Constraint Satisfaction Problems

Tractable Constraint Languages

Bernhard Nebel, **Julien Hué**, and **Stefan Wölfl**

Albert-Ludwigs-Universität Freiburg

June 27 & July 2/4, 2012

Expressiveness vs. complexity

- For some restricted constraint languages we know some polynomial time algorithms that solve each instance of that language
- Restricting constraint languages entails restricting expressiveness, i.e., the class of problems that can be expressed in the language

$\rightsquigarrow$ Expressiveness vs computational complexity?

CSP instances aka constraint networks

Definition

An instance of a constraint satisfaction problem (i.e., a constraint network) is a triple

$$P = \langle V, D, C \rangle ,$$

where:

- V is a non-empty and finite set of variables,
- D is an arbitrary set (domain),
- C is a finite set of constraints $C_1, \ldots, C_q$, i.e., each constraint C_i is a pair (s_i, R_i), where s_i is a tuple of variables of length m_i and R_i is an m_i-ary relation on D (s_i: constraint scope; R_i: constraint relation).

Restricting the general CSP

The general CSP decision problem is the following: Given an instance of a constraint satisfaction problem, N, determine if there exists solution to N, i.e., determine whether

$$\mathsf{Sol}(N) := \{(d_1, \dots, d_n) \in D^n \,:\, a(v_i) = d_i \text{ for a solution } a \text{ of } N\}$$

(where n is the number of variables of V) is not empty.

Restricting the general CSP:

- structural restriction: consider just CSP instances with particular constraint scopes (e. g., where the network hypergraph has specific properties)
- relational restriction: consider just CSP instances, where the constraint relations have a specific form or specific properties

Restricting the general CSP

The general CSP decision problem is the following: Given an instance of a constraint satisfaction problem, N, determine if there exists solution to N, i.e., determine whether

$$\mathsf{Sol}(N) := \{(d_1, \ldots, d_n) \in D^n \,:\, a(v_i) = d_i \text{ for a solution } a \text{ of } N\}$$

(where n is the number of variables of V) is not empty.

Restricting the general CSP:

- structural restriction: consider just CSP instances with particular constraint scopes (e. g., where the network hypergraph has specific properties)
- relational restriction: consider just CSP instances, where the constraint relations have a specific form or specific properties

Tractable Constraint Languages

Constraint language

Definition

A constraint language is an arbitrary set of relations, Γ, defined over some fixed domain (denoted by $D(\Gamma)$).

Definition

For a constraint language Γ, let CSP(Γ) be the class of CSP instances $N = \langle V, D, C \rangle$ such that for each $(s, R) \in C$, $R \in \Gamma$. CSP(Γ) is called the relational subclass associated with Γ.

Definition

A finite constraint language Γ is tractable if there exists a polynomial algorithm that solves all instances of CSP(Γ). An infinite constraint language Γ is tractable if each finite subset of the language is tractable.

Following, we present some examples:

Example: CHIP language

CHIP is a constraint language for arithmetic and other constraints. Basic constraints in CHIP are so-called:

- domain constraints: unary constraints that restrict the domains of variables to a finite set of natural numbers
- arithmetic constraints: constraints of one of the forms

$$ax = by + c$$
$$ax \leq by + c$$
$$ax \geq by + c$$

  $(a, b, c \in \mathbb{N}, a \neq 0)$. If these equations are conceived of as relations, the resulting constraint language is tractable.

The language is still tractable if we allow for relations expressed by

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \geq by + c$$
$$ax_1 \cdots x_n \geq by + c$$
$$(a_1x_1 \geq b_1) \vee \cdots \vee (a_nx_n \geq b_n) \vee (ay \geq b)$$

Example: Linear relations

Let D be any field (e.g., the field of real numbers).
A linear relation on D is any relation defined by some system of linear equations:

$$a_1x_1 + \cdots + a_nx_n = r \qquad (a_1, \ldots, a_n, r \in D).$$

Then any instance of CSP(Γ_{lin}) can be represented by a system of linear equations over D, and hence can be solved in polynomial time (apply Gaussian elimination).

Hence, the language of all linear relations over D is tractable.

Example: Relations on finite orderings

Let D be a finite ordered set.
Consider the binary disequality relation

$$\neq_D = \{(d_1, d_2) \in D^2 : d_1 \neq d_2\}$$

The class of CSP instances CSP($\{\neq_D\}$) corresponds to the graph colorability problem with $|D|$ colors.
CSP($\{\neq_D\}$) is tractable if $|D| \leq 2$ or $|D| = \infty$, and intractable, otherwise.

