Towards a Dichotomy Theorem for the Counting Constraint Satisfaction Problem

Andrei A. Bulatov
Computing Laboratory
University of Oxford, Oxford, UK
Andrei.Bulatov@comlab.ox.ac.uk

Víctor Dalmau
Department de Tecnologia
Universitat Pompeu Fabra, Barcelona, Spain
Victor.Dalmau@tecn.upf.es

Abstract

The Counting Constraint Satisfaction Problem (#CSP) over a finite domain can be expressed as follows: given a first-order formula consisting of a conjunction of predicates, determine the number of satisfying assignments to the formula. #CSP can be parametrized by the set of allowed constraint predicates. In this paper we start a systematic study of subclasses of #CSP restricted in this way. The ultimate goal of this investigation is to distinguish those restricted subclasses of #CSP which are tractable, i.e. solvable in polynomial time, from those which are not. We show that the complexity of any restricted #CSP class on a finite domain can be deduced from the properties of polymorphisms of the allowed constraints, similar to that for the decision CSP. Then we prove that if a subclass of the #CSP is tractable, then constraints allowed by the class satisfy some very restrictive condition: it has to have a Mal'tsev polymorphism, that is a ternary operation $m(x, y, z)$ *such that* $m(x, y, y) = m(y, y, x) = x$*. This condition uniformly explains all existing complexity results for particular cases of #CSP, and allows us to obtain new results and to conjecture a criteria distinguishing tractable counting CSPs. We also obtain a dichotomy theorem for the complexity of #CSP with a 3-element domain and give a new simpler proofs of the dichotomy results for the problem of counting graph homomorphisms.*

1. Introduction

In a counting combinatorial problem the objective is to find the number of feasible solutions to a certain search problem. Similar to its decision counterpart, the Counting Constraint Satisfaction Problem (#CSP) can be used to provide a generic framework for numerous combinatorial problems that arise frequently in a wide range of areas from logic, graph theory, and artificial intelligence [4, 13, 20, 24, 30, 37, 40, 45, 46, 49, 50], to statistical physics [3, 11, 35].

The prototypical counting problem, #SAT, i.e. the problem of counting the number of assignments that satisfy a CNF formula, constitutes an important particular case of #CSP. Since the pionnering papers of Valiant [49, 50] much efforts have been made to investigate the computational complexity of counting satisfying assignments to propositional formulas of various types [13, 37, 45, 46, 49, 50]. In particular, it has been found that #SAT is much more computationally demanding than its decision counterpart SAT, and is #P-complete even for Horn or monotone formulas, and even when the size of clauses and the number of occurrences of a variable in the formula are extremely limited. Finally, in [13], Creignou and Hermann have obtained a dichotomy theorem for #SAT, similar to that of Schaefer [47] for (decision) SAT.

The formalism of constraint networks, introduced by Montanari [39] provides a natural generalization of propositional formulas to domains with more than 2 elements. A constraint network is given by a collection of variables, a domain, and a family of constraints specifying some restrictions on the values of the domain that can be taken simultaneously by the variables. The problem of deciding whether there exists a solution to a constraint network, known as the constraint satisfaction problem (CSP) has recieved considerable attention in theoretical computer science and it also constitutes one of the major lines of research in artificial intelligence.

The class of counting constraint satisfaction problems is defined as the counting version on CSP, i.e. the problem of finding the number of solution to a constraint network. This problem can also be reformulated as (1) the problem of finding the number of models of a conjunctive formula (this is the form we use throughout the paper), as (2) the problem of counting the number of homomorphisms between two finite relational structures $\mathbf{A}$ and $\mathbf{B}$, and also as (3) the problem of computing the size (number of tuples) of the evaluation $Q(D)$ of a conjuntive query (without projection) Q on a database D; see [22, 34].

Further examples of combinatorial problems expressible in a natural way in #CSP terms include problems from propositional logic [13, 46], classical combinatorial problems such as #CLIQUE, GRAPH RELIABILITY, ANTICHAIN, PERMANENT [37, 45, 49, 50], enumerating of graph homomorphisms, and many others [4, 20, 24, 30].

A particular case of the counting graph homomorphisms problem, the class of $\#H$-COLORING problems, attracts a special attention. In a $\#H$-COLORING problem the goal is to count the number of homomorphisms from a graph G (the input) to a fixed graph H. Very recently, Dyer and Greenhill [20] have proved that, for every undirected graph H, its associated $\#H$-COLORING problem is either in FP or #P-complete and they have also provided a complete characterization of the tractable problems. This result has been extended to the counting LIST $\#H$-COLORING problem [18], which allows additional restrictions on possible images of a node. Furthermore, some other variants of the $\#H$-COLORING problem for undirected graphs have been intensively studied during last several years [16, 17].

Despite the large amount of results on its decision counterpart, numerous problems that can be stated in the form of the #CSP and massive work done for certain particular cases of #CSP, this problem, leaving apart the Boolean case (i.e. #SAT), has received little study. In this paper, we embark on a systematic study of the computational complexity of the subclasses of the #CSP parametrized by the set of allowed constraints. The ultimate goal of this study is to identify those restrictions which being imposed on the possible form of constraints lead to a problem solvable in polynomial time. To this end, we prove a somewhat surprising result claiming that the restricted classes of #CSP can be parametrized by sets of operations, polymorphisms, so that certain properties of the polymorphisms determine the complexity of a class. An analogous approach has proved to be very fruitful in the study of the decision constraint satisfaction problem [8, 9, 10, 31, 32, 33], and we expect that it also will be useful in the study of counting problems.

This algebraic approach allows us to identify a property common for all tractable restricted #CSP: we show that every #CSP-class having no polymorphism of a certain special type is #P-complete. Operations of this type are said to be Mal'tsev; these are ternary operations $m(x, y, z)$ such that $m(x,y,y) = m(y,y,x) = x$. Mal'tsev operations have been an object of intensive investigation in algebra for many years. We therefore get a new simple and powerful method of proving hardness results which has always been the most difficult part of any study of counting problems. Using this result we give a generic explanation to all existing complexity results on counting CSP and its particular cases including different types of $\#H$-COLORING [17, 16, 18, 20], and constraints considered in [15, 21, 22]. We also pose, for the first time, a plausible conjecture on a possible form of a criteria for #CSP, and prove this conjecture in the case of constraints on a 3-element domain. Another advantage of this approach is that it provides a new type of counting algorithms using deep algebraic results along with usual combinatorial methods.

Due to space restrictions most of the proofs are omitted. They can be found in [7].

2. The Counting Constraint Satisfaction Problem

2.1. Definitions and Examples

The counting constraint satisfaction problem can be formulated in several ways (see Section 1). We use the logical form of this problem. Throughout the paper we use the standard correspondence between predicates and relations: a relation consists of all tuples of values for which the corresponding predicate is true. We will use the same symbol for a predicate and its corresponding relation, since the meaning will always be clear from the context. By a *constraint language* on a set A we mean an arbitrary set of finitary predicates on A.

