THE CONSTRAINT SATISFACTION PROBLEM AND UNIVERSAL ALGEBRA

LIBOR BARTO

Abstract. This paper gives a brief survey of current research on the complexity of the constraint satisfaction problem over fixed constraint languages.

§1. Introduction. The Constraint Satisfaction Problem (CSP) provides a common framework for expressing a wide range of both theoretical and real-life combinatorial problems [50]. One solves an instance of the CSP by assigning values to its variables so that its constraints are satisfied.

The topic of this paper is a very active theoretical subfield which studies the computational complexity of the CSP over a fixed constraint language. This restricted framework is still broad enough to include many decision problems in the class NP, yet it is narrow enough to potentially allow for a complete classification of all such CSP problems.

One particularly important achievement is the understanding of what makes the problems over a fixed constraint language computationally easy or hard. It is not surprising that hardness comes from a lack of symmetry. However, the usual objects capturing symmetry, automorphisms (or endomorphisms) and their groups (or semigroups), are not sufficient in this context. It turns out that the complexity of the CSP over a template is determined by more general symmetries of it: polymorphisms and their clones.

My aim is to introduce the basics of this exciting area and to highlight selected deeper results, in a way that is understandable to readers with a basic knowledge of computational complexity (see [47, 1]). The presentation of the material is based on my talk "Universal algebra and the constraint satisfaction problem" delivered at the Association for Symbolic Logic North American Annual Meeting held in Boulder, Colorado, in 2014. A shorter version has appeared in SIGLOG News [3].

§2. CSP over a fixed constraint language. A constraint – such as $R(x_3, x_1, x_4)$ – restricts the allowed values for a tuple of variables – in this case (x_3, x_1, x_4) – to be an element of a particular relation on the domain – in this case $R \subseteq D^3$.[1] By an n-ary *relation* R on a domain D we mean a subset of the n-th cartesian

The author gratefully acknowledges the support of the Grant Agency of the Czech Republic, grant GAČR 13-01832S.

[1] There are also different types of constraints considered in the literature, see e.g. Chapter 7 in [50].

power D^n. It is sometimes convenient to work with the corresponding *predicate* which is a mapping from D^n to $\{\text{true}, \text{false}\}$ specifying which tuples are in R. We will use both formalisms, so e.g. $(a, b, c) \in R$ and $R(a, b, c)$ both mean that the triple $(a, b, c) \in D^3$ is from the relation R.

An instance of the CSP is a list of constraints, e.g.,

$$R(x),\ S(y, y, z),\ T(y, w),$$

where R, S, T are relations of appropriate arity on a common domain D and x, y, z, w are variables. A mapping f assigning values from the domain to the variables is a *solution* if it satisfies all the constraints, that is, in our example,

$$R(f(x)) \text{ and } S(f(y), f(y), f(z)) \text{ and } T(f(y), f(w)) \ .$$

A standard formal definition of an instance of the CSP over a finite domain goes as follows.

DEFINITION 2.1. *An* instance of the CSP *is a triple* $P = (V, D, \mathcal{C})$ *with*

- V *a finite set of* variables,
- D *a finite* domain,
- $\mathcal{C}$ *a finite list of constraints, where each constraint is a pair* $C = (\mathbf{x}, R)$ *with*
  - $\mathbf{x}$ *a tuple of variables of length* n*, called the* scope *of* C*, and*
  - R *an* n*-ary relation on* D*, called the* constraint relation *of* C*.*

An assignment*, that is, a mapping* $f : V \to D$*,* satisfies *a constraint* $C = (\mathbf{x}, R)$ *if* $f(\mathbf{x}) \in R$*, where* f *is applied component-wise. An assignment* f *is a* solution *if it satisfies all constraints.*

Three basic computational problems associated with an instance are the following:[2]

- **Satisfiability.** Does the given instance have a solution? (A related problem, the *search problem*, is to find some solution if at least one solution exists.)
- **Optimization.** Even if the instance has no solution, find an optimal assignment, i.e., one that satisfies the maximum possible number of constraints. (*Approximation algorithms* are extensively studied, where the aim is, for example, to find an assignment that satisfies at least 80% of the number of constraints satisfied by an optimal assignment.)
- **Counting.** How many solutions does the given instance have? (This problem also has an approximation version: *approximate counting*.)

2.1. Satisfiability over a fixed constraint language. Even the easiest of the problems, satisfiability, is computationally hard: It contains many NP-complete problems including, e.g., 3-SAT (see Example 2.3). However, certain natural restrictions to CSP satisfiability ensure tractability. The main types of restrictions that have been studied are *structural restrictions*, which limit how constraints interact, and *language restrictions*, which limit the choice of constraint relations.

[2]To study the computational complexity of these problems we need to specify a representation of instances. We will assume that the constraint relation in every constraint is given by a list of all its members. Note, however, that for most of the problems considered in this article any reasonable representation can be taken.

In this paper, we focus just on satisfiability problems with language restrictions. Please see [55] for optimization problems and a generalization to valued CSPs, [33] for approximation, [25] for counting, and [12] for a generalization to infinite domains.

DEFINITION 2.2. *A* constraint language $\mathcal{D}$ *is a set of relations on a common finite domain,* D*. We use* $\mathrm{CSP}(\mathcal{D})$ *to denote the set of CSP satisfiability problems whose relations are drawn from* $\mathcal{D}$*.*

2.2. Examples.

EXAMPLE 2.3. *An instance of the standard NP-complete problem [47, 1],* 3-SAT*, is a Boolean formula in conjunctive normal form with exactly three literals per clause. For example, the formula,*

$$\varphi \quad = \quad (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_4 \vee x_5 \vee \neg x_1) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_3)$$

is a satisfiable instance of 3-SAT*. (Any assignment making* x_1 *and* x_2 *false, satisfies* φ*.)* 3-SAT *is equivalent to* $\mathrm{CSP}(\mathcal{D}_{3SAT})$*, where* $D_{3SAT} = \{0,1\}$ *and*

$$\mathcal{D}_{3SAT} = \{S_{ijk} : i,j,k \in \{0,1\}\}, \text{ where } S_{ijk} = \{0,1\}^3 \setminus \{(i,j,k)\} \ .$$

For example, the above formula φ *corresponds to the following instance of* $\mathrm{CSP}(\mathcal{D}_{3SAT})$

$$S_{010}(x_1,x_2,x_3),\ S_{101}(x_4,x_5,x_1),\ S_{111}(x_1,x_4,x_3) \ .$$

More generally, for a natural number k*,* k-SAT *denotes a similar problem where each clause is a disjunction of* k *literals.*

Since 3-SAT *is NP-complete, it follows that* k-SAT *is NP-complete for each* $k \geq 3$*. On the other hand,* 2-SAT *is solvable in polynomial time, and is in fact complete for the complexity class NL (non-deterministic logarithmic space) [47, 1] (see also Example 2.7).*

EXAMPLE 2.4. HORN-3-SAT *is a restricted version of* 3-SAT*, where each clause may have at most one positive literal. This problem is equivalent to* $\mathrm{CSP}(\mathcal{D}_{HornSAT})$ *for* $\mathcal{D}_{HornSAT} = \{S_{011}, S_{101}, S_{110}, S_{111}\}$ *(or just* $\mathcal{D}_{HornSAT} = \{S_{011}, S_{111}\}$*).* HORN-3-SAT *is solvable in polynomial time, in fact, it is a P-complete problem [47, 1].*