The ternary betweenness relation over D is defined by:

$$B_D = \{(a, b, c) \in D^3 : a < b < c \vee c < b < a\}$$

CSP($\{B_D\}$) is tractable if $|D| \leq 4$, and intractable if $|D| \geq 5$.

Example: Connected row-convex relations

Let $D = \{d_1, \ldots, d_n\}$ be a finite (totally) ordered set.
For a binary relation R over D, the matrix representation of R is an $n \times n$ 0,1-matrix M_R, where $M_R[d, d'] = 1$ iff $(d, d') \in R$.
The pruned matrix representation of R results from the matrix representation of R, when we remove all rows and columns in which only 0's occur.
R is connected row-convex, if in the pruned matrix representation of R, the pattern of 1's is connected along each column, along each row, and forms a connected 2-dimensional region.

For example,

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

The constraint language on any class of connected row-convex relations is tractable.

Example: Boolean constraints

Let $D = \{0, 1\}$.
The class of CSP instances CSP$(\{N_D\})$, where

$$N_D = D^3 \setminus \{(0,0,0),(1,1,1)\}$$

is the not-all-equal relation over D, is intractable.
CSP$(\{N_D\})$ corresponds to the not-all-equal satisfiability problem (NAE-3SAT), which is known to be NP-hard.

The class of CSP instances CSP$(\{T_D\})$, where

$$T_D = \{(0,0,1),(0,1,0),(1,0,0)\},$$

is intractable.
CSP$(\{T_D\})$ corresponds to the one-in-three satisfiability problem (1-in-3 SAT).

Example: 0/1/all-relations

Let D be an arbitrary finite set. A relation R over D is called 0/1/all-relation if one of the following conditions holds:

- R is unary;
- $R = D_1 \times D_2$ for subsets D_1, D_2 of D;
- $R = \{(d, \pi(d)) : d \in D_1\}$, for some subset $D_1 \subseteq D$ and some permutation π of D;
- $R = \{(a, b) \in D_1 \times D_2 : a = d_1 \vee b = d_2\}$, for some subsets D_1, D_2 of D and some elements $d_1 \in D_1, d_2 \in D_2$.

The language defined by all 0/1/all-relations is tractable.

It is even maximal tractable: if we add any binary relation over D that is not a 0/1/all-relation, then the resulting constraint language becomes intractable.

max-closed relations

Let $(D, <)$ be a linear order. Define $\max : D \times D \to D$ in the usual way, i.e., $\max(a, b) = a$ if $a > b$, and $\max(a, b) = b$, otherwise.

We extend $\max$ to a function that can be applied to tuples, i.e., we define $\max : D^k \times D^k \to D^k$ by

$$\max((a_1, \ldots, a_k), (b_1, \ldots, b_k)) \\ := (\max(a_1, b_1), \ldots, \max(a_k, b_k)).$$

Definition

An n-ary relation R over D is max-closed if for all $(a_1, \ldots, a_n)$, $(b_1, \ldots, b_n) \in R$,

$$\max((a_1, \ldots, a_n), (b_1, \ldots, b_n)) \in R.$$

$\max$-closed relations and tractability

Lemma

Let Γ be a constraint language with max-closed relations only. Then CSP(Γ) is tractable.

Proof.

Enforce generalized arc consistency. If any domain of the resulting network is empty, the network is inconsistent. Otherwise, set each variable to its maximal value, ... $\square$

Example: max-closed relations

Consider the CHIP language. All relations of CHIP are max-closed. Hence any set of equations can be solved by establishing gen. arc consistency.

For example, consider a CSP instance with domain $\{1, \ldots, 5\}$, variables $\{v, w, x, y, z\}$, and equations

$$w \neq 3,\ \ z \neq 5,\ \ 3v \leq z,\ \ y \geq z + 2,\\ 3x + y + z \geq 5w + 1,\ \ wz \geq 2y.$$

Enforcing gen. arc consistency results in:

$$D(v) = \{1\},\ \ D(w) = \{4\},\ \ D(x) = \{4, 5\},\\ D(y) = \{5\},\ \ D(z) = \{3\}.$$

Hence

$$v \mapsto 1, w \mapsto 4, x \mapsto 5, y \mapsto 5, z \mapsto 3$$

is a solution of the constraint network.

Schaefer's Dichotomy Theorem

Boolean constraint languages

The key result in the literature on tractable constraint languages is Schaefer's Dichotomy Theorem (1978).