Definition 1 *Let A be a finite set, and Γ a constraint language on A. An* instance *of $\#\mathrm{CSP}(\Gamma)$ is a conjunctive formula $F_1 \wedge \ldots \wedge F_n$ where each F_i is an atomic formula interpreted as a predicate from Γ. The* objective *is to find the number of satisfying assignments (models) to the formula.*

Example 1 (#SAT, **[13, 14, 49, 50]**) An instance of the #SAT problem is specified by giving a propositional logic formula in CNF, and asking how many assignments satisfy it. Therefore, #SAT is equivalent to $\#\mathrm{CSP}(\Gamma)$ where Γ is the set of predicates defined on a two-element set and expressible by clauses. □

Example 2 (ANTICHAIN, **[45]**) In the problem ANTICHAIN we are given a finite poset $(P; \leq)$, and we aim to compute the number of antichains in P. This problem can be expressed in the #CSP-form as follows. Let $\varrho_{\prec}$ be the predicate of the natural order on $A = \{0, 1\}$. We assign a variable x_a to each element $a \in P$. Then the $\#\mathrm{CSP}(\{\varrho_{\prec}\})$ instance $\Phi = \bigwedge_{a \leq b} \varrho_{\prec}(x_a, x_b)$ is equivalent to the original ANTICHAIN instance.

To show this, notice that every model φ to Φ satisfies the following condition: if $\varphi(x_a) = 1$ and $a \leq b$ then $\varphi(x_b) = 1$. This means that the set $F_\varphi = \{a \in P \mid \varphi(x_a) = 1\}$ is a *filter* of P, that is a set such that if $a \in F_\varphi$ and $a \leq b$ then $b \in F_\varphi$. Finally, notice that there is a one-to-one correspondence between antichains of the poset P and its filters. Indeed, for any antichain $H \subseteq P$, the set $\{q \in P \mid$ there is $p \in H$ such that $p \leq q\}$ is a filter. Conversely, for a filter $F \subseteq P$, the set $\{q \in F \mid$ if $p \in F$ and $p \leq q$ then $p =$

$q\}$ of *minimal* elements from F forms an antichain. Thus the models of Φ one-to-one correspond to the filters of P, and consequently, to the antichains of P.

On the other hand, any $\#\mathrm{CSP}(\{\varrho_\prec\})$ instance is reducible to an ANTICHAIN instance, though not so straightforwardly. Thus ANTICHAIN is equivalent to $\#\mathrm{CSP}(\{\varrho_\prec\})$. $\square$

Example 3 ($\#H$-COLORING, [20, 27, 36]) Let H be a (directed) graph. In the H-COLORING problem we are asked, given a graph G, whether there is a homomorphism from G to H. Correspondingly in its counting version, $\#H$-COLORING, the objective is to find the number of such homomorphisms.

Every (directed) graph $H = (V(H), E(H))$ corresponds to a binary predicate ϱ_H: $\varrho_H(a, b)$ holds iff (a, b) is an edge of H. Then every instance $G = (V(G); E(G))$ of the $\#H$-COLORING corresponds to the instance $\bigwedge_{(a,b)\in E(G)} \varrho_H(x_a, x_b)$ of #CSP where x_a is a variable assigned to a vertex $a \in V(G)$. Therefore $\#H$-COLORING is equivalent to $\#\mathrm{CSP}(\{\varrho_H\})$. $\square$

Note: The notion of reduction used in the paper is one by oracles, as in [37, 45, 49, 50]. In fact, this notion of reduction was used also in [14], as its #P-completeness results relies upon the results of [37, 45, 49, 50]. Thus a problem L is #P-complete if $L \in \#\mathrm{P}$ and $\#P \subseteq \mathrm{FP}^L$. The general #CSP is known to be #P-complete, as follows from [50] and the examples above. However, some restricted problems have been shown to be computable in polynomial time.

Definition 2 *A constraint language* Γ *is called* #-tractable *if for any finite* $\Gamma' \subseteq \Gamma$ *the problem* $\#\mathrm{CSP}(\Gamma')$ *is solvable in polynomial time.*

A constraint language Γ *is called* #P-complete *if* $\#\mathrm{CSP}(\Gamma')$ *is #P-complete for a certain finite* $\Gamma' \subseteq \Gamma$.

Notice that this 'local' notion of tractability perfectly suits our aims, since, on the one hand, it deals with potentially infinite sets of predicates, that makes it possible to obtain general theoretical results; on the other hand, it is applicable to practical problems, because those mostly use finite sets of allowed predicates.

We are in a position to pose the main problem we tackle in this paper.

Problem 1 (classification problem) *Characterize #-tractable and #P-complete constraint languages on finite sets.*

This problem is completely solved for #SAT [13, 14].

Theorem 1 ([13, 14]) *A Boolean (that is on a 2-element set) constraint language* Γ *is #-tractable if and only if every relation from* Γ *is the solution space of a system of linear equations over the 2-element field. Otherwise,* Γ *is #P-complete.*

3. Invariance properties of #CSP

3.1. Relational clones

The main idea in tackling Problem 1 is to reduce the number of constraint languages to be considered by determining what predicates can be added to a #-tractable constraint language so that the obtained language remains #-tractable. This idea has been used in [13, 14] for #SAT, where new predicates are derived by the construction of *faithful implementation*. The following construction is equivalent to faithful implementation when $|A| = 2$, and, in fact, is prompted by the form of #CSP instances.

Proposition 1 *Let* Γ *be a constraint language on a finite set* A*, and* $=_A$ *is the predicate of equality on* A*. If* ϱ *is expressible by a conjunctive formula using predicates from* Γ*, then* $\#\mathrm{CSP}(\Gamma \cup \{\varrho\})$ *and* $\#\mathrm{CSP}(\Gamma \cup \{=_A\})$ *are polynomial time reducible to* $\#\mathrm{CSP}(\Gamma)$.

To prove Proposition 1 we observe that every instance from $\#\mathrm{CSP}(\Gamma \cup \{\varrho\})$ can be transformed to an instance of $\#\mathrm{CSP}(\Gamma)$ by replacing each occurence of ϱ with its representation by a conjunction of predicates from Γ. Clearly, such a transformation preserves the number of satisfying assignments. Furthermore, there is a standard procedure of eliminating predicates of equality from a conjunctive formula without changing the size of the set of models [31]; for more details see [7].

Further constructions preserving #-tractability are much less easy and obvious.

Proposition 2 *Let* Γ *be a constraint language, and* $\varrho(x_1, \ldots, x_n) \in \Gamma$*. Then* $\#\mathrm{CSP}(\Gamma \cup \{\exists x_n \varrho\})$ *is polynomial time reducible to* $\#\mathrm{CSP}(\Gamma)$.

Proof: We use the interpolation technique first observed in [50]: given an $n \times n$ integer matrix with each entry bounded in magnitude by 2^m, the determinant and the inverse can be computed in time polynomial in n and m.

Let $\sigma = \exists x_n \varrho(x_1, \ldots, x_{n-1}, x_n)$, $\varrho \in \Gamma$. Basically, for each tuple $(a_1, \ldots, a_{n-1}) \in \sigma$ there are several b such that $\varrho(a_1, \ldots, a_{n-1}, b)$ holds. If $\sigma = \{\mathbf{a}_1, \ldots, \mathbf{a}_q\}$ where every $\mathbf{a}_j$ is an $n-1$-tuple over A, then let u_j be the number of extensions of $\mathbf{a}_j$.