EXAMPLE 2.5. *For a fixed natural number* k*, the* k-COLORING *problem is to decide whether it is possible to assign colors* $\{0,1,\ldots,k-1\}$ *to the vertices of an input graph in such a way that adjacent vertices receive different colors. This problem is equivalent to* $\mathrm{CSP}(\mathcal{D}_{kCOLOR})$*, where* $D_k = \{0,1,2,\ldots,k-1\}$ *and* $\mathcal{D}_{kCOLOR} = \{\neq_k\}$ *consists of a single relation – the binary inequality relation* $\neq_k = \{(a,b) \in D_k^2 : a \neq b\}$*.*

Indeed, given an instance of $\mathrm{CSP}(\mathcal{D}_{kCOLOR})$*, we can form a graph whose vertices are the variables and whose edges correspond to the binary constraints (that is,* x *has an edge to* y *iff the instance contains the constraint* $x \neq_k y$*). It is easily seen that the original instance has a solution if and only if the obtained graph is* k*-colorable. The translation in the other direction is similar.*

The k-COLORING *problem is NP-complete for* $k \geq 3$ *[47, 1].* 2-COLORING *is equivalent to deciding whether an input graph is bipartite. It is solvable in polynomial time, in fact, it is an L-complete problem (where L stands for logarithmic space) by a celebrated result of Reingold [49].*

EXAMPLE 2.6. *Let p be a prime number. An input of* 3-LIN(p) *is a system of linear equations over the p-element field* GF(p)*, where each equation contains* 3 *variables, and the question is whether the system has a solution. This problem is equivalent to* CSP($\mathcal{D}_{3LIN_p}$)*, where $D_{3LIN_p} = \mathrm{GF}(p)$ and $\mathcal{D}_{3LIN_p}$ consists of all affine subspaces R_{abcd} of $\mathrm{GF}(p)^3$ of dimension* 2 *or* 3*, where*

$$R_{abcd} = \{(x, y, z) \in \mathrm{GF}(p)^3 : ax + by + cz = d\} \ .$$

This problem is solvable in polynomial time, e.g. by Gaussian elimination.[3] *It is complete for a somewhat less familiar class* $\mathrm{Mod}_p\mathrm{L}$ [24].

EXAMPLE 2.7. *An instance of the s,t-connectivity problem,* STCON*, consists of a directed graph and two of its vertices, s and t. The question is whether there exists a directed path from s to t.*

A closely related (but not identical) problem is CSP($\mathcal{D}_{STCON}$)*, where $D_{STCON} = \{0,1\}$ and $\mathcal{D}_{STCON} = \{C_0, C_1, \leq\}$, $C_0 = \{0\}$, $C_1 = \{1\}$, $\leq = \{(0,0),(0,1),(1,1)\}$. Indeed, given an instance of* CSP($\mathcal{D}_{STCON}$) *we form a directed graph much as we did in Example 2.5 and label some vertices* 0 *or* 1 *according to the unary constraints. Then the original instance has a solution if and only if there is no directed path from a vertex labeled* 1 *to a vertex labeled* 0*. Thus* CSP($\mathcal{D}_{STCON}$) *can be solved by invoking the complement of* STCON*, the s,t-non-connectivity problem, several times.*

Both STCON *and* CSP($\mathcal{D}_{STCON}$) *can clearly be solved in polynomial time. By the Immerman-Szelepcsényi theorem [37, 53] both problems are NL-complete.*

In the same way, the s,t-connectivity problem for undirected graphs is closely related to CSP($\mathcal{D}_{USTCON}$)*, where $D_{USTCON} = \{0,1\}$ and $\mathcal{D}_{USTCON} = \{C_0, C_1, =\}$. These problems are L-complete by [49].*

2.3. The dichotomy conjecture. The most fundamental problem in the area was formulated in the landmark paper by Feder and Vardi [31].

CONJECTURE 2.8 (The dichotomy conjecture). *For every finite*[4] *constraint language $\mathcal{D}$, the problem* CSP($\mathcal{D}$) *is in P or is NP-complete.*

Recall that if P $\neq$ NP, then there are problems of intermediate complexity [42]. Feder and Vardi argued that the class of CSPs over fixed constraint languages is a good candidate for the largest natural class of problems which exhibit a P versus NP-complete dichotomy.

At that time the conjecture was supported by two major cases: the dichotomy theorem for all languages over a two-element domain by Schaefer [51] and the dichotomy theorem for languages consisting of a single binary symmetric relation by Hell and Nešetřil [34].

[3]The problem of solving general systems of linear equations over GF(p) without the restriction on number of variables cannot be faithfully phrased as CSP($\mathcal{D}$) with $\mathcal{D}$ consisting of all affine subspaces, since the input representation of the latter problem can be substantially larger. However, a system of linear equation can be easily rewritten to an instance of 3-LIN(p) by introducing new variables.

[4]It is conjectured in [16] that the dichotomy remains true without the finiteness assumption on $\mathcal{D}$ (the domain D still needs to be finite). Namely, the local-global conjecture states that CSP($\mathcal{D}$) is in P (NP-complete) whenever CSP($\mathcal{D}'$) is in P (NP-complete) for every (some) finite $\mathcal{D}' \subseteq \mathcal{D}$.

Feder and Vardi identified two sources of polynomial-time solvability and made several important contributions towards understanding them. In particular, they observed that the known polynomial cases were tied to algebraic closure properties and asked whether polynomial solvability for CSP can always be explained in such a way. Subsequent papers have shown that this is indeed the case and this connection to algebra brought the area to another level.

The algebraic approach is outlined in section 3 and some fruits of the theory discussed in section 4.

2.4. Alternative views. Note that a constraint language $\mathcal{D}$ with domain D can be viewed as a relational structure $(D; R_1, R_2, \dots)$, or equivalently relational database, with universe D.

Recall that a *conjunctive query* over the database $\mathcal{D}$ is an existential sentence whose quantifier-free part is a conjunction of atoms. $\mathrm{CSP}(\mathcal{D})$ is exactly the problem of deciding whether $\mathcal{D}$ satisfies a given conjunctive query. For example, the instance

$$R(x),\ S(y,y,z),\ T(y,w)$$

has a solution if and only if the sentence

$$(\exists x,y,z,w \in D)\ R(x) \wedge S(y,y,z) \wedge T(y,w)$$

is true in $\mathcal{D}$.

From this perspective, it is natural to ask what happens if we allow some other combination of logical connectives from $\{\exists, \forall, \wedge, \vee, \neg, =, \neq\}$. It turns out that out of the 2^7 cases only 3 are interesting (the other cases either reduce to these, or are almost always easy or hard by known results): $\{\exists, \wedge\}$ which is CSP, $\{\exists, \forall, \wedge\}$ which is so called *quantified CSP*, and $\{\exists, \forall, \wedge, \vee\}$. Determining the complexity of quantified CSP is also an active research area [27] with a possible trichotomy – P, NP-complete or Pspace-complete. Recently, a tetrachotomy was obtained for the last case [45] – for every $\mathcal{D}$, the corresponding problem is either in P, NP-complete, co-NP-complete, or Pspace-complete.

The CSP over a fixed language can also be formulated as the homomorphism problem between relational structures with a fixed target structure [31]. The idea of the translation is shown in Examples 2.5, 2.7.

§3. Universal algebra in CSPs. If a computational problem $\mathcal{A}$ can simulate (in some sense) another problem $\mathcal{B}$, then $\mathcal{A}$ is at least as hard as $\mathcal{B}$. This simple idea is widely used in computational complexity; for instance, NP-completeness is often shown by a gadget reduction of a known NP-complete problem to the given one. A crucial fact for the algebraic theory of the CSP is that a so called primitive positive (pp-, for short) interpretation between constraint languages gives such a reduction between corresponding CSPs (more precisely, if $\mathcal{D}$ pp-interprets $\mathcal{E}$, then $\mathrm{CSP}(\mathcal{E})$ is reducible to $\mathrm{CSP}(\mathcal{D})$). Pp-interpretations have been, indirectly, the main subject of universal algebra for the last 80 years!

