Constraint Satisfaction Problem and Universal Algebra

Libor Barto,
Department of Algebra,
Faculty of Mathematics and Physics,
Charles University in Prague

This column gives a brief survey of current research on the complexity of the constraint satisfaction problem (CSP) 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 [Rossi et al. 2006]. One solves an instance of CSP by assigning values to the variables so that the constraints are satisfied.

In this column I describe some dramatic recent progress in our understanding of the computational complexity of CSP over a **fixed constraint language**. This restricted framework is still broad enough to include many NP-complete problems, yet it is narrow enough to potentially allow a complete classification of all such CSP problems.

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

My aim in this column is to introduce the basics of this exciting area and highlight selected deeper results, in a way that is understandable to readers with a basic knowledge of computational complexity (see [Papadimitriou 1994; Arora and Barak 2009]). The presentation of the material is based on my talk "Universal algebra and the constraint satisfaction problem" delivered at the Association of Symbolic Logic North American Annual Meeting held in Boulder, Colorado, in 2014. A more detailed version of this column is being prepared for the *Bulletin of Symbolic Logic*.

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 power D^n. It is sometimes convenient to work with the corresponding *predicate* which is a mapping from D^n to $\{\mathrm{true}, \mathrm{false}\}$ specifying which tuples are in R. We will use both formalism, 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 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 variables is a *solution*

[1] There are also different types of constraints considered in the literature, see e.g. Chapter 7 in [Rossi et al. 2006].

if it satisfies all the constraints, that is, in our example,

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

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

— **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 2.2). 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.

In this column, I focus just on satisfiability problems with language restrictions. Please see [Živný 2012] for optimization problems and a generalization to valued CSPs, [Håstad 2007] for approximation, [Cai and Chen 2012] for counting, and [Bodirsky 2008] for a generalization to infinite domains.

Definition 2.1. 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 2.2. An instance of the standard NP-complete problem, 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 $\mathrm{CSP}(\mathcal{D}_{\mathrm{3SAT}})$, where $D_{\mathrm{3SAT}} = \{0, 1\}$ and

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

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

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

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).

Example 2.3. 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 $\mathrm{CSP}(\mathcal{D}_{\mathrm{HornSAT}})$ for $\mathcal{D}_{\mathrm{HornSAT}} = \{S_{011}, S_{101}, S_{110}, S_{111}\}$ (or just $\mathcal{D}_{\mathrm{HornSAT}} = \{S_{011}, S_{111}\}$). HORN-3-SAT is solvable in polynomial time, in fact, it is a P-complete problem.

Example 2.4. For a fixed natural number k, the k-COLORING problem is to decide whether it is possible to assign colors $\{1, 2, \ldots, k\}$ to vertices of an input graph in such a way that adjacent vertices receive different colors. This problem is equivalent to $\mathrm{CSP}(\mathcal{D}_{\mathrm{kCOLOR}})$, where $D_k = \{1, 2, \ldots, k\}$ and $\mathcal{D}_{\mathrm{kCOLOR}} = \{\neq_k\}$ 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 $\mathrm{CSP}(\mathcal{D})$, we can form a graph whose vertices are the variables and 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 $k \geq 3$. 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 2008].

Example 2.5. Let p be a prime number. An input of 3-LIN(p) is a system of linear equations over the p-element field $\mathrm{GF}(p)$, where each equation contains 3 variables, and the question is whether the system has a solution. This problem is equivalent to $\mathrm{CSP}(\mathcal{D})$, where $D_{\mathrm{3LINp}} = \mathrm{GF}(p)$ and $\mathcal{D}_{\mathrm{3LINp}}$ consists of all affine subspaces of $\mathrm{GF}(p)^3$ of dimension 2 or 3:

$$\mathcal{D}_{\mathrm{3LINp}} = \{R_{abcd} : a, b, c, d \in \mathrm{GF}(p)\}, \text{, where } R_{abcd} = \{(x, y, z) \in \mathrm{GF}(p)^3 : ax{+}by{+}cz = d\} \ .$$

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

Example 2.6. An instance of the s, t-connectivity problem, STCON, is a directed graph and two vertices s, t. The question is whether there exists a directed path from s to t.