Take a problem instance $\Phi = F_1 \wedge \ldots \wedge F_t \in \#\mathrm{CSP}(\Gamma \cup \{\sigma\})$. Without loss of generality we may assume that $F_1, \ldots, F_s$ are the predicates equal to σ. Let $l \geq 1$ and let $\Phi^{(l)}$ be the problem instance from $\#\mathrm{CSP}(\Gamma)$ defined as follows:

$$\Phi^{(l)} = \bigwedge_{i=1}^{s} (F_i^1 \wedge \ldots \wedge F_i^l) \wedge F_{s+1} \wedge \ldots \wedge F_t$$

where, for each $i = 1, \dots, s$, if $F_i = \sigma(x_{i_1}, \dots, x_{i_{n-1}})$ then $F_i^j = \varrho(x_{i_1}, \dots, x_{i_{n-1}}, y_{ij})$, the variables y_{ij} are all different and do not occur in Φ.

For a solution φ to Φ, let $a_\varphi(j)$ denote the number of predicates $F_i(x_{i_1}, \dots, x_{i_{n-1}})$, $i \le s$, such that $(\varphi(x_{i_1}), \dots, \varphi(x_{i_{n-1}})) = \mathbf{a}_j \in \sigma$. Clearly, $a_\varphi(1) + \dots + a_\varphi(q) = s$. Furthermore, let $N(b_1, \dots, b_q)$ denote the number of those solutions φ for which $a_\varphi(j) = b_j$, $1 \le j \le q$. To solve the problem Φ we have to find the sum of all numbers of this form. Each $N(b_1, \dots, b_q)$ corresponds to a partition of s in q nonnegative summands $b_1 + \dots + b_q$. Hence the number p of Ns does not exceed $(s+1)^q$. Since q depends only on Γ and σ, this number is bounded by a polynomial in the size of Φ.

Every solution φ of Φ can be extended to a solution of $\Phi^{(l)}$. To this end, for each predicate $F_i(x_{i_1}, \dots, x_{i_{n-1}})$, $i \le s$, of Φ, let $(\varphi(x_{i_1}), \dots, \varphi(x_{i_{n-1}})) = \mathbf{a}_j \in \sigma$. The tuple $\mathbf{a}_j$ can be extended to a tuple from ϱ in one of u_j ways. As F_i corresponds to l predicates in $\Phi^{(l)}$, the number of extensions for $\mathbf{a}_j$ is u_j^l. Since the same holds for each constraint we get

$$(u_1^l)^{a_\varphi(1)} \cdot \dots \cdot (u_q^l)^{a_\varphi(q)} = (u_1^{a_\varphi(1)} \cdot \dots \cdot u_q^{a_\varphi(q)})^l$$

extensions of φ to a solution of $\Phi^{(l)}$. The problem instance $\Phi^{(l)}$ belongs to $\#\mathrm{CSP}(\Gamma)$ and the number of its solutions is

$$N_l = \sum_{b_1+\dots+b_q=s} N(b_1, \dots, b_q)(u_1^{b_1} \cdot \dots \cdot u_q^{b_q})^l.$$

To avoid cumbersome notation we assume that all the numbers $u_1^{b_1} \cdot \dots \cdot u_q^{b_q}$ are different. It will not be loss of generality because if $u_1^{b_1} \cdot \dots \cdot u_q^{b_q} = u_1^{c_1} \cdot \dots \cdot u_q^{c_q}$ then we may just replace $N(b_1, \dots, b_q)$ and $N(c_1, \dots, c_q)$ with their sum, and so shorten the sum above. The matrix of the linear system

$$\begin{aligned}
\sum_{b_1+\dots+b_q=s} N(b_1, \dots, b_q) u_1^{b_1} \cdot \dots \cdot u_q^{b_q} &= N_1 \\
\sum_{b_1+\dots+b_q=s} N(b_1, \dots, b_q)(u_1^{b_1} \cdot \dots \cdot u_q^{b_q})^2 &= N_2 \\
&\vdots \\
\sum_{b_1+\dots+b_q=s} N(b_1, \dots, b_q)(u_1^{b_1} \cdot \dots \cdot u_q^{b_q})^p &= N_p
\end{aligned}$$

is Vandermonde, and therefore the system can be solved in polynomial time. $\square$

Following are some examples of predicates that can be expressed by conjunction and existential quantification.

Example 4 (1) *Cartesian product* of relations (predicates). Let ϱ, σ be n- and m-ary relations respectively. Then

$$\begin{aligned}
&(\varrho \times \sigma)(x_1, \dots, x_n, x_{n+1}, \dots, x_{n+m} \\
&= \varrho(x_1, \dots, x_n) \wedge \sigma(x_{n+1}, \dots, x_{n+m}).
\end{aligned}$$

(2) The standard *composition* of binary relations (predicates). Let ϱ, σ be binary relations. Then

$$(\varrho \circ \sigma)(x, y) = \exists z\ \varrho(x, z) \wedge \sigma(z, y).$$

(3) Let ϱ be an n-ary relation (predicate), and $I = \{i_1, \dots, i_k\} \subseteq \{1, \dots, n\}$. The relation $\sigma = \mathrm{pr}_I \varrho$ called the *projection of ϱ onto I* is defined as follows

$$\sigma(x_{i_1}, \dots, x_{i_k}) = \exists x_i (i \notin I)\ \varrho(x_1, \dots, x_n).$$

Constraint languages containing the predicate of equality and closed with respect to taking conjuction and existential quantification have been intensively studied (see e.g. [41, 42, 43]) and have provided strong assistance in the study of the decision constraint satisfaction problem [9, 10, 31, 33].

Definition 3 *A constraint language Γ is said to be a* relational clone *if it contains the predicate of equality, and every predicate representable by a formula using predicates from Γ, conjunction, and existential quantification, belongs to Γ.*

For a constraint language Δ, the relational clone, consisting of all predicates expressible using predicates from Δ, conjunction, and existential quantification is denoted by $\langle \Delta \rangle$.

Propositions 1,2 imply the first main result of the paper.