The algebraic theory of CSPs was developed in a number of papers including [39, 38, 16, 43]. The viewpoint taken here is close to [12]. All results in this section come from these sources unless stated otherwise.

To simplify formulations, all structures (relational or algebraic) are assumed to have finite domains, all constraint languages are assumed to contain *finitely many*

relations, all of them *nonempty*. By a *reduction* we mean a logarithmic space reduction (although first-order reductions are often possible under additional weak assumptions).

3.1. Primitive positive interpretations. An important special case of pp-interpretability is pp-definability.

DEFINITION 3.1. *Let $\mathcal{D}, \mathcal{E}$ be constraint languages on the same domain $D = E$. We say that $\mathcal{D}$* pp-defines *$\mathcal{E}$ (or $\mathcal{E}$ is pp-definable from $\mathcal{D}$) if each relation in $\mathcal{E}$ can be defined by a first order formula which only uses relations in $\mathcal{D}$, the equality relation, conjunction and existential quantification.*

THEOREM 3.2. *If $\mathcal{D}$ pp-defines $\mathcal{E}$, then* $\mathrm{CSP}(\mathcal{E})$ *is reducible to* $\mathrm{CSP}(\mathcal{D})$.

PROOF BY EXAMPLE. Let R be an arbitrary ternary relation on a domain D. Consider the relations on D defined by

$$S(x,y) \text{ iff } (\exists z)R(x,y,z) \wedge R(y,y,x), \quad T(x,y) \text{ iff } R(x,x,x) \wedge (x=y) \ ,$$

where the existential quantification is understood over D. The relations S and T are defined by pp-formulae, therefore the constraint language $\mathcal{D} = \{R\}$ pp-defines the constraint language $\mathcal{E} = \{S, T\}$.

We sketch the reduction of $\mathrm{CSP}(\mathcal{E})$ to $\mathrm{CSP}(\mathcal{D})$ using the instance

$$S(x_3, x_2),\ T(x_1, x_4),\ S(x_2, x_4) \ .$$

We first replace S and T with their pp-definitions by introducing a new variable for each quantified variable:

$$R(x_3, x_2, y_1),\ R(x_2, x_2, x_3), \quad R(x_1, x_1, x_1),\ x_1 = x_4, \quad R(x_2, x_4, y_2),\ R(x_4, x_4, x_2)$$

and then we get rid of the equality constraint $x_1 = x_4$ by identifying these variables. This way we obtain an instance of $\mathrm{CSP}(\mathcal{D})$:

$$R(x_3, x_2, y_1),\ R(x_2, x_2, x_3),\ R(x_1, x_1, x_1),\ R(x_2, x_1, y_2),\ R(x_1, x_1, x_2) \ .$$

Clearly, the new instance of $\mathrm{CSP}(\mathcal{D})$ has a solution if and only if the original instance does. ⊣

This simple theorem provides a quite powerful tool for comparing CSPs over different languages *on the same domain*. A more powerful tool, which can also be used to compare languages with different domains, is pp-interpretability. Informally, a constraint language $\mathcal{D}$ pp-interprets $\mathcal{E}$, if the domain of $\mathcal{E}$ is a pp-definable relation (from $\mathcal{D}$) modulo a pp-definable equivalence, and the relations of $\mathcal{E}$ (viewed, in a natural way, as relations on D) are also pp-definable from $\mathcal{D}$.[5] Formally:

DEFINITION 3.3. *Let $\mathcal{D}, \mathcal{E}$ be constraint languages. We say that $\mathcal{D}$* pp-interprets *$\mathcal{E}$ if there exists a natural number n, $F \subseteq D^n$, and an onto mapping $f : F \to E$ such that $\mathcal{D}$ pp-defines*

- *the relation F,*
- *the f-preimage of the equality relation on E, and*
- *the f-preimage of every relation in $\mathcal{E}$,*

[5]This is the classical notion of interpretation from model theory restricted to pp-formulas.

where by the f-preimage of a k-ary relation S on E we mean the nk-ary relation $f^{-1}(S)$ on D defined by

$$f^{-1}(S)(x_{11},\dots,x_{1k},x_{21},\dots,x_{2k},\dots,x_{n1},\dots,x_{nk}) \text{ iff } S(f(x_{11},\dots,x_{n1}),\dots,f(x_{1k},\dots,x_{nk}))$$

THEOREM 3.4. *If $\mathcal{D}$ pp-interprets $\mathcal{E}$, then* CSP($\mathcal{E}$) *is reducible to* CSP($\mathcal{D}$).

PROOF SKETCH. The properties of the mapping f from Definition 3.3 allow us to rewrite an instance of CSP($\mathcal{E}$) to an instance of the CSP over a constraint language which is pp-definable from $\mathcal{D}$. Then we apply Theorem 3.2. ⊣

Pp-interpretability is a reflexive and transitive relation on the class of constraint languages. By identifying equivalent languages, i.e. languages which mutually pp-interpret each other, we get a partially ordered set, the *pp-interpretability poset*. Theorem 3.4 then says that the "higher" we are in the poset the "easier" the CSP we are dealing with. 3-SAT is terribly hard – we will see later that its constraint language is the least element of this poset. Surprisingly, this is "almost" the case for all known NP-complete CSPs! For a precise formulation of this we will need the reduction described in the following subsection.

3.2. Cores and singleton expansions. Let $\mathcal{D}$ be a constraint language on a finite set D. A mapping $f : D \to D$ is called an *endomorphism* if it preserves every relation $\mathcal{D}$, that is, $f(R) := \{f(\mathbf{a}) : \mathbf{a} \in R\} \subseteq R$ for every $R \in \mathcal{D}$.

THEOREM 3.5. *Let $\mathcal{D}$ be a constraint language and f an endomorphism of $\mathcal{D}$. Then* CSP($\mathcal{D}$) *is reducible to* CSP($f(\mathcal{D})$) *and vice versa, where $f(\mathcal{D})$ is the constraint language with domain $f(D)$ defined by $f(\mathcal{D}) = \{f(R) : R \in \mathcal{D}\}$.*

PROOF SKETCH. An instance of the CSP($\mathcal{D}$) has a solution if and only if the corresponding instance of CSP($f(\mathcal{D})$), obtained by replacing each $R \in \mathcal{D}$ with $f(R)$, has a solution. ⊣

A language $\mathcal{D}$ is a *core* if every endomorphism of $\mathcal{D}$ is a bijection. It is not hard to show that if f is an endomorphism of a constraint language $\mathcal{D}$ with minimal range, then $f(\mathcal{D})$ is a core. Moreover, this core is unique up to isomorphism, therefore we speak about *the core* of $\mathcal{D}$.

An important fact is that we can add all singleton unary relations to a core constraint language without increasing the complexity of its CSP:

THEOREM 3.6. *Let $\mathcal{D}$ be a core constraint language and $\mathcal{E} = \mathcal{D} \cup \bigcup_{a\in D} C_a$, where C_a denotes the unary relation $C_a = \{a\}$. Then* CSP($\mathcal{E}$) *is reducible to* CSP($\mathcal{D}$).