A closely related (but not identical) problem is $\mathrm{CSP}(\mathcal{D}_{\mathrm{STCON}})$, where $D_{\mathrm{STCON}} = \{0, 1\}$ and $\mathcal{D}_{\mathrm{STCON}} = \{C_0, C_1, \leq\}$, $C_0 = \{0\}$, $C_1 = \{1\}$, $\leq = \{(0, 0), (0, 1), (1, 1)\}$. Indeed, given an instance of $\mathrm{CSP}(\mathcal{D}_{\mathrm{STCON}})$ we form a directed graph much as we did in Example 2.4 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 $\mathrm{CSP}(\mathcal{D}_{\mathrm{STCON}})$ can be solved by invoking the complement of STCON, the s, t-non-connectivity problem, several times.

Both STCON and $\mathrm{CSP}(\mathcal{D}_{\mathrm{STCON}})$ can clearly be solved in polynomial time. By the Immerman-Szelepcsényi theorem [Immerman 1988; Szelepcsényi 1988] both problems are NL-complete.

In the same way, the s, t-connectivity problem for undirected graphs is closely related to $\mathrm{CSP}(\mathcal{D}_{\mathrm{USTCON}})$, where $D_{\mathrm{USTCON}} = \{0, 1\}$ and $\mathcal{D}_{\mathrm{USTCON}} = \{C_0, C_1, =\}$. These problems are L-complete by [Reingold 2008].

2.3. The dichotomy conjecture

The most fundamental problem in the area was formulated in the landmark paper [Feder and Vardi 1998].

CONJECTURE 2.7 (THE DICHOTOMY CONJECTURE). *For every finite*[2] *constraint language $\mathcal{D}$, the problem $\mathrm{CSP}(\mathcal{D})$ is in P or is NP-complete.*

[2]It is conjectured in [Bulatov et al. 2005] that the dichotomy remains true without the finiteness assumption. Namely, the local-global conjecture states that $\mathrm{CSP}(\mathcal{D})$ is in P (NP-complete) whenever $\mathrm{CSP}(\mathcal{D}')$ is in P (NP-complete) for every (some) finite $\mathcal{D}' \subseteq \mathcal{D}$.

Recall that if P $\neq$ NP, then there are problems of intermediate complexity [Ladner 1975]. Feder and Vardi argued that CSPs over a fixed constraint language is a good candidate for a largest natural class of problems with P versus NP-complete dichotomy.

At that time the conjecture was supported by two major cases: the dichotomy theorem for all languages over the two-element domain [Schaefer 1978] and the dichotomy theorem for languages consisting of a single binary symmetric relation [Hell and Nešetřil 1990].

Feder and Vardi have identified two sources of polynomial-time solvability and made several important contributions toward understanding these sources. 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, $\mathcal{D} = (D; R_1, R_2, \dots)$, is exactly a relational structure, 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 $\{\exists, \forall, \wedge, \vee, \neg, =, \neq\}$. 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 $\{\exists, \forall, \wedge, \vee\}$. The complexity of quantified CSP is also an active research area [Chen 2012] with possible trichotomy P, NP-complete or Pspace-complete. Recently, a tetrachotomy was obtained for the last choice [Madelaine and Martin 2011] – 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 [Feder and Vardi 1998]. The idea of the translation is shown in Examples 2.4, 2.6.

3. UNIVERSAL ALGEBRA IN CSP

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 CSP is that 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 [Jeavons et al. 1997; Jeavons 1998; Bulatov et al. 2005; Larose and Tesson 2009]. The view-

point taken here is close to [Bodirsky 2008]. All results in this section come from these sources unless stated otherwise.

To simplify formulations, 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* $\mathrm{CSP}(\mathcal{D})$.

Example 3.3. 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 $\mathcal{D}$.[3] Formally:

Definition 3.4. 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 $\mathcal{E}$,

[3]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 3.5. *If $\mathcal{D}$ pp-interprets $\mathcal{E}$, then* $\mathrm{CSP}(\mathcal{E})$ *is reducible to* $\mathrm{CSP}(\mathcal{D})$.

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 3.5 then says that the "higher" we are in the poset the "easier" CSP we deal with. 3-SAT is terribly hard – 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 we need a 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 3.6. *Let $\mathcal{D}$ be a constraint language and f an endomorphism of $\mathcal{D}$. Then* $\mathrm{CSP}(\mathcal{D})$ *is reducible to* $\mathrm{CSP}(f(\mathcal{D}))$ *and vice versa, where $f(\mathcal{D})$ is a constraint language with domain $f(D)$ defined by $f(\mathcal{D}) = \{f(R) : R \in \mathcal{D}\}$.*

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 3.7. *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* $\mathrm{CSP}(\mathcal{E})$ *is reducible to* $\mathrm{CSP}(\mathcal{D})$.