Definition

A Boolean constraint language is a constraint language over the two-element domain $D = \{0, 1\}$.

Schaefer's theorem states that any Boolean constraint language is either tractable or NP-complete. Moreover, it provides a classification of all tractable constraint languages.

Schaefer's theorem

Theorem (Schaefer 1978)

Let Γ be a Boolean constraint language. Then Γ is tractable if at least one of the following conditions is satisfied:

1. *Each relation in Γ contains the tuple $(0, \ldots, 0)$.*
2. *Each relation in Γ contains the tuple $(1, \ldots, 1)$.*
3. *Each relation in Γ is definable by a formula in CNF s. t. each conjunct has at most one negative literal.*
4. *Each relation in Γ is definable by a formula in CNF s. t. each conjunct has at most one positive literal.*
5. *Each relation in Γ is definable by a formula in CNF s. t. each conjunct has at most two literals.*
6. *Each relation in Γ is the set of solutions of a system of linear equations over the finite field with 2 elements.*

In all other cases, Γ is NP-complete.

Algorithm selector

Let Γ be a Boolean constraint language.

Class 1: any CSP instance N can be solved by simply assigning 0 to each variable of N.

Class 2: cf. Class 1 ($v \mapsto 1$).

Class 6: any CSP instance N can be solved by applying the Gaussian elimination procedure.

Class 5: any CSP instance N can be solved by resolution: in this case CSP(Γ) corresponds to the 2-SAT satisfiability problem and this can be solved efficiently by resolution.

Class 4: any CSP instance N can be solved by unit resolution: here CSP(Γ) corresponds to the Horn-SAT satisfiability problem, which can be solved efficiently by unit resolution.

Class 3: cf. Class 4 ("anti-Horn").

Relational Clones

Gadgets

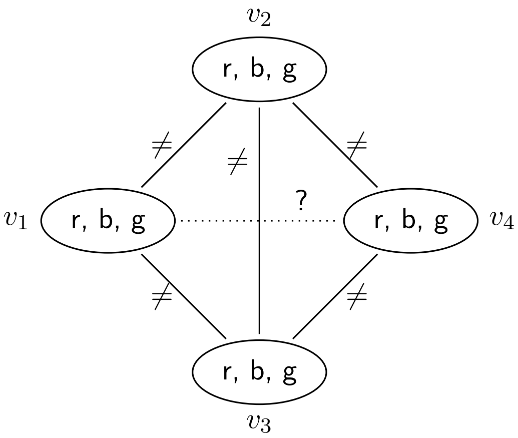
Definition

Let Γ be constraint language and R be a relation on $D(\Gamma)$. R is expressible in Γ if there exists a CSP instance $N \in \mathsf{CSP}(\Gamma)$ and a sequence of variables $x_1, \ldots, x_r$ in N such that

$$R = \pi_{x_1,\ldots,x_r}(\mathsf{Sol}(N)).$$

N is referred to as a gadget for expressing R in $\mathsf{CSP}(\Gamma)$, the sequence $x_1, \ldots, x_r$ as construction site for R.

Example



Which relation is expressed by the edge (v_1, v_4)?

Relational clones

Expressiveness can also be reformulated in the following way: Let Γ, Γ' be constraint languages (def. on the same domain D).

Definition

Γ' is a relational clone of Γ if Γ' contains each relation definable by a FO-formula with

- relations from $\Gamma \cup \{=_D\}$,
- conjunctions, and
- existential quantification.

(Formulae of this form are called primitive positive formulae.)

Definition

Let Γ be a constraint language. $\langle\Gamma\rangle$ denotes the smallest relational clone containing Γ, the clone generated by Γ.

Example

Consider a Boolean constraint language with the following relations:

$$R_1 = \{(0,1),(1,0),(1,1)\} \qquad R_2 = \{(0,0),(0,1),(1,0)\}.$$

The relational clone generated by the set of these two relations contains all 16 binary Boolean relations. For example:

$$
\begin{array}{ll}
R_3 := \{(0,1),(1,0)\} & R_1(v_1,v_2) \wedge R_2(v_1,v_2) \\
R_4 := \{(0,0),(1,0),(1,1)\} & \exists y(R_1(v_1,y) \wedge R_2(y,v_2)) \\
R_5 := \{(0,0),(1,1)\} & v_1 = v_2 \\
R_6 := \{(0,0)\} & R_2(v_1,v_2) \wedge R_5(v_1,v_2) \\
R_7 := \{(1,1)\} & R_1(v_1,v_2) \wedge R_5(v_1,v_2) \\
R_8 := \{(0,1)\} & \exists y(R_6(v_1,y) \wedge R_1(y,v_2)) \\
\ldots &
\end{array}
$$

Reducibility I

Theorem

Let Γ be a set of relations on a fixed domain D, and let Δ be a finite subset of $\langle\Gamma\rangle$. Then there exists a polynomial time reduction from CSP(Δ) to CSP(Γ).