PROOF IDEA. The crucial step is to observe that the set of endomorphisms of $\mathcal{D}$, viewed as a $|D|$-ary relation, is pp-definable from $\mathcal{D}$. More precisely, the relation

$$S = \{(f(a_1),\dots,f(a_n)) : f \text{ is an endomorphism of } \mathcal{D}\} \ ,$$

where $a_1,\dots,a_n$ is a list of all elements of D, is pp-definable from $\mathcal{D}$ (even without existential quantification). Indeed, f is, by definition, an endomorphism of $\mathcal{D}$ if for every $R \in \mathcal{D}$ of arity $\mathrm{ar}(R)$ and every $(b_1,\dots,b_{\mathrm{ar}(R)}) \in R$ we have $(f(b_1),\dots,f(b_{\mathrm{ar}(R)})) \in R$. This directly leads to a pp-definition of S:

$$S(x_{a_1},\dots,x_{a_n}) \text{ iff } \bigwedge_{R\in\mathcal{D}} \bigwedge_{(b_1,\dots,b_{\mathrm{ar}(R)})\in R} R(x_{b_1},\dots,x_{b_{\mathrm{ar}(R)}}) \ .$$

Given an instance of CSP($\mathcal{E}$) we introduce new variables $x_{a_1}, \dots, x_{a_n}$, replace every constraint of the form $C_a(x)$ by $x = x_a$, and add the constraint $S(x_{a_1}, \dots, x_{a_n})$. In this way we obtain an instance of CSP($\mathcal{D} \cup \{=\}$). Clearly, if the original instance has a solution, then the new instance has a solution as well. In the other direction, if g is a solution to the new instance, then its values on $x_{a_1}, \dots, x_{a_n}$ determine an endomorphism f of $\mathcal{D}$. As $\mathcal{D}$ is a core, f is a bijection, thus f^{-1} is an endomorphism as well, and $f^{-1} \circ g$ restricted to the original variables is a solution of the original instance. ⊣

We will call constraint languages containing all singletons *idempotent*. Note that an idempotent constraint language is automatically a core as the only endomorphism is the identity. By Theorems 3.5, 3.6, CSP over $\mathcal{D}$ is reducible to CSP over the singleton expansion of the core of $\mathcal{D}$ and vice versa. It is therefore enough to study CSPs over idempotent constraint languages.

An interesting consequence of these reductions is that the search problem for CSP($\mathcal{D}$) is solvable in polynomial time whenever CSP($\mathcal{D}$) is. The idea is to gradually guess values for variables using the unary singleton constraints.

3.3. Example.

EXAMPLE 3.7. *We show that* 3-SAT *is reducible to* 3-COLORING.

Recall the constraint language $\mathcal{D}_{3COLOR} = \{\neq_{\{0,1,2\}}\}$ *of* 3-COLORING *from Example 2.5 and the constraint language* $\mathcal{D}_{3SAT} = \{S_{000}, \dots, S_{111}\}$ *of* 3-SAT *from Example 2.3.*

Since $\mathcal{D}_{3COLOR}$ *is a core,* CSP($\mathcal{D}'_{3COLOR}$)*, where* $\mathcal{D}'_{3COLOR} = \{\neq, C_0, C_1, C_2\}$*, is reducible to* CSP($\mathcal{D}_{3COLOR}$) *by Theorem 3.6. By Theorem 3.4, it is now enough to show that* $\mathcal{D}'_{3COLOR}$ *pp-interprets* $\mathcal{D}_{3SAT}$*. We give a pp-interpretation with* $n = 1$*,* $F = \{0, 1\}$*, and* f *the identity map (see Definition 3.3). The set (=unary relation)* $\{0, 1\}$ *can be pp-defined by*

$$E(x) \quad \textit{iff} \quad (\exists y)\ C_2(y) \wedge x \neq y \quad (\textit{iff } x \neq 2)\ .$$

The preimage of the equality relation is the equality relation on $\{0, 1\}$ *which is clearly pp-definable. The relation* S_{000} *can be defined by*

$$S_{000}(x_1, x_2, x_3) \quad \textit{iff} \quad (\exists y_1, y_2, y_3, z)\ C_2(z) \wedge y_1 \neq y_2 \wedge y_2 \neq y_3 \wedge y_1 \neq y_3 \wedge \bigwedge_{i=1,2,3} z \neq x_i \wedge T(x_i, y_i)\ ,$$

where T *is the binary relation*

$$T(x, y) \quad \textit{iff} \quad (\exists u, v)\ C_1(u) \wedge u \neq v \wedge x \neq v \wedge y \neq v$$

The other relations S_{ijk} *are defined similarly.*

While it is easy to verify that the presented pp-definitions work, it is not so easy to find them without any tools. The proof of Theorem 3.9 gives an algorithm to produce pp-definitions whenever they exist (although the obtained definitions will usually be very long).

3.4. Tractability conjecture. Now we return to the pp-interpretability poset. Recall that "higher" in the poset means "easier" CSP and that 3-SAT corresponds to the least (the hardest) element. When we restrict to idempotent constraint languages (which we can do by the previous discussion), all known

NP-complete CSPs are at the bottom of the poset. Bulatov, Jeavons and Krokhin conjectured that this is not a coincidence.

CONJECTURE 3.8 (Tractability conjecture). *If an idempotent constraint language $\mathcal{D}$ does not pp-interpret the language of* 3-SAT*, then* CSP($\mathcal{D}$) *is solvable in polynomial time.*

This conjecture is also known as the *algebraic dichotomy conjecture* because many equivalent formulations, including the original one, are algebraic.

Similar hardness results and conjectures have been formulated for other computational/descriptive complexity classes.

3.5. Algebraic counterpart of pp-definability. The link between relations and operations is provided by a natural notion of compatibility. An n-ary operation f on a finite set D (that is, a mapping $f : D^n \to D$) is *compatible* with a k-ary relation $R \subseteq D^k$ if f applied component-wise to any n-tuple of elements of R gives an element of R. In more detail, whenever (a_{ij}) is an $n \times k$ matrix such that every row is in R, then f applied to the columns gives a k-tuple which is in R as well.

We say that an operation f on D is a *polymorphism* of a constraint language $\mathcal{D}$ if f is compatible with every relation in $\mathcal{D}$. Note that a unary polymorphism is the same as an endomorphism. If endomorphisms can be thought of as symmetries, then polymorphisms can be viewed as symmetries of higher arities.

The set of all polymorphisms of $\mathcal{D}$ will be denoted by $\mathbf{D}$. This algebraic object has the following two properties.

- $\mathbf{D}$ *contains all projections*, that is, for every natural number n and $i \leq n$ the n-ary projection onto the i-th coordinate, defined by
$$\pi_i^n(a_1, \ldots, a_n) = a_i,$$
is in $\mathbf{D}$.
- $\mathbf{D}$ *is closed under composition*, that is, for any n-ary $g \in \mathbf{D}$ and k-ary $f_1, \ldots, f_n \in \mathbf{D}$ their (k-ary) composition $g(f_1, \ldots, f_n)$, defined by
$$g(f_1, \ldots, f_n)(a_1, \ldots, a_k) = g(f_1(a_1, \ldots, a_k), \ldots, f_n(a_1, \ldots, a_k)),$$
is in $\mathbf{D}$.

Sets of operations with these properties are called *concrete clones* (or *function clones*, or simply *clones*), therefore we refer to $\mathbf{D}$ as the *clone of polymorphisms* of $\mathcal{D}$.

The clone of polymorphisms controls pp-definability in the sense of the following old result [32, 13].