Theorem 2 *Let Γ_1, Γ_2 be constraint languages on a finite set A such that Γ_2 is finite and $\Gamma_2 \subseteq \langle \Gamma_1 \rangle$. Then $\#\mathrm{CSP}(\Gamma_2)$ is polynomial time reducible to $\#\mathrm{CSP}(\Gamma_1)$.*

Corollary 1 *A constraint language Γ is #-tractable (#P-complete) if and only if so is $\langle \Gamma \rangle$.*

3.2. Polymorphisms

The results of the previous section show that the class of constraint languages to be studied can be considerably reduced. To reduce it even further we use the invariance property of relations [31, 33, 43]. Recall that A^n denotes the set of all n-tuples of elements from A. Any operation on a set A can be extended in a standard way to an operation on tuples over A, as follows. For any (m-ary) operation f, and any collection of tuples $\mathbf{a}_1, \dots, \mathbf{a}_m \in A^n$, where $\mathbf{a}_i = (\mathbf{a}_i[1], \dots, \mathbf{a}_i[n])$ ($i = 1 \dots m$), define $f(\mathbf{a}_1, \dots, \mathbf{a}_m)$ to be $(f(\mathbf{a}_1[1], \dots, \mathbf{a}_m[1]), \dots, f(\mathbf{a}_1[n], \dots, \mathbf{a}_m[n]))$. Then f *preserves* an n-ary relation ϱ (or ϱ is *invariant* under f, or f is a *polymorphism of ϱ*) if for any $\mathbf{a}_1, \dots, \mathbf{a}_m \in \varrho$ the tuple $f(\mathbf{a}_1, \dots, \mathbf{a}_m)$ belongs to ϱ. For a given set of operations, C, the set of all relations invariant under every operation from C is denoted by $\mathsf{Inv}\ C$. Conversely, for a constraint language, Γ, the set of all operations preserving every relation from Γ is denoted by $\mathsf{Pol}\ \Gamma$.

Example 5 *Let ϱ be the solution space of a system of linear equations over a field F. Then the operation $m(x, y, z) = x - y + z$ is a polymorphism of ϱ. Indeed, let $A \cdot \mathbf{x} = \mathbf{b}$ be the system defining ϱ, and $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \varrho$. Then*

$$A \cdot m(\mathbf{x}, \mathbf{y}, \mathbf{z}) = A \cdot (\mathbf{x} - \mathbf{y} + \mathbf{z}) = A \cdot \mathbf{x} - A \cdot \mathbf{y} + A \cdot \mathbf{z} = \mathbf{b}.$$

In fact, the converse can also be shown: if ϱ is invariant under m then it is the solution space of a certain system of linear equations. □

The sets of the form $\mathsf{Inv}\ C$ are relational clones, and every relational clone can be represented in this form [43, 48]. For example, for a relational clone Γ, we have $\Gamma = \mathsf{Inv}\ \mathsf{Pol}\ \Gamma$.

Proposition 3 ([43, 48]) *If Γ is a constraint language then $\langle\Gamma\rangle = \mathsf{Inv}\ \mathsf{Pol}\ \Gamma$. Therefore, $\mathsf{Pol}\ \Gamma = \mathsf{Pol}\ \langle\Gamma\rangle$.*

As a consequence of Theorem 2 and Proposition 3 we deduce the following important result that constitutes the basis of the algebraic approach to the counting CSP.

Theorem 3 *Let Γ_1, Γ_2 be constraint languages on A such that Γ_2 is finite and $\mathsf{Pol}\ \Gamma_1 \subseteq \mathsf{Pol}\ \Gamma_2$. Then $\#\mathrm{CSP}(\Gamma_2)$ is polynomial time reducible to $\#\mathrm{CSP}(\Gamma_1)$. Therefore, if $\#\mathrm{CSP}(\Gamma_1)$ is #-tractable then so is $\#\mathrm{CSP}(\Gamma_2)$.*

Consequently, all the information about the complexity of $\#\mathrm{CSP}(\Gamma)$ can be extracted from the family of polymorphisms of Γ. Sets of polymorphisms often provide a more convenient and concise way of describing a class of constraint satisfaction problems. In particular, the dichotomy result for Boolean constraint languages can be reformulated as follows.

Theorem 4 ([14]) *A Boolean constraint language Γ is #-tractable if and only if every relation from Γ is invariant with respect to the operation $x - y + z$ where $+, -$ are addition and subtraction modulo 2. Otherwise Γ is #P-complete.*

The operation $x - y + z$ is one of the simplest examples of so called Mal'tsev operations: a ternary operation $m(x, y, z)$ on a set A is said to be *Mal'tsev* if it satisfies the condition $m(x, y, y) = m(y, y, x) = x$ for any $x, y \in A$. Another well known example of a Mal'tsev operation is the operation $xy^{-1}z$ of a group.

The following theorem, our second main result, shows that Mal'tsev operations play, possibly, the crucial role in the study of #CSP.

Theorem 5 *If Γ is a constraint language which is invariant under no Mal'tsev operation then Γ is #P-complete.*

In the next section we apply Theorem 5 to some particular cases of #CSP, obtaining new algorithms, reproving and generalizing existing results. We show that in all cases considered the existence of a Mal'tsev polymorphism is sufficient for the #-tractability of a constraint language. Therefore the following conjecture seems to be plausible.

Conjecture 1 (dichotomy conjecture) *A constraint language Γ is #-tractable if and only if it is invariant under a Mal'tsev operation. Otherwise Γ is #P-complete.*

In Section 5 we develop the technique that allows us to prove Theorem 5.

4. Applications

4.1. Small domains

By making use of Theorem 5 we may obtain a very easy proof of the dichotomy theorem for #SAT [13]. On the one hand, by the results of [44], if a Boolean constraint language Γ is invariant with respect to a Mal'tsev operation it is also invariant with respect to $x - y + z$. Therefore, any #-tractable Boolean constraint language is invariant with respect to $x - y + z$. On the other hand, any relation from such a language is the solution space of a system of linear equation over a 2-element field. Hence it is possible to find its basis in polynomial time, and furthermore, the number of solutions equals 2^n where n is the number of vectors in the basis.

Mal'tsev operations on a 3-element set and the structure of relations invariant with respect to them are exhaustively studied in [5]. Making use of these results we are able to prove Conjecture 1 for constraint languages over a 3-element set.

Theorem 6 *A constraint language Γ over a 3-element set is #-tractable if and only if $\mathsf{Pol}\ \Gamma$ contains a Mal'tsev operation.*

4.2. Rectangularity and permutability

Relations invariant with respect to a Mal'tsev operation satisfy strong restrictions on their form. One of them is especially useful. An (n-ary) relation ϱ is said to be *rectangular* if, for any partition of $\{1, \ldots, n\}$ into subsets I, J, and any $\mathbf{a}, \mathbf{b} \in \mathrm{pr}_I \varrho$, $\mathbf{c}, \mathbf{d} \in \mathrm{pr}_J \varrho$, if $(\mathbf{a}, \mathbf{c}), (\mathbf{a}, \mathbf{d}), (\mathbf{b}, \mathbf{c}) \in \varrho$, then $(\mathbf{b}, \mathbf{d}) \in \varrho$, see Fig. 1 ($(\mathbf{a}, \mathbf{c})$ here denotes the tuple $\mathbf{e}$ such that $\mathbf{e}[i] = \mathbf{a}[i]$ if $i \in I$ and $\mathbf{e}[i] = \mathbf{c}[i]$ if $i \in J$).