Reducibility II

Proof.

Let $\Delta = \{S_1, \ldots, S_k\}$ be a finite set of relations, where each S_j is definable by a pp-formula with relations from Γ and the relation $=_D$. For each S_j fix such a formula $\varphi_j(x_1, \ldots, x_{r_j})$, where r_j is the arity of S_j. Without loss of generality, we may assume that each $\varphi_j(x_1, \ldots, x_{r_j})$ has the form

$$\exists u_1 \ldots u_m (R_1(w_1^1, \ldots, w_{k_1}^1) \wedge \cdots \wedge R_n(w_1^n, \ldots, w_{k_n}^n)) \qquad (1)$$

where $w_1^1, \ldots, w_{k_1}^1, \ldots, w_1^n, \ldots, w_{k_n}^n \in \{x_1, \ldots, x_{r_j}, u_1, \ldots, u_m\}$ for some auxiliary variables $u_1, \ldots, u_m$, and $R_1, \ldots, R_n \in \Gamma \cup \{=_D\}$.

...

Reducibility III

...

Let $N = \langle V, D, C \rangle$ be an arbitrary instance in CSP(Δ). Initially, set $V' := V, D' := D, C' := C$. For each constraint (s, R) (where $s = (v_1, \ldots, v_r)$) of N, proceed as follows:

1. add the auxiliary variables $u_1, \ldots, u_m$ to V' (always add new variables, rename variables if necessary (also in (1)))
2. remove (r, R) from C' and instead add to C' the constraints (cf. (1)):

$$((w_1^1, \ldots, w_{k_1}^1), R_1), \ldots, (w_1^n, \ldots, w_{k_n}^n, R_n)$$

The CSP instance N' obtained by this procedure is contained in CSP($\Gamma \cup \{=_D\}$) and is obviously equivalent to N. Furthermore, from N' we can obtain a CSP instance N'' in CSP(Γ) by deleting constraints of the form $((v_i, v_j), =_D)$ and replacing any occurrence of v_j by v_i. Obviously, both transformation can be done in polynomial time. □

Reducibility IV

Corollary

A constraint language Γ is tractable if and only if its relational clone $\langle\Gamma\rangle$ is tractable. Γ is NP-complete if and only if $\langle\Gamma\rangle$ is NP-complete.

Remark: Γ is called NP-complete if CSP(Δ) is NP-complete for some finite subset $\Delta \subseteq \Gamma$.

Corollary

Let Γ be a constraint language and let R be a relation. R is expressible in Γ if and only if $R \in \langle\Gamma\rangle$.

Expressiveness

The indicator problem

Let $k \geq 1$ be a fixed natural number.
Let $s = (x_1, \ldots, x_m)$ be a list of k-tuples in D^k.
Let R be an n-ary relation on D.

We say, that s matches R if $n = m$ and if for each $1 \leq i \leq k$, the n-tuple $(x_1[i], \ldots, x_n[i])$ is in R.

Let now Γ be a fixed finite constraint language over a finite domain.
Set $I_k(\Gamma) = \langle V, D, C \rangle$, where

$$V := D^k$$
$$C := \{(s, R) : R \in \Gamma, s \text{ matches } R\}$$

Note: $I_k(\Gamma) \in \mathrm{CSP}(\Gamma)$ and contains constraints from Γ on every possible scope which matches some relation in Γ.

Definition

$I_k(\Gamma)$ is said to be the indicator problem of order k for Γ.