THEOREM 3.9. *Let $\mathcal{D}$, $\mathcal{E}$ be constraint languages with $D = E$. Then $\mathcal{D}$ pp-defines $\mathcal{E}$ if and only if $\mathbf{D} \subseteq \mathbf{E}$.*[6]

PROOF SKETCH. The implication "⇒" is quite easy. For the other implication it is enough to prove that whenever R is a relation compatible with every polymorphism of $\mathcal{D}$, then R is pp-definable from $\mathcal{D}$. A crucial step is a more general version of the observation made in the proof of Theorem 3.6: For any k,

[6]Moreover, every concrete clone is the clone of polymorphisms of some (possibly infinite) constraint language.

the set of k-ary polymorphisms of $\mathcal{D}$ can be viewed as a $|D|^k$-ary relation S on D, and this relation is pp-definable from $\mathcal{D}$. Now R can be defined from such a relation S (where k is the number of tuples in R) by existential quantification over suitable coordinates as in Example 3.11. ⊣

In view of this result, Theorem 3.2 says that the complexity of CSP($\mathcal{D}$) only depends on the clone $\mathbf{D}$. More precisely, if $\mathbf{D} \subseteq \mathbf{E}$, then CSP($\mathcal{E}$) is reducible to CSP($\mathcal{D}$). Moreover, the proof of Theorem 3.9 gives a generic pp-definition of $\mathcal{E}$ from $\mathcal{D}$, which gives us a generic reduction of CSP($\mathcal{E}$) to CSP($\mathcal{D}$).

EXAMPLE 3.10. *It is a nice exercise to show that the language $\mathcal{D}_{3SAT}$ of* 3-SAT *has no polymorphisms except for the projections. This means that $\mathcal{D}_{3SAT}$ pp-defines every constraint language with domain $\{0,1\}$. It follows (see also Theorem 3.14) that $\mathcal{D}_{3SAT}$ pp-interprets every constraint language, so it is the least element of the pp-interpretability poset, as claimed earlier.*

EXAMPLE 3.11. *Another nice exercise is to show that the language $\mathcal{D}'_{3COLOR} = \{\neq, C_0, C_1, C_2\}$ on the domain $\{0,1,2\}$ (see Example 3.7) also does not have any polymorphisms except for projections.*

We show how the proof of Theorem 3.9 produces a pp-definition of the relation

$$R = \{(0,1),(0,2),(1,1),(2,2)\} \ .$$

Since R contains 4 pairs, we pp-define the 3^4-ary relation

$$S = \{(f(0,0,0,0), f(0,0,0,1), \dots, f(2,2,2,2)) : f \text{ is a 4-ary polymorphism of } \mathcal{D}'_{3COLOR}\}.$$

which corresponds to the set of all 4-ary polymorphisms of $\mathcal{D}'_{3COLOR}$:

$$S(x_{0000}, \dots, x_{2222}) \text{ iff } \bigwedge_i x_{iiii} = i \wedge \bigwedge_{i_1 \neq i_2, j_1 \neq j_2, k_1 \neq k_2, l_1 \neq l_2} x_{i_1 j_1 k_1 l_1} \neq x_{i_2 j_2 k_2 l_2} \ .$$

Now we existentially quantify over all variables but x_{0012} and x_{1212} – the exceptions are those variables which correspond to the i-th coordinates of pairs in R, $i \in \{1,2\}$. The obtained binary relation $R'(x_{0012}, x_{1212})$ contains R since S contains the projections, and is contained in R since R is compatible with every polymorphism of $\mathcal{D}'_{3COLOR}$.

Note that the definition of S_{000} from Example 3.7 obtained in this way contains 3^7 variables. This is the price we need to pay for genericity.

3.6. Algebraic counterpart of pp-interpretability. For the algebraic description of pp-interpretability we introduce three constructions which are clone versions of standard constructions for groups, rings, etc.

Let $\mathbf{D}$ be a (concrete) clone.

The domain D of $\mathbf{D}$ is also called the *universe* of $\mathbf{D}$. We say that $E \subseteq D$ is a *subuniverse* of $\mathbf{D}$ if it is closed under all operations of $\mathbf{D}$. In this situation, we can form a clone $\mathbf{E}$ by restricting all operations of $\mathbf{D}$ to the set E. The clone $\mathbf{E}$ is called a *subalgebra* of $\mathbf{D}$ (the word *subclone* is reserved for set theoretic inclusion).

For a natural number n we can form the *n-th power* $\mathbf{D}^n$ of $\mathbf{D}$ with domain D^n and operations from $\mathbf{D}$ acting coordinate-wise. (More generally, we can form the X-th power $\mathbf{D}^X$ of $\mathbf{D}$ for any set X.) A *subpower* is a subuniverse (or a

subalgebra, depending on the context) of a power. Note that if $\mathbf{D}$ is the clone of polymorphisms of a constraint language $\mathcal{D}$, then R is a subpower of $\mathbf{D}$ if and only if R is pp-definable from $\mathcal{D}$ (by Theorem 3.9).

Finally, let $\phi : D \to E$ be an onto mapping such that for any operation $f \in \mathbf{D}$ (say of arity n), the formula

$$f_\phi(\phi(a_1), \ldots, \phi(a_n)) = \phi(f(a_1, \ldots, a_n)) \quad \forall a_1, \ldots, a_n \in D$$

correctly defines an operation f_ϕ on E. Then $\mathbf{E} = \{f_\phi : f \in \mathbf{D}\}$ is a clone with domain E called a *concrete homomorphic image* of $\mathbf{D}$ and ϕ is called a *concrete homomorphism*.

The definition of pp-interpretability can be translated into algebraic terms as follows.

THEOREM 3.12. *Let $\mathcal{D}$, $\mathcal{E}$ be constraint languages. Then $\mathcal{D}$ pp-interprets $\mathcal{E}$ if and only if $\mathbf{E}$ contains a concrete homomorphic image of a subpower of $\mathbf{D}$.*

3.7. Identities and Mal'tsev conditions. An alternative algebraic characterization of pp-interpretability, which is missing on the relational side, follows from the foundation stone of universal algebra, the Birkhoff HSP theorem [11]: pp-interpretability depends on the identities (i.e. universally quantified equations) satisfied by polymorphisms.

We first present a formulation using abstract clone homomorphisms and then explain the connection to identities.

DEFINITION 3.13. *A mapping H from a clone $\mathbf{D}$ to a clone $\mathbf{E}$ is called a* clone homomorphism *if*

- *it preserves the arities of operations,*
- *it maps projections to projections (that is, $H(\pi_i^n) = \pi_i^n$, where the projection on the left hand side works on the set D, while on the right hand side on the set E), and*
- *it preserves the composition (that is,*

  $$H(g(f_1, \ldots, f_n)) = H(g)(H(f_1), \ldots, H(f_n))$$

  if $g, f_1, \ldots, f_n$ are from $\mathbf{D}$ and have appropriate arities).

THEOREM 3.14. *Let $\mathcal{D}$, $\mathcal{E}$ be constraint languages. Then $\mathcal{D}$ pp-interprets $\mathcal{E}$ if and only if there exists an abstract clone homomorphism from $\mathbf{D}$ to $\mathbf{E}$.*

PROOF SKETCH. There are natural abstract clone homomorphisms associated to the three constructions on clones (taking sublagebras, powers and concrete homomorphic images). The implication $\Rightarrow$ follows from this observation and Theorem 3.12.