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 3.6, 3.7, 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 $\mathrm{CSP}(\mathcal{D})$ is solvable in polynomial time whenever $\mathrm{CSP}(\mathcal{D})$ is. The idea is to gradually guess values for variables using the unary singleton constraints.

3.3. 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.[4]

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

[4]Similar hardness results and conjectures are formulated for other computational/descriptive complexity classes.

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

3.4. Algebraic side

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 unary polymorphism is the same as endomorphism. Endomorphisms can be thought of as symmetries, polymorphisms are then symmetries of higher arity.

The set of all polymorphisms of $\mathcal{D}$ will be denoted by $\mathbf{D}$. This algebraic object is always a *concrete clone* meaning that it contains all projection operations (that is, operations of the form $\pi_i^n(x_1, \dots, x_n) = x_i$, also known as dictators) and is closed under composition. Therefore we refer to $\mathbf{D}$ as the *clone of polymorphims* of $\mathcal{D}$.

The clone of polymorphisms controls pp-definability in the sense of the following old result [Geiger 1968; Bodnarčuk et al. 1969].

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 $\mathbf{D} \subseteq \mathbf{E}$.*[5]

In view of this result, Theorem 3.2 says that the complexity of $\mathrm{CSP}(\mathcal{D})$ only depends on the clone $\mathbf{D}$. More precisely, if $\mathbf{D} \subseteq \mathbf{E}$, then $\mathrm{CSP}(\mathcal{E})$ is reducible to $\mathrm{CSP}(\mathcal{D})$. 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 $\mathrm{CSP}(\mathcal{E})$ to $\mathrm{CSP}(\mathcal{D})$.

Example 3.10. It is a nice exercise to show that the language $\mathcal{D}_{\mathrm{3SAT}}$ of 3-SAT has no polymorphisms but projections. This means that $\mathcal{D}_{\mathrm{3SAT}}$ pp-defines every constraint language with domain $\{0, 1\}$.

Finally, we very briefly discuss the algebraic counterpart to pp-interpretability. The construction in Definition 3.4 corresponds to a similar construction on clones. An alternative viewpoint, which is missing on the relational side, follows from the foundation stone of universal algebra, the Birkhoff HSP theorem [Birkhoff 1935]: pp-interpretability depends on the identities (i.e. universally quantified equations) satisfied by polymorphisms. To illustrate this vague claim, we state one of many (e.g., [Taylor 1977; Hobby and McKenzie 1988; Maróti and McKenzie 2008; Kun and Szegedy 2009; Siggers 2010]) characterizations of the conjectured borderline between P and NP-complete CSPs [Barto and Kozik 2012a].

THEOREM 3.11. *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*

$$(\forall x_1, \dots, x_p \in D)\ t(x_1, \dots, x_p) = t(x_2, \dots, x_p, x_1)\ .$$

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 identities concerning polymorphisms.

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

4. RESULTS

Universal algebra serves the investigation in two ways: as a toolbox containing heavy hammers (such as [Hobby and McKenzie 1988]) 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 [Bulatov 2011]. 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 [Barto et al. 0809].
— The dichotomy conjecture was proved for all constraint languages containing all unary relations [Bulatov 2011] (a simpler proof is in [Barto 2011]).

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 [Feder and Vardi 1998] for formalizations using Datalog programs and games, to [Bulatov et al. 2008] for description using dualities, and to [Bulatov 2011; Barto 2014] 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 $\mathrm{CSP}(\mathcal{D})$) has *width* (k,l), if "yes" answers are correct for every instance of $\mathrm{CSP}(\mathcal{D})$. 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 $\mathcal{D}_{2\mathrm{COLOR}}$ 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.

In [Feder and Vardi 1998], the authors proved that problems 3-LIN(p) (and more generally, similar problems 3-LIN(M) over finite modules) do not have bounded width and conjectured that linear equations are essentially the only obstacle for having bounded width. An algebraic formulation was given by [Larose and Zádori 2007]. They proved that analogues of results in section 3 hold for bounded width, therefore no problem

which pp-interprets the language of 3-LIN(M) has bounded width, and conjectured that the converse is also true. After a sequence of partial results [Kiss and Valeriote 2007; Carvalho et al. 2009; Barto and Kozik 2009; Bulatov 2006], the conjecture was eventually confirmed in [Barto and Kozik 2014][6] and independently in [Bulatov 2009].