Example: $\neg, \oplus$

Consider the Boolean constraint language containing the unary relation $\neg$ and the exclusive-or relation $\oplus$, i.e.,

$$R_{\oplus} = \{(0,1),(1,0)\} \quad \text{and} \quad R_{\neg} = \{(0)\}.$$

The 3-rd order indicator problem of this language is:

$$\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{array}$$

Example: $\neg$, $\oplus$

Consider the Boolean constraint language containing the unary relation $\neg$ and the exclusive-or relation $\oplus$, i.e.,

$$R_\oplus = \{(0,1),(1,0)\} \quad \text{and} \quad R_\neg = \{(0)\}.$$

The 3-rd order indicator problem of this language is:

$$\begin{array}{cccccccc}
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 1 & 0
\end{array}$$

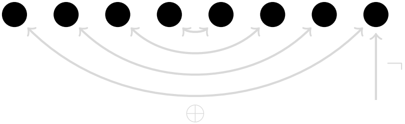
Example: $\neg, \oplus$

Consider the Boolean constraint language containing the unary relation $\neg$ and the exclusive-or relation $\oplus$, i.e.,

$$R_{\oplus} = \{(0,1),(1,0)\} \quad \text{and} \quad R_{\neg} = \{(0)\}.$$

The 3-rd order indicator problem of this language is:

$$\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{array}$$



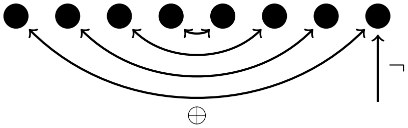
Example: $\neg, \oplus$

Consider the Boolean constraint language containing the unary relation $\neg$ and the exclusive-or relation $\oplus$, i.e.,

$$R_\oplus = \{(0,1),(1,0)\} \quad \text{and} \quad R_\neg = \{(0)\}.$$

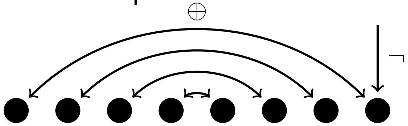
The 3-rd order indicator problem of this language is:

$$\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{array}$$



Example (cont'd): $\neg$, $\oplus$

Solutions of this indicator problem:



1	1	1	1	0	0	0	0
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0

Solutions

1	0	1	1	0	0	1	0
1	0	1	0	1	0	1	0
1	0	0	1	0	1	1	0
1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	0
1	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0
1	1	0	0	1	1	0	0

Expressiveness and the indicator problem

Theorem (Jeavons (1998))

Let Γ be a finite constraint language over some finite domain D and let $R = \{t_1, \ldots, t_k\}$ be any n-ary relation on D. Equivalent are:

(a) *R is expressible in Γ (i.e., $R \in \langle\Gamma\rangle$).*

(b) *$I_k(\Gamma)$ is a gadget for expressing R with construction site $(x_1, \ldots, x_n)$, where for each $1 \leq i \leq n$,*

$$x_i := (t_1[i], \ldots, t_k[i]).$$

Proof.

The direction from (b) to (a) is trivial, since $I_k(\Gamma)$ is contained in CSP(Γ). The other direction will be proved later. □

Example: $\neg, \oplus$

Problem: Is the implication expressible in the Boolean language $\{\neg, \oplus\}$?

Consider the 3rd indicator problem (since $R_{\Rightarrow}$ has three elements $(1,1), (0,0), (0,1)$). Consider the variables $v = (1,0,0)$ and $w = (1,0,1)$:

1	1	1	1	0	0	0	0
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0

Solutions

1	0	1	1	0	0	1	0
1	0	1	0	1	0	1	0
1	0	0	1	0	1	1	0
1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	0
1	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0
1	1	0	0	1	1	0	0

Example: $\neg, \oplus$

Problem: Is the implication expressible in the Boolean language $\{\neg, \oplus\}$?

Consider the 3rd indicator problem (since $R_{\Rightarrow}$ has three elements $(1,1),(0,0),(0,1)$). Consider the variables $v = (1,0,0)$ and $w = (1,0,1)$:

1	1	1	1	0	0	0	0
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0

Solutions

1	0	1	1	0	0	1	0
1	0	1	0	1	0	1	0
1	0	0	1	0	1	1	0
1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	0
1	1	1	0	1	0	0	0
1	1	0	1	0	1	0	0
1	1	0	0	1	1	0	0

Example: $\neg, \oplus$

Problem: Is the implication expressible in the Boolean language $\{\neg, \oplus\}$?

Consider the 3rd indicator problem (since $R_{\Rightarrow}$ has three elements $(1,1),(0,0),(0,1)$). Consider the variables $v = (1,0,0)$ and $w = (1,0,1)$:

1	1	**1**	**1**	0	0	0	0
1	1	**0**	**0**	1	1	0	0
1	0	**1**	**0**	1	0	1	0

Solutions

1	0	**1**	**1**	0	0	1	0
1	0	**1**	**0**	1	0	1	0
1	0	**0**	**1**	0	1	1	0
1	0	**0**	**0**	1	1	1	0
1	1	**1**	**1**	0	0	0	0
1	1	**1**	**0**	1	0	0	0
1	1	**0**	**1**	0	1	0	0
1	1	**0**	**0**	1	1	0	0

Example: $\neg, \oplus$

Problem: Is the implication expressible in the Boolean language $\{\neg, \oplus\}$?