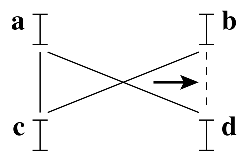

Figure 1. The property of rectangularity

Indeed, if $(\mathbf{a}, \mathbf{c}), (\mathbf{a}, \mathbf{d}), (\mathbf{b}, \mathbf{c}) \in \varrho$ then

$$m\left(\begin{pmatrix}\mathbf{a}\\ \mathbf{d}\end{pmatrix}, \begin{pmatrix}\mathbf{a}\\ \mathbf{c}\end{pmatrix}, \begin{pmatrix}\mathbf{b}\\ \mathbf{c}\end{pmatrix}\right) = \begin{pmatrix}\mathbf{b}\\ \mathbf{d}\end{pmatrix} \in \varrho.$$

By Theorem 5, we get

Corollary 2 *If Γ is a #-tractable constraint language, then every relation from Γ is rectangular. If Γ contains a non-rectangular relation, then Γ is #P-complete.*

Unfortunately, the rectangularity of a relation or a set of relations does not guarantee existence of a Mal'tsev polymorphism. For instance, if Γ is a constraint language, m is a Mal'tsev polymorphism of Γ, and $\varrho_1, \varrho_2 \in \langle\Gamma\rangle$ are equivalence relations, then, in spite of they are always rectangular, they also must be *permutable*, that is $\varrho_1 \circ \varrho_2 = \varrho_2 \circ \varrho_1$. A proof of this fact is non-trivial and can be found e.g. in [12, 38].

4.3. The $\#H$-Coloring problem

The problem GRAPH HOMOMORPHISM [23] and its counting counterpart are among the most well established combinatorial problems. H-COLORING and $\#H$-COLORING problems constitute their subproblems when the target graph H is fixed. A massive work has been done in the study of the complexity of H-COLORING and $\#H$-COLORING for different types of the graph H, and also for restrictions on the class of source graphs [1, 2, 17, 18, 19, 25, 26, 28, 51].

In particular, undirected graphs H for which H-COLORING is tractable have been characterized in [27]. An analogous result for $\#H$-COLORING has been obtained by Dyer and Greenhill.

Theorem 7 ([20]) *If every connected component of an undirected graph H is either an isolated vertex, or a complete graph with all loops present, or a complete unlooped bipartite graph, then $\#H$-COLORING is #-tractable. Otherwise, it is #P-complete.*

Undirected graphs correspond to symmetric binary relations. Observe that graphs of rectangular symmetric binary relations are precisely those specified in Theorem 7. Indeed, a complete graph with all loops present correspond to the total binary relation which is rectangular, and if ϱ is a relation corresponding to a complete bipartite graph and $(a, c), (a, d), (b, c) \in \varrho$ then b, d are in different parts and therefore the edge (b, d) presents.

Conversely, let H be a connected graph such that ϱ_H is rectangular. Suppose first that H has a loop at vertex a and (a, b) is an edge. Then $(a, a), (a, b), (b, a) \in \varrho_H$ and, by the rectangularity, we get $(b, b) \in \varrho_H$. Therefore, all loops in H are present. Furthermore, if $(a, b), (b, c)$ is a path then $(b, a), (b, c), (a, a) \in \varrho_H$ implies $(a, c) \in \varrho_H$, and we conclude that H is complete. If H has no loops then notice that, for any path $(a, b), (b, c), (c, d)$, the edge (a, d) also presents, because $(a, b), (c, b), (c, d) \in \varrho_H$. Therefore if H contains a cycle of the odd length then it also has a loop, a contradiction. Thus H is bipartite, and the completeness follows straightforwardly from the rectangularity.

Making use of Theorem 5, one may easily derive the #P-completeness part of Theorem 7. The tractability part of this theorem is fairly simple.

Conjecture 1 being specified to the case of $\#H$-COLORING enables us to conjecture a criteria for digraphs which give rise to tractable $\#H$-COLORING problems.

Conjecture 2 *The $\#H$-COLORING problem for a digraph H is tractable if and only if ϱ_H is invariant under a Mal'tsev operation.*

Unfortunately, it is not easy to express the criteria in graph-theoretic terms. However one necessary condition can be identified. A digraph is said to be *N-free* if it does not contain a subgraph shown on Fig. 2.

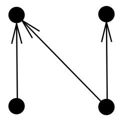

Figure 2.

Lemma 1 *If a digraph H is not N-free, then the $\#H$-COLORING problem is #P-complete.*

In fact, the effect of the property of rectangularity is that it excludes the configuration shown on Fig. 2.

Theorem 7 confirms Conjecture 2 for undirected graphs. For some classes of digraphs this conjecture is also true.

Proposition 4 *Let H be an oriented cycle. Then the $\#H$-COLORING problem is tractable if and only if H is one of C_n, C_n^1, C_n^2 (see Fig. 3). Otherwise it is #P-complete.*

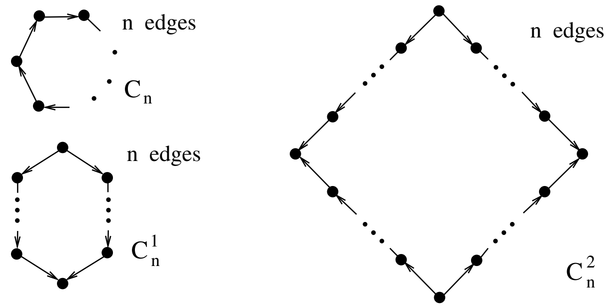

Figure 3.

Proof (Hardness): Let $H = (V; E)$ be an oriented cycle. We prove that if there is a Mal'tsev polymorphism of ϱ_H then H is of the form specified in the theorem.

Clearly, if H is not a directed cycle then H contains no directed cycles. So, we shall assume that H has no directed

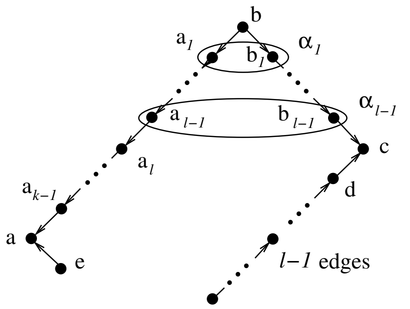

Figure 4.

cycles and N-free. By α_H we denote the relation $\varrho_H^{-1} \circ \varrho_H$, an equivalence relation on the set V^+ of vertices of non-zero indegree. For any $v, w \in V$ with $(v, w) \in \alpha_H$, there is $u \in V$ such that $(u, v), (u, w) \in \varrho_H$. Moreover, since H is N-free, if $(u', v) \in \varrho_H$, then $(u', w) \in \varrho_H$. Let us denote by H_α the homomorphic image of the subgraph spanned on V^+ with the kernel α_H.

If H contains no vertex of zero indegree then $H = C_{|V|}$. Otherwise, H may contain several directed paths. The maximal ones are the paths from a vertex of zero indegree to the nearest vertex of zero outdegree.

Take a vertex b of zero indegree and maximal directed paths $b = a_0, a_1, \ldots, a_k = a$ and $b = b_0, b_1, \ldots, b_l = c$ originated at b. Suppose that $l \leq k$, and $b_0, \ldots, b_l$ is the shortest maximal path. Then a, c are the vertices of zero outdegree. Let us denote $H_0 = H$, α_0 is $=_V$, and $H_i = (H_{i-1})_{\alpha_i}$, $\alpha_i = \alpha_{H_{i-1}}$. Clearly, we may treat the α_i as relations on V. Then $(u, v) \in \alpha_i$ if and only if there exist $w \in V$ and directed paths of the length i connecting w with u and v.