Now assume that $H : \mathbf{D} \to \mathbf{E}$ is a clone homomorphism. For simplicity, let $E = \{1, 2, \ldots, n\}$. It is easy to check that the set F of all n-ary operations in $\mathbf{D}$ is a subuniverse of $\mathbf{D}^{D^n}$. Let $\mathbf{F}$ be the corresponding subalgebra of $\mathbf{D}^{D^n}$. (This important object, the *n-generated free algebra for $\mathbf{D}$*, already appeared in the proof of Theorem 3.9. Indeed, if $\mathbf{D}$ is the clone of polymorphisms of a constraint language $\mathcal{D}$, then F is the set of all n-ary polymorphisms of $\mathcal{D}$.) A simple calculation shows that the mapping $\phi : F \to E$, defined by $\phi(f) =$

$(H(f))(1,2,\dots,n)$, is a concrete clone homomorphism from $\mathbf{F}$ onto $H(\mathbf{D}) \subseteq \mathbf{E}$ and thus $\mathcal{D}$ pp-interprets $\mathcal{E}$ by Theorem 3.12. ⊣

Observe that the existence of an abstract clone homomorphism $H : \mathbf{D} \to \mathbf{E}$ does not depend on the concrete operations in $\mathbf{D}$ and $\mathbf{E}$ – it only depends on the way in which operations compose and which operations are projections. The skeleton of a concrete clone, which only remembers projections and compositions, is called an *abstract clone*.[7]

We now explain the promised link to identities, first with an example. A binary operation f on D is a *semilattice* operation if satisfies the identities

$$f(f(x,y),z) \approx f(x,f(y,z)),\ f(x,y) \approx f(y,x),\ \text{and } f(x,x) \approx x,$$

meaning that $f(f(a,b),c) = f(a,f(b,c))$, $f(a,b) = f(b,a)$, and $f(a,a) = a$ hold for any $a,b,c \in D$. This can be expressed in terms of composition and projections: f is a semilattice operation if and only if

$$f(f(\pi^3_1,\pi^3_2),\pi^3_3) = f(\pi^3_1,f(\pi^3_2,\pi^3_3)),\ f(\pi^2_1,\pi^2_2) = f(\pi^2_2,\pi^2_1),\ \text{and } f(\pi^1_1,\pi^1_1) = \pi^1_1\ .$$

It follows that if $H : \mathbf{D} \to \mathbf{E}$ is an abstract clone homomorphism and $\mathbf{D}$ contains a semilattice operation f, then $\mathbf{E}$ contains a semilattice operation as well, namely $H(f)$.

More generally, if there exists an abstract clone homomorphism from $\mathbf{D}$ to $\mathbf{E}$, then $\mathbf{E}$ satisfies all properties of the form "there exist operations ... satisfying identities ... " which are satisfied by $\mathbf{D}$. We will call such properties *Mal'tsev conditions* (although we deviate from the standard definition).

It is not hard to see that the converse is also true: if no abstract clone homomorphism $\mathbf{D} \to \mathbf{E}$ exists, then there is some Mal'tsev condition which is satisfied by $\mathbf{D}$ while not satisfied by $\mathbf{E}$. In short:

> The complexity of CSP($\mathcal{D}$) only depends on the Mal'tsev conditions satisfied by the clone of polymorphisms of $\mathcal{D}$.

To illustrate this, we state one of increasingly many (e.g., [54, 35, 46, 41, 52]) characterizations of the conjectured borderline between P and NP-complete CSPs by means of cyclic operations [6].

THEOREM 3.15. *Let $\mathcal{D}$ be an idempotent constraint language and $p > |D|$ a prime. Then the following are equivalent.*

- *$\mathcal{D}$ does not interpret the language of* 3-SAT.
- *$\mathbf{D}$ contains an operation t (equivalently, $\mathcal{D}$ has a polymorphism t) of arity p such that*
$$t(x_1,\dots,x_p) \approx t(x_2,\dots,x_p,x_1)\ .$$

Even if the tractability conjecture or the dichotomy conjecture (or finer classification conjectures) turns out to be incorrect, we *know* that classes of CSPs in P, L, NL, ... *can be* characterized by Mal'tsev conditions on polymorphisms.

EXAMPLE 3.16. *We show how to apply cyclic operations to prove the dichotomy theorem for undirected graphs [34].*

[7]The relation between abstract clones and concrete clones is similar to the relation between groups and permutation groups, or between monoids and transformation monoids.

Let R be a symmetric binary relation viewed as an undirected graph and $\mathcal{D} = \{R\}$. Let $\mathcal{D}' = \{R', ...\}$ be the singleton expansion of the core of $\mathcal{D}$. If R contains a loop then CSP($\mathcal{D}$) *is trivially tractable. If R is bipartite, then the core of R is an edge and* CSP($\mathcal{D}$) *is essentially* 2-COLORING*, which is tractable.*

Finally, if R is not bipartite and does not contain a loop, then R' does not contain a loop and does contain a closed walk $a_1, a_2, \dots, a_p, a_1$ for some prime $p > |D'|$. Assume that $\mathbf{D}'$ contains a cyclic operation t of arity p. Since t is a polymorphism, the pair

$$t((a_1, a_2), \dots, (a_{p-1}, a_p), (a_p, a_1)) = (t(a_1, \dots, a_p), t(a_2, \dots, a_p, a_1))$$

is in R', but it is a loop since t is cyclic. This contradiction shows that $\mathbf{D}'$ does not contain a cyclic operation of arity p, therefore CSP($\mathcal{D}'$) *(and thus* CSP($\mathcal{D}$)*) is NP-complete.*

§4. Results. Universal algebra serves the investigation in two ways: as a toolbox containing heavy hammers (such as the Tame Congruence Theory by Hobby and McKenzie [35]) and as a guideline for identifying interesting intermediate cases, which are hard to spot from the purely relational perspective. Major results include the following.

- The dichotomy theorem of Schaefer for CSPs over a two-element domain was generalized to a three-element domain by Bulatov [19]. A simplification of this result and a generalization to four-element domains was announced by Marković et al.
- The dichotomy theorem of Hell and Nešetřil for CSPs over undirected graphs was generalized to digraphs with no sources or sinks [9].
- The dichotomy conjecture was proved for all constraint languages containing all unary relations by Bulatov [18] (a simpler proof is in [2]).

Notably, all known tractable cases are solvable by a combination of two basic algorithms, or rather algorithmic principles – local consistency, and the "few subpowers" algorithm. It is another significant success of the algebraic approach that the applicability of these principles is now understood.

4.1. Local consistency. The CSP over some constraint languages can be decided in polynomial time by constraint propagation algorithms, or, in other words, by enforcing local consistency. Such CSPs are said to have *bounded width.*

This notion comes in various versions and equivalent forms. We refer to [31] for formalizations using Datalog programs and games, to [23] for a description using dualities, and to [21, 4] for a notion suitable for infinite languages.

We informally sketch one possible definition. Let $k \leq l$ be positive integers. The (k, l)*-algorithm* derives the strongest possible constraints on k variables by considering l variables at a time. If a contradiction is found, the algorithm answers "no (solution)", otherwise it answers "yes". These algorithms work in polynomial time (for fixed k, l) and "no" answers are always correct. A constraint language $\mathcal{D}$ (or CSP($\mathcal{D}$)) has *width* (k, l), if "yes" answers are correct for every instance of CSP($\mathcal{D}$). If $\mathcal{D}$ has width (k, l) for some k, l, we say that $\mathcal{D}$ has *bounded width.*

As an example, we consider the constraint language $\mathcal{D}_{2\text{COLOR}}$ and the instance

$$x_1 \neq x_2,\ x_2 \neq x_3,\ x_3 \neq x_4,\ x_4 \neq x_5,\ x_5 \neq x_1\ .$$

The $(2,3)$-algorithm can certify that this instance has no solution as follows:

- We consider the variables x_1, x_2, x_3. Using $x_1 \neq x_2$, $x_2 \neq x_3$ we derive $x_1 = x_3$.
- We consider x_1, x_3, x_4. Using $x_3 \neq x_4$ and the already derived constraint $x_1 = x_3$ we derive $x_1 \neq x_4$.
- We consider x_1, x_4, x_5 and using $x_1 \neq x_4$, $x_4 \neq x_5$ and $x_5 \neq x_1$ we derive a contradiction.