Consider the 3rd indicator problem (since $R_\Rightarrow$ has three elements $(1,1),(0,0),(0,1)$). Consider the variables $v = (1,0,0)$ and $w = (1,0,1)$:

1	1	**1**	**1**	0	0	0	0
1	1	**0**	**0**	1	1	0	0
1	0	**1**	**0**	1	0	1	0
Solutions							
1	0	**1**	**1**	0	0	1	0
1	0	**1**	**0**	1	0	1	0
1	0	**0**	**1**	0	1	1	0
1	0	**0**	**0**	1	1	1	0
1	1	**1**	**1**	0	0	0	0
1	1	**1**	**0**	1	0	0	0
1	1	**0**	**1**	0	1	0	0
1	1	**0**	**0**	1	1	0	0

From this we obtain that $\pi_{(v,w)}(\mathrm{Sol}(I_3(\Gamma))) = D \times D \neq R_\Rightarrow$.
Thus, the implication is not expressible.

Polymorphisms

Polymorphisms

Let f be a k-ary operation, i.e., a function $f : D^k \rightarrow D$.
For any collection of n-tuples, $t_1, \ldots, t_k \in D^n$, let $f(t_1, \ldots, t_k)$ be defined as the n-tuple:

$$\big(f(t_1[1], \ldots, t_k[1]), \ldots, f(t_1[n], \ldots, t_k[n])\big).$$

Definition

Let $f : D^k \rightarrow D$ be a k-ary operation, and R be an n-ary relation.
f is a polymorphism of R (or: R is invariant under f) if for all $t_1, \ldots, t_k \in R$, $f(t_1, \ldots, t_k) \in R$.

Polymorphisms and invariant relations

Let Γ be a set of relations on a fixed domain D, and let F be a set of operations on D. Then define:

$\mathsf{Pol}(\Gamma)$:	the set of operations on D that preserve each relation in Γ
$\mathsf{Inv}(F)$:	the set of relations on D that are invariant under each operation of F

Lemma

Pol and Inv define anti-monotone functions, and are related by the following Galois connection:

$$\Gamma \subseteq \mathsf{Inv}(F) \iff F \subseteq \mathsf{Pol}(\Gamma).$$

In particular, it holds:

$$\Gamma \subseteq \mathsf{Inv}(\mathsf{Pol}(\Gamma)) \quad \textit{and} \quad F \subseteq \mathsf{Pol}(\mathsf{Inv}(F)).$$

Indicator problem and polymorphisms

Lemma

Let Γ be a constraint language. The solutions of the k-th indicator problem $I_k(\Gamma)$ are precisely the k-ary polymorphisms of Γ.

Proof.

Apply the definitions . . . $\square$

Expressiveness and polymorphisms

Lemma

Let Γ be a constraint language over some domain D. If $f : D^k \to D$ is a polymorphism of each $R \in \Gamma$, then f is a polymorphism of each $R \in \langle\Gamma\rangle$.

Proof.

Induction on primitive positive formula (exercise). □

Expressiveness and the indicator problem (Part 2)

The following lemma completes the proof of Jeavons' theorem:

Lemma

Let $R = \{t_1, \dots, t_k\}$ be an n-ary relation (over some finite domain D). For $1 \leq i \leq n$, set $x_i := (t_1[i], \dots, t_k[i])$. If R is expressible in Γ, then $R = \pi_{x_1,\dots,x_n}(\mathsf{Sol}(I_k(\Gamma)))$.

Proof.

Blackboard. □

Expressiveness and Invariants

Theorem

For any constraint language Γ over some finite domain D,

$$\langle \Gamma \rangle = \mathsf{Inv}(\mathsf{Pol}(\Gamma))$$

Proof.

