$ can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$.

2. Factorize all the constraints, i.e. replace every predicate $\rho$ by $\rho'(x_1, \ldots, x_n) = \exists y_1 \ldots \exists y_n \rho(y_1, \ldots, y_n) \land (x_1, y_1) \in \sigma_i \land \cdots \land (x_n, y_n) \in \sigma_n$.

   The obtained CSP instance we denote by $\Theta$.

3. Solve $\Theta$ using any algorithm for Mal’tsev case.
Maximal clone of quasi-linear functions

- If a WNU is a quasi-linear function then it can be represented as \( t \cdot (x_1 + x_2 + \ldots + x_n) \) for an integer \( t \) and an operation + from an abelian group.

### Step 7: Linear restriction

1. For every \( i \) choose the minimal congruence \( \sigma_i \) on \( D_i \) such that the WNU \( w/\sigma_i \) can be represented as \( t \cdot (x_1 + x_2 + \ldots + x_n) \).

2. Factorize all the constraints, i.e. replace every predicate \( \rho \) by

\[
\rho'(x_1, \ldots, x_n) = \exists y_1 \ldots \exists y_n \rho(y_1, \ldots, y_n) \land (x_1, y_1) \in \sigma_{i_1} \land \cdots \land (x_n, y_n) \in \sigma_{i_n}
\]

The obtained CSP instance we denote by \( \Theta \).

3. Solve \( \Theta \) using any algorithm for Mal’tsev case.

4. If \( \Theta \) has a solution, we reduce every domain \( D_i \) to the equivalence class from the solution. **This restriction is 1-consistent!!!**
Maximal clone of quasi-linear functions

- If a WNU is a quasi-linear function then it can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$ for an integer $t$ and an operation $+$ from an abelian group.

**Step 7: Linear restriction**

1. For every $i$ choose the minimal congruence $\sigma_i$ on $D_i$ such that the WNU $w/\sigma_i$ can be represented as $t \cdot (x_1 + x_2 + \ldots + x_n)$.

2. Factorize all the constraints, i.e. replace every predicate $\rho$ by

   $$\rho'(x_1, \ldots, x_n) = \exists y_1 \ldots \exists y_n \rho(y_1, \ldots, y_n) \land (x_1, y_1) \in \sigma_i \land \cdots \land (x_n, y_n) \in \sigma_i$$

   The obtained CSP instance we denote by $\Theta$

3. Solve $\Theta$ using any algorithm for Mal’tsev case.

4. If $\Theta$ has a solution, we reduce every domain $D_i$ to the equivalence class from the solution. **This restriction is 1-consistent!!!**

5. If $\Theta$ doesn’t have a solution then we find a subset $A'_i$ of the original domain $A_i$ such that no solutions with $x_i \in A'_i \setminus A_i$. 

Dmitriy Zhuk zhuk.dmitriy@gmail.com (Moscow State University)
Algorithm

1. Generate all binary constraints.
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.

If necessary go to Step 1.

If there exists a binary absorption
Apply Absorbing Restriction and go to Step 3.

If there exists a center
Apply Central Restriction and go to Step 3.

If we get all functions after factorization
Apply All Functions restriction and go to Step 3.

If the WNU $w$ is quasi-linear after factorization
Solve the Maltsev CSP.
If there is a solution, apply Linear Restriction and go to Step 4.
otherwise, reduce the original domain $A_i$ to $A'_i$.

Either it gives a solution, or it reduces the original domain $A_i$ to $A'_i$, or it proves that no general solutions.
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.
3. Provide 1-consistency. If necessary go to Step 1.

If there exists a binary absorption
Apply Absorbing Restriction and go to Step 3.

If there exists a center
Apply Central Restriction and go to Step 3.

If we get all functions after factorization
Apply All Functions restriction and go to Step 3.