In fact, 2-COLORING has width $(2,3)$, that is, such reasoning finds a contradiction for every unsatisfiable instance. Other examples of bounded width problems include HORN-3-SAT and 2-SAT.

Feder and Vardi [31] proved that problems 3-LIN(p) (and more generally, similar problems 3-LIN($\mathbf{M}$) over finite modules) do not have bounded width and conjectured that linear equations are essentially the only obstacles for having bounded width. An algebraic formulation of this was given by Larose and Zádori [44]. They proved that analogues of results in section 3 hold for bounded width, therefore no problem which pp-interprets the language of 3-LIN($\mathbf{M}$) has bounded width, and conjectured that the converse is also true. After a sequence of partial results [40, 26, 5, 20], the conjecture was eventually confirmed in [8][8] and independently in [14].

THEOREM 4.1. *An idempotent constraint language $\mathcal{D}$ has bounded width if and only if $\mathcal{D}$ does not interpret the language of* 3-LIN($\mathbf{M}$) for a finite module $\mathbf{M}$.[9]

4.2. Few subpowers. Gaussian elimination not only solves 3-LIN(p), it also describes all the solutions in the sense that the algorithm can output a small (polynomially large) set of points in $\mathrm{GF}(p)^n$ so that the affine hull of these points is equal to the solution set of the original instance. A sequence of papers [31, 17, 15, 29] culminating in [36, 10] pushed this idea, in a way, to its limit.

We need some terminology to state the result. Let $\mathcal{D}$ be a constraint language and $\mathbf{D}$ its clone of polymorphisms. Recall that a relation on D is a *subpower* of $\mathbf{D}$ if and only if it is pp-definable from $\mathcal{D}$. Note that the set of solutions of any instance of CSP($\mathcal{D}$) can be viewed as a subpower of $\mathbf{D}$. Now $\mathbf{D}$ has *few subpowers* if each subpower can be obtained as a closure under polymorphisms of a small set (polynomially large with respect to the arity).[10]

THEOREM 4.2. *Let $\mathcal{D}$ be an idempotent constraint language. If $\mathbf{D}$ has few subpowers, then* CSP($\mathcal{D}$) *can be solved in polynomial time.*

---

[8] A modification required to handle infinite languages was given in [4].

[9] Moreover, if $\mathcal{D}$ has bounded width, then it has width $(2,3)$ with an appropriate notion of width. Also, the property of having bounded width can be checked in polynomial time given an idempotent $\mathcal{D}$ on input.

[10] The name comes from an equivalent property that $\mathbf{D}$ has only exponentially many subpowers.

§5. Conclusion. We have seen that the complexity of the satisfiability problem for CSP over a fixed constraint language depends on "higher arity symmetries" – polymorphisms of the language. (We have only discussed languages with finite domains. The algebraic theory extends to interesting subclasses of infinite domain CSP [12]). Significant progress has been achieved using this insight, but the main problem, the dichotomy conjecture, is still open.

A similar approach can be applied to other variants of CSP over a fixed constraint language. In two of them, the main goal has been reached: the dichotomy for the counting problem was proved in [22] (substantially simplified in [30]) and for the robust satisfiability problem in [7]. A generalization of the theory for the optimization problem and valued CSPs was given in [28], and some links to universal algebra are emerging from research in the area of approximation algorithms (such as [48]).

Is this approach only applicable to CSPs over fixed languages? Or are we merely seeing a piece of a bigger theory?

Acknowledgement. I thank Matt Valeriote for carefully reading the paper and correcting mistakes.

REFERENCES

[1] Sanjeev Arora and Boaz Barak, ***Computational complexity: A modern approach***, 1st ed., Cambridge University Press, New York, NY, USA, 2009.

[2] Libor Barto, *The dichotomy for conservative constraint satisfaction problems revisited*, ***26th Annual IEEE Symposium on Logic in Computer Science—LICS 2011***, IEEE Computer Soc., Los Alamitos, CA, 2011, pp. 301–310.

[3] ———, *Constraint satisfaction problem and universal algebra*, ***ACM SIGLOG News***, vol. 1 (2014), no. 2, pp. 14–24.

[4] ———, *The collapse of the bounded width hierarchy*, ***Journal of Logic and Computation***, (published online 2014).

[5] Libor Barto and Marcin Kozik, *Congruence distributivity implies bounded width*, ***SIAM Journal on Computing***, vol. 39 (2009), no. 4, pp. 1531–1542.

[6] Libor Barto and Marcin Kozik, *Absorbing subalgebras, cyclic terms, and the constraint satisfaction problem*, ***Logical Methods in Computer Science***, vol. 8 (2012), no. 1.

[7] ———, *Robust satisfiability of constraint satisfaction problems*, ***Proceedings of the 44th symposium on theory of computing*** (New York, NY, USA), STOC '12, ACM, 2012, pp. 931–940.

[8] ———, *Constraint satisfaction problems solvable by local consistency methods*, ***J. ACM***, vol. 61 (2014), no. 1, pp. 3:1–3:19.

[9] Libor Barto, Marcin Kozik, and Todd Niven, *The CSP dichotomy holds for digraphs with no sources and no sinks (a positive answer to a conjecture of Bang-Jensen and Hell)*, ***SIAM J. Comput.***, vol. 38 (2008/09), no. 5, pp. 1782–1802.

[10] Joel Berman, Pawel Idziak, Petar Marković, Ralph McKenzie, Matthew Valeriote, and Ross Willard, *Varieties with few subalgebras of powers*, ***Transactions of The American Mathematical Society***, vol. 362 (2009), pp. 1445–1473.

[11] Garrett Birkhoff, *On the structure of abstract algebras*, ***Proc. Camb. Philos. Soc.***, vol. 31 (1935), pp. 433–454.

[12] Manuel Bodirsky, *Constraint satisfaction problems with infinite templates*, ***Complexity of constraints*** (Nadia Creignou, Phokion G. Kolaitis, and Heribert Vollmer, editors), Lecture Notes in Computer Science, vol. 5250, Springer, 2008, pp. 196–228.

[13] V. G. Bodnarčuk, L. A. Kalužnin, V. N. Kotov, and B. A. Romov, *Galois theory for Post algebras. I, II*, ***Kibernetika (Kiev)***, (1969), no. 3, pp. 1–10; ibid. 1969, no. 5, 1–9.

[14] Andrei Bulatov, *Bounded relational width*, manuscript, 2009.

[15] Andrei Bulatov and Víctor Dalmau, *A simple algorithm for Mal′tsev constraints*, ***SIAM J. Comput.***, vol. 36 (2006), no. 1, pp. 16–27 (electronic).

[16] Andrei Bulatov, Peter Jeavons, and Andrei Krokhin, *Classifying the complexity of constraints using finite algebras*, ***SIAM J. Comput.***, vol. 34 (2005), pp. 720–742.

[17] Andrei A. Bulatov, *Mal'tsev constraints are tractable*, ***Electronic Colloquium on Computational Complexity (ECCC)***, (2002), no. 034.

[18] ———, *Tractable conservative constraint satisfaction problems*, ***Proceedings of the 18th annual ieee symposium on logic in computer science*** (Washington, DC, USA), IEEE Computer Society, 2003, pp. 321–.

[19] Andrei A. Bulatov, *A graph of a relational structure and constraint satisfaction problems*, ***Proceedings of the nineteenth annual ieee symposium on logic in computer science (lics 2004)***, IEEE Computer Society Press, July 2004, pp. 448–457.