$\subseteq$ is clear. For the converse let R be an n-ary relation that is invariant for each polymorphism of Γ. We have to show that $R \in \langle \Gamma \rangle$. Let $R = \{t_1, \ldots, t_k\}$ and consider the k-th indicator problem of Γ. First define $x_i := (t_1[i], \ldots, t_k[i])$ $(1 \leq i \leq n)$, then consider $R_t = \pi_{x_1,\ldots,x_n}(\mathsf{Sol}(I_k(\Gamma)))$. By one of the lemmas above, R is expressible if $R = R_t$.
$R_t \subseteq R$ follows from the facts that every solution of $I_k(\Gamma)$ is a k-ary polymorphism and that each polymorphism of Γ preserves R. For $R \subseteq R_t$, consider t_j in R. Now the j-th projection function $p_j : D^k \to D$ is a polymorphism. Hence $t_j = p_j(t_1, \ldots, t_k) \in R$. $\square$

Expressiveness, Polymorphisms, and Complexity

Corollary

A relation R on a finite domain is expressible by a constraint language if and only if $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\{R\})$.

Corollary

Let Γ and Δ be constraint languages on a finite domain. If Δ is finite and $\mathrm{Pol}(\Gamma) \subseteq \mathrm{Pol}(\Delta)$*, then CSP*$(\Delta)$ *is polynomial-time reducible to CSP*(Γ).

Tractability over Finite Domains

Operations

Following, we study k-ary operations $f : D^k \to D$.

Definition

- f is idempotent if for each $x \in D$, $f(x, \dots, x) = x$.
- Given $k = 3$, f is a majority operation if for all $x, y \in D$,

$$f(x, x, y) = f(x, y, x) = f(y, x, x) = x.$$

- Given $k = 3$, f is a Mal'tsev operation if for all $x, y \in D$,

$$f(y, y, x) = f(x, y, y) = x.$$

- f is conservative if for all $x_1, \dots, x_k \in D$,

$$f(x_1, \dots, x_k) \in \{x_1, \dots, x_k\}.$$

Operations (cont'd)

Definition

- Given $k = 2$, f is a semi-lattice operation if it is
  - associative (i.e., $f(x, f(y, z)) = f(f(x, y), z)$),
  - commutative (i.e., $f(x, y) = f(y, x)$), and
  - idempotent.
- Given $k = 3$ and an Abelian group structure on D, f is affine if for all $x, y, z \in D$,

$$f(x, y, z) = x - y + z.$$

- Given $k \geq 3$, f is a near-unanimity operation if for all $x, y \in D$,

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

Operations (cont'd)

Definition

- f is essentially unary if there exists an $1 \leq i \leq k$ and a unary non-constant operation g on D such that for all $x_1, \ldots, x_k \in D$,

$$f(x_1, \ldots, x_k) = g(x_i).$$

  If g is the identity operation, then f is called a projection.

- Given $k \geq 3$, f is a semi-projection if f is not a projection and there exists an $1 \leq i \leq k$, such that for all $x_1, \ldots, x_k \in D$ with $|\{x_1, \ldots, x_k\}| < k$,

$$f(x_1, \ldots, x_k) = x_i.$$

A necessary condition for tractability

Theorem

Given $P \neq NP$, any tractable constraint language Γ over a finite domain has a solution to an indicator problem $I_k(\Gamma)$ that defines

- *a constant operation,*
- *a majority operation,*
- *an idempotent binary operation,*
- *an affine operation, or*
- *a semi-projection.*

Boolean CSPs

The complexity of any language over a domain of size 2 can be determined by considering the solutions of its 3rd order indicator problem. The problem is intractable unless this indicator problem has one of the following six solutions:

Variables									
1	1	1	1	0	0	0	0		
1	1	0	0	1	1	0	0		
1	0	1	0	1	0	1	0		
Solutions								**Schaefer class**	**Name**
0	0	0	0	0	0	0	0	1	Constant 0
1	1	1	1	1	1	1	1	2	Constant 1
1	1	1	1	1	1	1	0	3	Anti-Horn
1	0	0	0	0	0	0	0	4	Horn-SAT
1	1	1	0	1	0	0	0	5	2-SAT
1	0	0	1	0	1	1	0	6	Linear

Example: $\neg$, $\oplus$

●	●	●	●	●	●	●	●
1	1	1	1	0	0	0	0
1	1	0	0	1	1	0	0
1	0	1	0	1	0	1	0
Solutions							
1	0	1	1	0	0	1	0
1	0	1	0	1	0	1	0
1	0	0	1	0	1	1	0
1	0	0	0	1	1	1	0
1	1	1	1	0	0	0	0
1	**1**	**1**	**0**	**1**	**0**	**0**	**0**
1	1	0	1	0	1	0	0
1	1	0	0	1	1	0	0

Sufficient conditions: Semi-lattice operations

In what follows let Γ always be a constraint language over a finite domain D. We present some sufficient criteria for (in-) tractability.