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(M) for a finite module M.[7]

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 [Feder and Vardi 1998; Bulatov 2002; Bulatov and Dalmau 2006; Dalmau 2006] culminating in [Idziak et al. 2007; Berman et al. 2009] 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 polymorphism. A relation on D is a *subpower* of $\mathbf{D}$ if it is pp-definable from $\mathcal{D}$. Note that the set of solutions of any instance of $\mathrm{CSP}(\mathcal{D})$ 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).[8]

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

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 [Bodirsky 2008]). 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 [Bulatov 2013] (substantially simplified in [Dyer and Richerby 2013]) and for the robust satisfiability problem in [Barto and Kozik 2012b]. A generalization of the theory for the optimization problem and valued CSPs was given in [Cohen et al. 2013], and some links to universal algebra are emerging from research in the area of approximation algorithms (such as [Raghavendra 2008]).

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

ACKNOWLEDGMENTS

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

REFERENCES

Sanjeev Arora and Boaz Barak. 2009. *Computational Complexity: A Modern Approach* (1st ed.). Cambridge University Press, New York, NY, USA.

[6] A modification required to handle infinite languages was given in [Barto 2014].

[7] 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.

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

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

Libor Barto. 2014. The Collapse of the Bounded Width Hierarchy. (2014). manuscript.

Libor Barto and Marcin Kozik. 2009. Congruence Distributivity Implies Bounded Width. *SIAM J. Comput.* 39, 4 (2009), 1531–1542.

Libor Barto and Marcin Kozik. 2012a. Absorbing Subalgebras, Cyclic Terms, and the Constraint Satisfaction Problem. *Logical Methods in Computer Science* 8, 1 (2012).

Libor Barto and Marcin Kozik. 2012b. Robust satisfiability of constraint satisfaction problems. In *Proceedings of the 44th Symposium on Theory of Computing (STOC '12)*. ACM, New York, NY, USA, 931–940.

Libor Barto and Marcin Kozik. 2014. Constraint Satisfaction Problems Solvable by Local Consistency Methods. *J. ACM* 61, 1, Article 3 (Jan. 2014), 19 pages.

Libor Barto, Marcin Kozik, and Todd Niven. 2008/09. 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.* 38, 5 (2008/09), 1782–1802.

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

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

Manuel Bodirsky. 2008. Constraint Satisfaction Problems with Infinite Templates. In *Complexity of Constraints* (2009-05-05) *(Lecture Notes in Computer Science)*, Nadia Creignou, Phokion G. Kolaitis, and Heribert Vollmer (Eds.), Vol. 5250. Springer, 196–228.

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

Andrei Bulatov. 2009. Bounded relational width. (2009). manuscript.

Andrei Bulatov and Víctor Dalmau. 2006. A simple algorithm for Mal′tsev constraints. *SIAM J. Comput.* 36, 1 (2006), 16–27 (electronic).

Andrei Bulatov, Peter Jeavons, and Andrei Krokhin. 2005. Classifying the Complexity of Constraints Using Finite Algebras. *SIAM J. Comput.* 34 (March 2005), 720–742. Issue 3.

Andrei A. Bulatov. 2002. Mal'tsev constraints are tractable. *Electronic Colloquium on Computational Complexity (ECCC)* 034 (2002).

Andrei A. Bulatov. 2006. Combinatorial problems raised from 2-semilattices. *J. Algebra* 298, 2 (2006), 321–339.

Andrei A. Bulatov. 2011. Complexity of Conservative Constraint Satisfaction Problems. *ACM Trans. Comput. Logic* 12, 4, Article 24 (July 2011), 66 pages.

Andrei A. Bulatov. 2013. The Complexity of the Counting Constraint Satisfaction Problem. *J. ACM* 60, 5, Article 34 (Oct. 2013), 41 pages.

Andrei A. Bulatov, Andrei Krokhin, and Benoit Larose. 2008. Complexity of Constraints. Springer-Verlag, Berlin, Heidelberg, Chapter Dualities for Constraint Satisfaction Problems, 93–124.

Jin-Yi Cai and Xi Chen. 2012. Complexity of Counting CSP with Complex Weights. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing (STOC '12)*. ACM, New York, NY, USA, 909–920.

Catarina Carvalho, Víctor Dalmau, Petar Marković, and Miklós Maróti. 2009. CD(4) has bounded width. *Algebra Universalis* 60, 3 (2009), 293–307.