Therefore, $(a_0, b_0) \in \alpha_0, (a_1, b_1) \in \alpha_1, \ldots, (a_{l-1}, b_{l-1}) \in \alpha_{l-1}$ (see Fig. 4). Notice that, since l is the minimal length of a maximal path, the pair (d, d) is in α_{l-1}. Furthermore,

$$m\left(\begin{pmatrix} a_{l-1} \\ a_l \end{pmatrix}, \begin{pmatrix} b_{l-1} \\ c \end{pmatrix}, \begin{pmatrix} d \\ c \end{pmatrix}\right) \in \varrho_H,$$

which means $m(a_{l-1}, b_{l-1}, d) = g \in \{a_{l-1}, \}$. Then, as m preserves α_{l-1}, we have

$$m\left(\begin{pmatrix} a_{l-1} \\ a_{l-1} \end{pmatrix}, \begin{pmatrix} b_{l-1} \\ a_{l-1} \end{pmatrix}, \begin{pmatrix} d \\ d \end{pmatrix}\right) = \begin{pmatrix} g \\ d \end{pmatrix} \in \alpha_{l-1}.$$

Therefore, there is a vertex v of zero indegree and paths of legth $l-1$ from v to g and d. If $g = a_{l-1}$ then $v = b, c = a$ and $H = C_l^1$. If $g = e \neq a_{l-1}$ then $H = C_l^2$. $\square$

Algorithms solving $\#H$-COLORING problem for the graphs C_n, C_n^1, C_n^2 can be found in [7].

4.4. The List $\#H$-Coloring problem

In the LIST $\#H$-COLORING problem we are given an input (directed) graph G, and for every vertex g of G, a *list* $L(g)$, that is a subset of vertices of H. The objective is to find the number of homomorphisms $\varphi: G \to H$ such that $\varphi(g) \in L(g)$ for every vertex g of G. It is not hard to see that this problem is equivalent to $\#\mathrm{CSP}(\Gamma_H)$ where $\Gamma_H = \{\varrho_H\} \cup \{\sigma \mid \sigma$ is a unary predicate over the vertex set of $H\}$.

In the case of undirected graphs a dichotomy theorem was obtained independently in [18] and [26]. The criteria happened to be the same as that for $\#H$-COLORING.

Theorem 8 ([18, 26]) *If every connected component of an undirected graph H is either an isolated vertex, or a complete graph with all loops present, or a complete unlooped bipartite graph, then* LIST $\#H$-COLORING *is tractable. Otherwise, it is #P-complete.*

If LIST $\#H$-COLORING is tractable then a Mal'tsev polymorphism m of the constraint language Γ_H preserves every unary predicate. As is easily seen this is equivalent to the condition $m(x, y, z) \in \{x, y, z\}$ for any x, y, z. An operation satisfying this condition is called *conservative*. Therefore Conjecture 1 in the case of LIST $\#H$-COLORING problem looks as follows.

Conjecture 3 *The* LIST $\#H$-COLORING *problem is tractable if and only if ϱ_H is invariant with respect to a conservative Mal'tsev operation. Otherwise it is #P-complete.*

Remarkably, in contrast to the case of undirected graphs, the classes of digraph specified in Conjectures 2 and 3 are not equal. For example, consider the digraph $H = C_2^2$ (Fig. 5). By Proposition 4, $\#H$-COLORING problem can be solved in polynomial time, while ϱ_H has no conservative Mal'tsev polymorphism, which means that LIST $\#H$-COLORING is #P-complete. Indeed, let m be a conservative polymorphism of ϱ_H. The relations $\delta_1 = \varrho_H \circ \varrho_H^{-1}$ and $\delta_2 = \varrho_H^{-1} \circ \varrho_H$ are equivalence relations on the sets $\{a, b, c, d, g, h\}, \{a, b, c, d, e, f\}$, respectively, with classes $\{a, b\}, \{c, d\}, \{g\}, \{h\}$ and $\{a, d\}, \{b, c\}, \{e\}, \{f\}$, respectively. Since m preserves δ_1, δ_2, we have

$$m\left(\begin{pmatrix} a \\ b \end{pmatrix}, \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} c \\ c \end{pmatrix}\right) = \begin{pmatrix} c \\ c \end{pmatrix},$$
$$m\left(\begin{pmatrix} a \\ a \end{pmatrix}, \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} c \\ b \end{pmatrix}\right) = \begin{pmatrix} a \\ a \end{pmatrix}.$$

This means $m(a, b, c) = c$ and $m(a, b, c) = a$, a contradiction.

4.5. Uniform constraint languages.

In [6], it was shown that the decision constraint satisfaction problem arising from a constraint language invariant

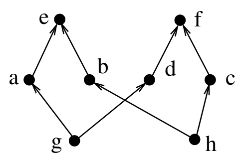

Figure 5.

with respect to a Mal'tsev operation is tractable. Moreover, the algorithm presented in that paper allows one not only determine whether there is a solution to a problem instance, and not only find a solution, but to find a basis of the solution space. In some cases we may find the number of solutions by making use of this basis.

A constraint language Γ is said to be *uniform* if the blocks of any equivalence relation from $\langle\Gamma\rangle$ are of the same size. Clearly, every Boolean constraint language is uniform. Further examples of uniform constraint languages are provided by 'group constraints' [21, 22], and also by constraints studied in [15]. For more examples see also [7].

Theorem 9 *A uniform constraint language invariant with respect to a Mal'tsev operation is #-tractable.*

5. Algebraic Structure of #CSP

In this section we further develop algebraic machinery for the counting constraint satisfaction problem. Our aimes are, first, to make it possible to use the most advanced and powerful results from universal algebra in the study of #CSP, and, second, to develop techniques sufficient to prove Theorem 5.

5.1. Algebraic constructions and #CSP

In this section we give basic algebraic definitions. We also introduce the notion of "#-tractable algebra" and show how it relates to the complexity of problem classes of the form $\#\mathrm{CSP}(\Gamma)$. In our algebraic definitions we follow [12, 38].

A (*universal*) *algebra* is an ordered pair $\mathbb{A} = (A, F)$ where A is a non-empty set and F is a family of finitary operations on A. The set A is called the *universe* of $\mathbb{A}$, the operations from F are called *basic*. An algebra with a finite universe is referred to as a *finite algebra*.

Any constraint language Γ on a set A can be converted into an algebra $\mathbb{A}_\Gamma = (A; \mathsf{Pol}\ \Gamma)$, and vice versa, for any algebra $(A; F)$, there is a corresponding constraint language $\mathsf{Inv}\ F$. By Theorem 3, if $\mathsf{Pol}\ \Gamma_1 = \mathsf{Pol}\ \Gamma_2$ or, equivalently, $\mathbb{A}_{\Gamma_1} = \mathbb{A}_{\Gamma_2}$, then Γ_1, Γ_2 are #-tractable or #P-complete simultaneously. Therefore all the problem classes can be parametrized by finite algebras so that classes with the same parameter have the same complexity. We make the following definition.