Theorem

If $\mathrm{Pol}(\Gamma)$ *contains a semi-lattice operation, then*

- Γ *is tractable, and*
- *each instance of CSP*(Γ) *can be solved by enforcing generalized arc consistency.*

Examples

Example 1:

If Γ is the Boolean constraint language containing all relations expressible by conjunctions of Horn clauses, then

$$\wedge : \{0,1\}^2 \to \{0,1\}$$

is a semi-lattice operation that is a polymorphism of Γ.

Example 2:

If D is ordered, then $\max$ is a semi-lattice operation, which is a polymorphism of each set of max-closed relations.

Sufficient conditions: Conservative operations

Theorem

If $\mathrm{Pol}(\Gamma)$ *contains a conservative and commutative operation, then* Γ *is tractable.*

Note: If Γ contains all unary relations on D, then all operations in $\mathrm{Pol}(\Gamma)$ are conservative.

Sufficient conditions: Near-unanimity operations

Theorem

If Pol(Γ) *contains a* k*-ary near-unanimity operation, then*

- Γ *is tractable.*
- *Each instance of CSP*(Γ) *can be solved by enforcing strong* k*-consistency.*

Proof.

Blackboard. $\square$

Examples

Example 3:

Let Γ be the Boolean constraint language that consists of all relations definable by a PL-formula in CNF s. t. each conjunct has at most two literals.

Then

$$d(x, y, z) := (x \wedge y) \vee (y \wedge z) \vee (x \wedge z)$$

is a near-unanimity operation on $\{0, 1\}$ and a polym. of Γ.

Example 4:

The 0/1/all relations are invariant under the ternary operation

$$d(x, y, z) := \begin{cases} x & \text{if } y \neq z \\ y & \text{else} \end{cases}$$

which is a near-unanimity operation.

Sufficient conditions: Mal'tsev operations

Theorem

If $\mathrm{Pol}(\Gamma)$ *contains a* k*-ary Mal'tsev operation, then CSP*(Γ) *is tractable.*

Note: Affine relations are Mal'tsev operations.

Reduced constraint languages

Lemma

Let Γ be a constraint language over D, and let f be a unary operation in $\mathrm{Pol}(\Gamma)$. *Let $f(\Gamma)$ be the set of all $f(R) := \{f(t) : t \in R\}$ with $R \in \Gamma$. Then, CSP(Γ) is polynomial-time equivalent to CSP$(f(\Gamma))$.*

Definition

A constraint language Γ is reduced if all its unary polymorphisms are surjective.

Note: Each constraint language can be transformed into a reduced language. For this find all unary polymorphisms by generating and solving the 1st order indicator problem. Choose one of these polymorphisms f with a minimal number of values in its range.

A sufficient condition for intractability

Theorem

Let Γ be a constraint language over a finite domain. If Pol(Γ) *contains only essentially unary operations, then CSP(Γ) is NP-complete.*

Proof idea:

We can assume that Γ is reduced. One can show that

- $\neq_D$ is in Inv(Pol(Γ));
- if $|D| = 2$, Inv(Pol(Γ)) contains the not-all-equal relation:

$$D^3 \setminus \{(x, x, x) : x \in D\}$$

which ensures that CSP(Γ) intractable. $\square$

Towards a classification

It can be shown that for any reduced constraint language Γ on a finite domain D, one of the following conditions holds:

- Pol(Γ) contains a constant operation;
- Pol(Γ) contains a ternary near-unanimity operation;
- Pol(Γ) contains a Mal'tsev operation;
- Pol(Γ) contains an idempotent binary operation;
- Pol(Γ) contains a semi-projection;
- Pol(Γ) contains essentially unary operations only.

Literature

David Cohen and Peter Jeavons.
Tractable constraint languages.
In: R. Dechter *Constraint Processing*, Chapter 11, Morgan Kaufmann, 2003

Andrei Bulatov, Andrei Krokhin, and Peter Jeavons.
The complexity of maximal constraint languages.
In: *Proceedings of STOC'01*, 2001

Andrei Bulatov, P. Jeavons, and Andrei Krokhin.
Classifying the complexity of constraint using finite algebras.
SIAM J. Comput. 34(3), 2005

David Cohen and Peter Jeavons.
The complexity of constraint languages.
In: F. Rossi, P. v. Beek, and T. Walsh, *Handbook of Constraint Programming*, Elsevier, 2006