Hubie Chen. 2012. Meditations on Quantified Constraint Satisfaction. In *Logic and Program Semantics*, Robert L. Constable and Alexandra Silva (Eds.). Lecture Notes in Computer Science, Vol. 7230. Springer Berlin Heidelberg, 35–49.

David A. Cohen, Martin C. Cooper, Páidí Creed, Peter Jeavons, and Stanislav Živný. 2013. An algebraic theory of complexity for discrete optimisation. *SIAM J. Comput.* 42, 5 (2013), 1915–1939.

Víctor Dalmau. 2006. Generalized majority-minority operations are tractable. *Log. Methods Comput. Sci.* 2, 4 (2006), 4:1, 14.

Martin E. Dyer and David Richerby. 2013. An Effective Dichotomy for the Counting Constraint Satisfaction Problem. *SIAM J. Comput.* 42, 3 (2013), 1245–1274.

Tomás Feder and Moshe Y. Vardi. 1998. The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory. *SIAM J. Comput.* 28, 1 (1998), 57–104.

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

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

Pavol Hell and Jaroslav Nešetřil. 1990. On the complexity of H-coloring. *J. Combin. Theory Ser. B* 48, 1 (1990), 92–110.

David Hobby and Ralph McKenzie. 1988. *The structure of finite algebras*. Contemporary Mathematics, Vol. 76. American Mathematical Society, Providence, RI. xii+203 pages.

Pawel Idziak, Petar Marković, Ralph McKenzie, Matthew Valeriote, and Ross Willard. 2007. Tractability and learnability arising from algebras with few subpowers. In *Proceedings of the Twenty-Second Annual IEEE Symposium on Logic in Computer Science (LICS 2007)*. IEEE Computer Society Press, 213–222.

Neil Immerman. 1988. Nondeterministic Space is Closed Under Complementation. *SIAM J. Comput.* 17, 5 (Oct. 1988), 935–938.

Peter Jeavons. 1998. On the algebraic structure of combinatorial problems. *Theoretical Computer Science* 200, 12 (1998), 185 – 204.

Peter Jeavons, David Cohen, and Marc Gyssens. 1997. Closure properties of constraints. *J. ACM* 44, 4 (1997), 527–548.

Emil Kiss and Matthew Valeriote. 2007. On tractability and congruence distributivity. *Log. Methods Comput. Sci.* 3, 2 (2007), 2:6, 20 pp. (electronic).

Gábor Kun and Mario Szegedy. 2009. A New Line of Attack on the Dichotomy Conjecture. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing (STOC '09)*. ACM, New York, NY, USA, 725–734.

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

Benoît Larose and Pascal Tesson. 2009. Universal algebra and hardness results for constraint satisfaction problems. *Theor. Comput. Sci.* 410 (April 2009), 1629–1647. Issue 18.

Benoit Larose and László Zádori. 2007. Bounded width problems and algebras. *Algebra Universalis* 56, 3-4 (2007), 439–466.

F. Madelaine and B. Martin. 2011. A Tetrachotomy for Positive First-Order Logic without Equality. In *26th Annual IEEE Symposium on Logic in Computer Science—LICS 2011*. IEEE Computer Soc., Los Alamitos, CA, 311–320.

Miklós Maróti and Ralph McKenzie. 2008. Existence theorems for weakly symmetric operations. *Algebra Universalis* 59, 3-4 (2008), 463–489.

Christos H. Papadimitriou. 1994. *Computational complexity*. Addison-Wesley Publishing Company, Reading, MA. xvi+523 pages.

Prasad Raghavendra. 2008. Optimal algorithms and inapproximability results for every CSP?. In *STOC'08*. 245–254.

Omer Reingold. 2008. Undirected Connectivity in Log-space. *J. ACM* 55, 4, Article 17 (Sept. 2008), 24 pages.

Francesca Rossi, Peter van Beek, and Toby Walsh. 2006. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., New York, NY, USA.

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

Mark H. Siggers. 2010. A strong Malcev condition for locally finite varieties omitting the unary type. *Algebra universalis* 64, 1-2 (2010), 15–20.

R. Szelepcsényi. 1988. The Method of Forced Enumeration for Nondeterministic Automata. *Acta Inf.* 26, 3 (Nov. 1988), 279–284.

Walter Taylor. 1977. Varieties obeying homotopy laws. *Canad. J. Math.* 29, 3 (1977), 498–527.

Stanislav Živný. 2012. *The complexity of valued constraint satisfaction problems*. Springer, Heidelberg. xviii+170 pages.