If the WNU \( w \) is quasi-linear after factorization
Solve the Maltsev CSP.
If there is a solution, apply Linear Restriction and go to Step 4.
otherwise, reduce the original domain \( A_i \) to \( A_i' \).
Either it gives a solution, or it reduces the original domain \( A_i \) to \( A_i' \), or it proves that no general solutions.
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.
3. Provide 1-consistency. If necessary go to Step 1.
4. If there exists a binary absorption
   Apply Absorbing Restriction and go to Step 3.
5. If there exists a center
   Apply Central Restriction and go to Step 3.
6. If we get all functions after factorization
   Apply All Functions restriction and go to Step 3.
7. If the WNU $w$ is quasi-linear after factorization
   Solve the Maltsev CSP.
   If there is a solution, apply Linear Restriction and go to Step 4.
   otherwise, reduce the original domain $A_i$ to $A'_i$.
   Either it gives a solution, or it reduces the original domain $A_i$ to $A'_i$, or it proves that no general solutions.
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.
3. Provide 1-consistency. If necessary go to Step 1.
4. If there exists a binary absorption
   Apply Absorbing Restriction and go to Step 3.
5. If there exists a center
   Apply Central Restriction and go to Step 3.
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.
3. Provide 1-consistency. If necessary go to Step 1.
4. If there exists a binary absorption
   Apply Absorbing Restriction and go to Step 3.
5. If there exists a center
   Apply Central Restriction and go to Step 3.
6. If we get all functions after factorization
   Apply “All Functions” restriction and go to Step 3.
7. If the WNU $w$ is quasi-linear after factorization
   Solve the Maltsev CSP.
   If there is a solution, apply Linear Restriction and go to Step 4.
   otherwise, reduce the original domain $A_i$ to $A_i'$. Either it gives a solution,
   or it reduces the original domain $A_i$ to $A_i'$, or it proves that no general solutions.
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.
3. Provide 1-consistency. If necessary go to Step 1.
4. If there exists a binary absorption
   Apply Absorbing Restriction and go to Step 3.
5. If there exists a center
   Apply Central Restriction and go to Step 3.
6. If we get all functions after factorization
   Apply “All Functions” restriction and go to Step 3.
7. If the WNU $w$ is quasi-linear after factorization
   - Solve the Maltsev CSP.
   - If there is a solution, apply Linear Restriction and go to Step 4.
   - otherwise, reduce the original domain $A_i$ to $A'_i$. 
Algorithm

1. Generate all binary constraints.
2. Transitive Closure.
3. Provide 1-consistency. If necessary go to Step 1.
4. If there exists a binary absorption
   Apply Absorbing Restriction and go to Step 3.
5. If there exists a center
   Apply Central Restriction and go to Step 3.
6. If we get all functions after factorization
   Apply “All Functions” restriction and go to Step 3.
7. If the WNU $w$ is quasi-linear after factorization
   - Solve the Maltsev CSP.
   - If there is a solution, apply Linear Restriction and go to Step 4.
   - Otherwise, reduce the original domain $A_i$ to $A'_i$.

Either it gives a solution,
or it reduces the original domain $A_i$ to $A'_i$,
or it proves that no general solutions.
Does the algorithm work?

I can prove that it works if we don't apply linear restrictions twice.

Why does it work for 5-element domain? It probably doesn't... But since \(2+2+2>5\), there are only few possibilities for the case when we can apply a linear restriction twice. I updated my algorithm a bit for these cases.

Theorem

CSP Dichotomy conjecture holds for domain 5:

\[
\text{CSP} (G) \text{ is tractable if there exists a WNU preserving } G, \text{ and NP-complete otherwise.}
\]

Dmitriy Zhuk
zhuk.dmitriy@gmail.com (Moscow State University)
Does the algorithm work?

I can prove that it works if we don’t apply linear restrictions twice.

Theorem

CSP Dichotomy conjecture holds for domain 5: $\text{CSP}(G)$ is tractable if there exists a WNU preserving $G$, and NP-complete otherwise.

Theorem

If an algebra $A$ omits unary type and any type, then Steps 1-3 of the algorithm solve $\text{CSP}(A)$. 
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
It probably doesn’t… But
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
It probably doesn’t… But
- Since $2+2+2>5$, 
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
It probably doesn’t... But

- Since $2+2+2 > 5$, there are only few possibilities for the case when we can apply a linear restriction twice.
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
It probably doesn’t... But

- Since $2+2+2>5$, there are only few possibilities for the case when we can apply a linear restriction twice.
- I updated my algorithm a bit for these cases.
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
It probably doesn’t... But
- Since $2 + 2 + 2 > 5$, there are only few possibilities for the case when we can apply a linear restriction twice.
- I updated my algorithm a bit for these cases.

Theorem
CSP Dichotomy conjecture holds for domain 5: \( \text{CSP}(G) \) is tractable if there exists a WNU preserving \( G \), and NP-complete otherwise.
Does the algorithm work?
I can prove that it works if we don’t apply linear restrictions twice.

Why does it work for 5-element domain?
It probably doesn’t... But
- Since $2+2+2>5$, there are only few possibilities for the case when we can apply a linear restriction twice.
- I updated my algorithm a bit for these cases.

Theorem
CSP Dichotomy conjecture holds for domain 5: \( \text{CSP}(G) \) is tractable if there exists a WNU preserving \( G \), and NP-complete otherwise.

Theorem
If an algebra \( A \) omits unary type and affine type, then Steps 1-3 of the algorithm solve \( \text{CSP}(A) \).
Thank you for your attention