Definition 4 *An algebra* $\mathbb{A} = (A; F)$ *is said to be* #-tractable (#P-complete) *if the constraint language* $\mathsf{Inv}\ F$ *is #-tractable (#P-complete).*

By $\#\mathrm{CSP}(\mathbb{A})$ *we denote the problem class* $\#\mathrm{CSP}(\mathsf{Inv}\ F)$.

Making use of this definition we reformulate Problem 1.

Problem 2 (classification problem) *Characterise #-tractable and #P-complete finite algebras.*

Theorem 4 provides the first step towards a solution of this problem, because it yields a complete classification of two-element algebras with respect to #-tractability. An operation f is said to be a *term* operation of an algebra $\mathbb{A} = (A; F)$ if $f \in \mathsf{Pol}\ \mathsf{Inv}\ F$. It is straightforward that, if a relation ϱ is invariant under F then it is also invariant under every term operation of $\mathbb{A}$.

Theorem 10 ([13, 14]) *A two-element algebra* $\mathbb{A} = (\{0, 1\}, F)$ *is #-tractable if and only if* $x - y + z$ *is a term operation of* $\mathbb{A}$. *Otherwise* $\mathbb{A}$ *is #P-complete.*

The main idea of the algebraic approach is to use some properties of an algebra in order to determine the complexity of the associated #CSP. To identify these properties, some connections between the complexity of an algebra and standard algebraic constructions will be very helpful.

Definition 5 • *Let* $\mathbb{A} = (A, F)$ *be an algebra. The* n-th direct power *of* $\mathbb{A}$ *is the algebra* $\mathbb{A}^n = (A^n, F)$ *where we treat each (*m*-ary) operation* $f \in F$ *as acting on* A^n.

• *Let* $\mathbb{A} = (A, F)$ *be an algebra, and* B *a subset of* A *such that, for any (*n*-ary)* $f \in F$, *and for any* $b_1, \ldots, b_n \in B$, *we have* $f(b_1, \ldots, b_n) \in B$. *Then the algebra* $\mathbb{B} = (B, F\big|_B)$, *where* $F\big|_B$ *consists of restrictions of operations* $f \in F$ *to* B, *is called a* subalgebra *of* $\mathbb{A}$.

• *Let* $\mathbb{A}_1 = (A_1, F_1)$ *and* $\mathbb{A}_2 = (A_2, F_2)$ *such that* $F_1 = \{f_i^1 \mid i \in I\}$, $F_2 = \{f_i^2 \mid i \in I\}$, *and* f_i^1, f_i^2 *are of the same arity,* $i \in I$. *A mapping* $\varphi : A_1 \to A_2$ *is called a* homomorphism *from* $\mathbb{A}_1$ *to* $\mathbb{A}_2$ *if* $\varphi f_i^1(a_1, \ldots, a_{n_i}) = f_i^2(\varphi(a_1), \ldots, \varphi(a_{n_i}))$ *holds for all* $i \in I$ *and all* $a_1, \ldots, a_{n_i} \in A_1$. *If the mapping* φ *is onto then* $\mathbb{A}_2$ *is said to be a* homomorphic image *of* $\mathbb{A}_1$.

A property of algebras such that if an algebra enjoys the property then any its subalgebra, homomorphic image, and direct power also enjoys it, is said to be *hereditary*. Universal algebra mostly deals with hereditary properties [29, 38]. Therefore, the next theorem allows us to apply the most powerful methods of modern algebra to the study of the complexity of the counting constraint satisfaction problem.

Theorem 11 *Let $\mathbb{A}$ be a finite algebra. Then*

(i) if $\mathbb{A}$ is #-tractable then so is every subalgebra, homomorphic image, and direct power of $\mathbb{A}$.

(ii) if $\mathbb{A}$ has an #P-complete subalgebra, homomorphic image, or direct power, then $\mathbb{A}$ is #P-complete itself.

An operation f is said to be *idempotent* if the identity $f(x, \ldots, x) = x$ holds. Algebras whose basic (and therefore term) operations are idempotent posess many useful properties that will assist in our investigation. The *full idempotent reduct* of an algebra $\mathbb{A} = (A; F)$ is the algebra $\mathsf{Id}(\mathbb{A}) = (A; F_{\mathrm{id}})$ where F_{id} consists of all idempotent term operations of $\mathbb{A}$. There is another way to characterize F_{id}: $F_{\mathrm{id}} = \mathsf{Pol}\ (\mathsf{Inv}\ F \cup \{\{(a)\} \mid a \in A\})$ where $\{(a)\}$ means a unary relation containing only one tuple.

Theorem 12 *A finite algebra $\mathbb{A}$ is #-tractable (#P-complete) if and only if so is $\mathsf{Id}(\mathbb{A})$.*

5.2. Hard cases of #CSP

As we shall see, binary reflexive, but not symmetric relations play the central role in solving Problems 1,2.

Theorem 13 *If ϱ is a binary reflexive but not symmetric relation on a finite set then $\#\mathrm{CSP}(\{\varrho\})$ is #P-complete.*

Theorems 11,13, and results from [29] provide a link between the complexity of #CSP and Mal'tsev operations. The next statement follows from Theorem 9.13 of [29].

Theorem 14 ([29]) *For a finite algebra $\mathbb{A}$ the following conditions are equivalent.*

1. $\mathbb{A}$ has not a Mal'tsev term operation.

2. There is $\mathbb{B} = (B; F)$, a homomorphic image of a subalgebra of a direct power of $\mathbb{A}$, such that $\mathsf{Inv}\ F$ contains a binary reflexive but not symmetric relation.

By Theorem 13, the algebra $\mathbb{B}$ from Theorem 14(2) is #P-complete. Furthermore, Theorem 11 implies that $\mathbb{A}$ is also #P-complete.

Corollary 3 *If $\mathbb{A}$ is a finite algebra that has no Mal'tsev term operation then $\mathbb{A}$ is #P-complete.*

Finally, Theorem 5 is just Corollary 3 expressed in terms of constraint languages.

6. Conclusion

We have shown that the algebraic approach developed originally to deal with the decision constraint satisfaction problem is applicable, in some aspects even more efficiently, to the counting constraint satisfaction problem. On the one hand, this has allowed us, by making use of advanced and deep algebraic results (Theorem 14), to obtain a strong necessary condition on tractable cases of the counting constraint satisfaction problem (Theorem 5), and to pose a plausible conjecture about a possible form of a criteria for such cases. On the other hand, earlier results on algorithmic and structural aspects of Mal'tsev algebras make it possible to design polynomial time algorithms solving many of the counting constraint satisfaction problems satisfying the criteria. In particular, the criteria appear to be true for problems over small domains.

The algebraic approach also appears to be very helpful in systematization of existing complexity results on #CSP by identifying a common property of classes solvable in polynomial time, and providing a strong guidance for further research.

The applicability of the algorithm solving Mal'tsev decision constraint satisfaction problems to #CSP is limited. However, as Theorem 6 shows, those limitations can be overcome by using of structural properties of relations invariant under a Mal'tsev operation. We strongly believe that future developments in algebraic theory related Mal'tsev operations will lead to a complete solution of Problem 1.