[20] Andrei A. Bulatov, *Combinatorial problems raised from 2-semilattices*, ***J. Algebra***, vol. 298 (2006), no. 2, pp. 321–339.

[21] ———, *Complexity of conservative constraint satisfaction problems*, ***ACM Trans. Comput. Logic***, vol. 12 (2011), no. 4, pp. 24:1–24:66.

[22] ———, *The complexity of the counting constraint satisfaction problem*, ***J. ACM***, vol. 60 (2013), no. 5, pp. 34:1–34:41.

[23] Andrei A. Bulatov, Andrei Krokhin, and Benoit Larose, *Complexity of constraints*, (Nadia Creignou, Phokion G. Kolaitis, and Heribert Vollmer, editors), Springer-Verlag, Berlin, Heidelberg, 2008, pp. 93–124.

[24] Gerhard Buntrock, Carsten Damm, Ulrich Hertrampf, and Christoph Meinel, *Structure and importance of logspace-mod class*, ***Mathematical systems theory***, vol. 25 (1992), no. 3, pp. 223–237 (English).

[25] Jin-Yi Cai and Xi Chen, *Complexity of counting csp with complex weights*, ***Proceedings of the forty-fourth annual acm symposium on theory of computing*** (New York, NY, USA), STOC '12, ACM, 2012, pp. 909–920.

[26] Catarina Carvalho, Víctor Dalmau, Petar Marković, and Miklós Maróti, *CD(4) has bounded width*, ***Algebra Universalis***, vol. 60 (2009), no. 3, pp. 293–307.

[27] Hubie Chen, *Meditations on quantified constraint satisfaction*, ***Logic and program semantics*** (RobertL. Constable and Alexandra Silva, editors), Lecture Notes in Computer Science, vol. 7230, Springer Berlin Heidelberg, 2012, pp. 35–49 (English).

[28] David A. Cohen, Martin C. Cooper, Páidí Creed, Peter Jeavons, and Stanislav Živný, *An algebraic theory of complexity for discrete optimisation*, ***SIAM Journal on Computing***, vol. 42 (2013), no. 5, pp. 1915–1939.

[29] Víctor Dalmau, *Generalized majority-minority operations are tractable*, ***Log. Methods Comput. Sci.***, vol. 2 (2006), no. 4, pp. 4:1, 14.

[30] Martin E. Dyer and David Richerby, *An effective dichotomy for the counting constraint satisfaction problem*, ***SIAM J. Comput.***, vol. 42 (2013), no. 3, pp. 1245–1274.

[31] Tomás Feder and Moshe Y. Vardi, *The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory*, ***SIAM Journal on Computing***, vol. 28 (1998), no. 1, pp. 57–104.

[32] David Geiger, *Closed systems of functions and predicates*, ***Pacific J. Math.***, vol. 27 (1968), pp. 95–100.

[33] Johan Håstad, *On the efficient approximability of constraint satisfaction problems*, ***Surveys in combinatorics 2007***, London Math. Soc. Lecture Note Ser., vol. 346, Cambridge Univ. Press, Cambridge, 2007, pp. 201–221.

[34] Pavol Hell and Jaroslav Nešetřil, *On the complexity of H-coloring*, ***J. Combin. Theory Ser. B***, vol. 48 (1990), no. 1, pp. 92–110.

[35] David Hobby and Ralph McKenzie, ***The structure of finite algebras***, Contemporary Mathematics, vol. 76, American Mathematical Society, Providence, RI, 1988.

[36] Pawel Idziak, Petar Marković, Ralph McKenzie, Matthew Valeriote, and Ross Willard, *Tractability and learnability arising from algebras with few subpowers*, ***Proceedings of the twenty-second annual ieee symposium on logic in computer science (lics 2007)***, IEEE Computer Society Press, July 2007, pp. 213–222.

[37] Neil Immerman, *Nondeterministic space is closed under complementation*, ***SIAM J. Comput.***, vol. 17 (1988), no. 5, pp. 935–938.

[38] Peter Jeavons, *On the algebraic structure of combinatorial problems*, ***Theoretical Computer Science***, vol. 200 (1998), no. 12, pp. 185 – 204.

[39] Peter Jeavons, David Cohen, and Marc Gyssens, *Closure properties of constraints*, ***J. ACM***, vol. 44 (1997), no. 4, pp. 527–548.

[40] Emil Kiss and Matthew Valeriote, *On tractability and congruence distributivity*, ***Log. Methods Comput. Sci.***, vol. 3 (2007), no. 2, pp. 2:6, 20 pp. (electronic).

[41] Gábor Kun and Mario Szegedy, *A new line of attack on the dichotomy conjecture*, ***Proceedings of the forty-first annual acm symposium on theory of computing*** (New York, NY, USA), STOC '09, ACM, 2009, pp. 725–734.

[42] Richard E. Ladner, *On the structure of polynomial time reducibility*, ***J. Assoc. Comput. Mach.***, vol. 22 (1975), pp. 155–171.

[43] Benoît Larose and Pascal Tesson, *Universal algebra and hardness results for constraint satisfaction problems*, ***Theor. Comput. Sci.***, vol. 410 (2009), pp. 1629–1647.

[44] Benoit Larose and László Zádori, *Bounded width problems and algebras*, ***Algebra Universalis***, vol. 56 (2007), no. 3-4, pp. 439–466.

[45] F. Madelaine and B. Martin, *A tetrachotomy for positive first-order logic without equality*, ***Logic in computer science (lics), 2011 26th annual ieee symposium on***, June 2011, pp. 311–320.

[46] Miklós Maróti and Ralph McKenzie, *Existence theorems for weakly symmetric operations*, ***Algebra Universalis***, vol. 59 (2008), no. 3-4, pp. 463–489.

[47] Christos H. Papadimitriou, ***Computational complexity***, Addison-Wesley Publishing Company, Reading, MA, 1994.

[48] Prasad Raghavendra, *Optimal algorithms and inapproximability results for every CSP?*, ***Stoc'08***, 2008, pp. 245–254.

[49] Omer Reingold, *Undirected connectivity in log-space*, ***J. ACM***, vol. 55 (2008), no. 4, pp. 17:1–17:24.

[50] Francesca Rossi, Peter van Beek, and Toby Walsh, ***Handbook of constraint programming (foundations of artificial intelligence)***, Elsevier Science Inc., New York, NY, USA, 2006.

[51] Thomas J. Schaefer, *The complexity of satisfiability problems*, ***Conference Record of the Tenth Annual ACM Symposium on Theory of Computing (San Diego, Calif., 1978)***, ACM, New York, 1978, pp. 216–226.

[52] MarkH. Siggers, *A strong malcev condition for locally finite varieties omitting the unary type*, ***Algebra universalis***, vol. 64 (2010), no. 1-2, pp. 15–20 (English).

[53] R. Szelepcsényi, *The method of forced enumeration for nondeterministic automata*, ***Acta Inf.***, vol. 26 (1988), no. 3, pp. 279–284.

[54] Walter Taylor, *Varieties obeying homotopy laws*, ***Canad. J. Math.***, vol. 29 (1977), no. 3, pp. 498–527.

[55] Stanislav Živný, ***The complexity of valued constraint satisfaction problems***, Cognitive Technologies, Springer, Heidelberg, 2012.

DEPARTMENT OF ALGEBRA
FACULTY OF MATHEMATICS AND PHYSICS
CHARLES UNIVERSITY IN PRAGUE
SOKOLOVSKÁ 83, 18675 PRAHA 8
E-mail: libor.barto@gmail.com
URL: http://www.karlin.mff.cuni.cz/~barto