References

[1] J. Bang-Jensen and P. Hell. The effect of two cycles on the complexity of colourings by directed graphs. *Discrete Applied Math.*, 26(1):1–23, 1990.

[2] J. Bang-Jensen, P. Hell, and G. MacGillivray. Hereditarily hard H-coloring problems. *Discrete Math.*, 138:75–92, 1995.

[3] G. Brightwell and P. Winkler. Graph homomorphisms and phase transitions. *Journal of Combinatorial Theory, Ser. B*, 77:221–262, 1999.

[4] R. Bubley, M. Dyer, C. Greenhill, and M. Jerrum. On approximately counting colourings of small degree graphs. *SIAM Journal of Computing*, 29:387–400, 1999.

[5] A. Bulatov. Three-element Mal'tsev algebras. Submitted to FOCS'03.

[6] A. Bulatov. Mal'tsev constraints are tractable. Technical Report PRG-RR-02-05, Computing Laboratory, University of Oxford, Oxford, UK, 2002.

[7] A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. Technical Report PRG-RR-03-12, Computing Laboratory, University of Oxford, Oxford, UK, 2003.

[8] A. Bulatov and P. Jeavons. Algebraic approach to multi-sorted constraints. Technical Report PRG-RR-01-18, Computing Laboratory, University of Oxford, Oxford, UK, 2001.

[9] A. Bulatov and P. Jeavons. Algebraic structures in combinatorial problems. Technical Report MATH-AL-4-2001, Technische universität Dresden, Dresden, Germany, 2001.

[10] A. Bulatov, A. Krokhin, and P. Jeavons. Constraint satisfaction problems and finite algebras. In *Proceedings of 27th International Colloquium on Automata, Languages and Programming—ICALP'00*, volume 1853 of *Lecture Notes in Computer Science*, pages 272–282. Springer-Verlag, 2000.

[11] R. Burton and J. Steif. Nonuniqueness of mesures of maximal entropy for subshifts of finite type. *Ergodic Theory and Dynamical Systems*, 14:213–236, 1994.

[12] P. Cohn. *Universal Algebra*. Harper & Row, 1965.

[13] N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125(1):1–12, 1996.

[14] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 2001.

[15] V. Dalmau. A new tractable class of constraint satisfaction problems. In *Proceedings 6th International Symposium on Artificial Intelligence and Mathematics*, 2000.

[16] J. Diaz, M. Serna, and D. Thilikos. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science, DIMACS/DIMATIA Workshop on Graphs, Morphism and Statistical Physics. American Mathematical Society*, chapter Recent results on parameterized H-coloring. To appear.

[17] J. Diaz, M. Serna, and D. Thilikos. The complexity of restrictive H-coloring. *To appear in Discrete Applied Mathematics*.

[18] J. Diaz, M. Serna, and D. Thilikos. Counting list H-colorings and variants. Technical Report LSI-01-27-R, Departament LSI, Universitat Politècnica de Catalunya, 2001.

[19] J. Diaz, M. Serna, and D. Thilikos. Counting h-colorings of partial k-trees. *Theoretical Computer Science*, 281:291–309, 2002.

[20] M. Dyer and C. Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17:260–289, 2000.

[21] T. Feder. Constraint satisfaction on finite groups with near subgroups. Submitted to Journal of Algebra.

[22] T. Feder and M. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM Journal of Computing*, 28:57–104, 1998.

[23] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA., 1979.

[24] C. Greenhill. The complexity of counting colourings and independent sets in sparse graphs and hypergraphs. *Computational Complexity*, 9:52–73, 2000.

[25] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *Proceedings of the 33rd Annual ACM Simposium on Theory of Computing*, pages 657–666, Hersonissos, Crete, Greece, July 2001. ACM Press.

[26] P. Hell and J. Nešetřil. Counting list homomorphisms for graphs with bounded degrees. *Discrete Mathematics*. to appear.

[27] P. Hell and J. Nešetřil. On the complexity of H-coloring. *Journal of Combinatorial Theory, Ser.B*, 48:92–110, 1990.

[28] P. Hell, J. Nešetřil, and X. Zhu. Duality and polynomial testing of tree homomorphisms. *Trans. of the AMS*, 348(4):1281–1297, 1996.

[29] D. Hobby and R. McKenzie. *The Structure of Finite Algebras*, volume 76 of *Contemporary Mathematics*. American Mathematical Society, Providence, R.I., 1988.

[30] H. Hunt III, M. Marathe, V. Radhakrishnan, and R. Stearns. The complexity of planar counting problems. *SIAM Journal on Computing*, 27:1142–1167, 1998.

[31] P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200:185–204, 1998.

[32] P. Jeavons, D. Cohen, and M. Cooper. Constraints, consistency and closure. *Artificial Intelligence*, 101(1-2):251–265, 1998.

[33] P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44:527–548, 1997.

[34] P. Kolaitis and M. Vardi. Conjunctive-query containment and constraint satisfaction. *J. Comput. Syst. Sci.*, 61:302–332, 2000.

[35] J. Lebowitz and G. Gallavotti. Phase transitions in binary lattice gases. *Journal of Math. Physics*, 12:1129–1133, 1971.

[36] L. Levin. Universal enumeration problems. *Problems on Information Transmission*, 9:265–266, 1973.

[37] N. Linial. Hard enumeration problems in geometry and combinatorics. *SIAM Journal on Algebraic and Discrete Methods*, 7(2):331–335, 1986.

[38] R. McKenzie, G. McNulty, and W. Taylor. *Algebras, Lattices and Varieties*, volume I. Wadsworth and Brooks, California, 1987.

[39] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 1974.

[40] P. Orponen. Dempster's rule of combination is #-complete. *Artificial Intelligence*, 44:245–253, 1990.

[41] N. Pippenger. *Theories of Computability*. Cambridge University Press, Cambridge, 1997.

[42] R. Pöschel. Galois connection for operations and relations. Technical Report MATH-AL-8-2001, Technische Universität Dresden, Germany, 2001.

[43] R. Pöschel and L. Kalužnin. *Funktionen- und Relationenalgebren*. DVW, Berlin, 1979.

[44] E. Post. *The two-valued iterative systems of mathematical logic*, volume 5 of *Annals Mathematical Studies*. Princeton University Press, 1941.

[45] J. Provan and M. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing*, 12(4):777–788, 1983.

[46] D. Roth. On the hardness of approximate reasonning. *Artificial Intelligence*, 82:273–302, 1996.

[47] T. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th ACM Symposium on Theory of Computing (STOC'78)*, pages 216–226, 1978.

[48] A. Szendrei. *Clones in Universal Algebra*, volume 99 of *Seminaires de Mathematiques Superieures*. Université de Móntreal, 1986.

[49] L. Valiant. The complexity of computing the permanent. *Theoretical Computing Science*, 8:189–201, 1979.

[50] L. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

[51] X. Zhu. A polynomial algorithm for homomorphisms to oriented cycles. *Journal of Algorithms*, 19:333–345, 1995.