Towards a Dichotomy Theorem for the Counting Constraint Satisfaction Problem

Andrei A. Bulatov [a,*] Víctor Dalmau [b]

[a] *School of Computing Science*
Simon Fraser University, Burnaby, Canada

[b] *Department de Tecnologia*
Universitat Pompeu Fabra, Barcelona, Spain

Abstract

The Counting Constraint Satisfaction Problem (#CSP) can be expressed as follows: given a set of variables, a set of values that can be taken by the variables, and a set of constraints specifying some restrictions on the values that can be taken simultaneously by some variables, determine the number of assignments of values to variables that satisfy all the constraints. The #CSP provides a general framework for numerous counting combinatorial problems including counting satisfying assignments to a propositional formula, counting graph homomorphisms, graph reliability and many others. This problem can be parametrized by the set of relations that may appear in a constraint. In this paper we start a systematic study of subclasses of the #CSP restricted in this way. The ultimate goal of this investigation is to distinguish those restricted subclasses of the #CSP which are 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 constraint satisfaction problem. Then we prove that if a subclass of the #CSP is solvable in polynomial time, then constraints allowed by the class satisfy some very restrictive condition: they need 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 many existing complexity results for particular cases of the #CSP, including the dichotomy results for the problem of counting graph homomorphisms, and it allows us to obtain new results.

Key words: Constraint Satisfaction Problem, counting problems, complexity

---

* Corresponding author. Address: School of Computing Science, Simon Fraser University, 8888 University Drive, Burnaby, V5A 1S6, Canada. E-mail: abulatov@cs.sfu.ca

Email addresses: e-mail: abulatov@cs.sfu.ca (Andrei A. Bulatov), *e-mail:*

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 counting combinatorial problems that arise frequently in a wide range of areas from logic, graph theory, and artificial intelligence [4,13,21,26,33,41,45,51,52,55,56], to statistical physics [3,11,39].

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 the #CSP. Since the pionnering papers of Valiant [55,56] the computational complexity of counting satisfying assignments to propositional formulas of various types [13,41,51,52,55,56] has been intensively investigated. 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. In [13], Creignou and Hermann obtained a dichotomy theorem for #SAT, similar to that of Schaefer [53] for (decision) SAT.

The formalism of constraint networks introduced by Montanari [43] 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 where a constraint is a pair given by a list of variables, called the *scope*, and a relation indicating the valid combinations of values for the variables in the scope. The problem of deciding whether there exists a solution to a constraint network, i.e., an assignment of values to variables satisfying all the constraints, is known as the constraint satisfaction problem (CSP). This problem received 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 solutions to a constraint network. This problem can also be reformulated as (1) the problem of finding the number of models of a conjunctive formula, 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 [24,37].

Further examples of combinatorial problems expressible in a natural way in

victor.dalmau@tecn.upf.es (Víctor Dalmau).

#CSP terms include problems from propositional logic [13,52], classical combinatorial problems such as #CLIQUE, GRAPH RELIABILITY, ANTICHAIN, PERMANENT [41,51,55,56], counting graph homomorphisms, and many others [4,21,26,33].

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. Recently, Dyer and Greenhill [21] proved that, for every undirected graph H, its associated #H-COLORING problem is either in FP or #P-complete (even when restricted to graphs of bounded degree) and they have also provided a complete characterization of the tractable problems. This result has been extended to the counting LIST #H-COLORING problem [19], 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 [17,18].

In general, there are two most usual ways to parametrize the constraint satisfaction problem and its variants: by restricting either the scopes or the relations that may appear in a constraint. It is perhaps more usual to think of it as of restricting the left side or the right side of the homomorphism formulation of the CSP. Constraint problems with restrictions on both sides have also been investigated, especially in graph theory. However, studying such problems in general seems to be very challenging.

Left side restrictions of the #CSP are studied in [16]. In this paper we study restrictions on the right side. Most of the previous results on the #CSP, such as the dichotomy theorem for #SAT [13], or the work in #H-COLORING, fit in this framework. More precisely, we embark on a systematic study of the computational complexity of the subclasses of the #CSP parametrized by the set of allowed constraint relations. 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 the #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 [8–10,34–36], and we expect that it will also be useful for the study of counting problems.

This algebraic approach allows us to identify a common property of all tractable restricted #CSPs: 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 many existing complexity results on counting CSP and its particular cases including different types of #H-Coloring [17–19,21], and constraints considered in [15,23,24]. It is worth noticing that in all these examples the frontier between tractability and intractability is given by invariance under Mal'tsev operations. It is therefore very natural to conjecture that the existence of such a polymorphism is a sufficient condition for tractability, and so we did in the conference version of this paper [5]. Now we know that the conjecture is false. We finish the present paper by presenting a concrete Mal'tsev operation that gives rise to a #P-complete problem.

The paper is organized as follows. In Section 2, we give basic definitions and examples and also formulate the main research problem. The primary goal of Section 3 is to show that, similar to the case of the decision CSP, polymorphisms of constraint languages capture the complexity of restricted #CSPs. In Section 4, we develop the algebraic approach by introducing the notion of a "#-tractable algebra" and showing that the usual algebraic constructions preserve the tractability of an algebra. Then we use these results to show that the presence of a Mal'tsev polymorphism is a necessary condition for a problem to be solvable in polynomial time. Then, in Section 5, we use this necessary condition to obtain short and simple proofs for many existing results and also characterize the #H-Coloring problems solvable in polynomial time, when H is an oriented cycle. Finally, in Section 6, we discuss the possible form of a general criterion for polynomial time solvability of #CSPs.

We should note that in the conference version of this paper [5] we mistakenly claimed that we obtained a dichotomy theorem for counting CSPs on a 3-element set. That result is incorrect, and the problem of characterizing tractable counting CSPs on a 3-element set remains open.

2 The Counting Constraint Satisfaction Problem

2.1 Definitions and Examples

Let A be a finite set. An r-ary, $r \geq 1$, *relation* R on A is any subset of A^r.

Definition 1 *The* counting constraint satisfaction problem (#CSP) *is the combinatorial function problem with*

Instance: *a triple* $(V; A; \mathcal{C})$ *where* V *is a finite set of* variables, *A is a finite* *set of values* [domain], $\mathcal{C}$ *is a finite set of* constraints. *Each constraint* $C \in \mathcal{C}$ *is a pair* $\langle s, \varrho \rangle$, *where*

- $s = (v_1, \ldots, v_{m_C})$ *is a tuple of variables of length* m_C, *called the* constraint scope*;*
- ϱ *is an* m_C*-ary relation on* A, *called the* constraint relation.

Objective: *compute the number of* solutions, *i.e. functions* φ, *from* V *to* A, *such that, for each constraint* $\langle s, \varrho \rangle \in \mathcal{C}$, *with* $s = (v_1, \ldots, v_m)$, *the tuple* $(\varphi(v_1), \ldots, \varphi(v_m))$ *belongs to* ϱ.

Example 1 ($\#k$**-SAT, [13,14,55,56])** An instance of the $\#k$-SAT problem is specified by giving a propositional logic formula in k-CNF, and asking how many assignments satisfy it.

Suppose that $\Phi = F_1 \wedge \cdots \wedge F_n$ is such a formula, where the F_i are clauses with k literals. The satisfiability question for Φ can be expressed as the constraint satisfaction problem instance $(V; \{0,1\}; \mathcal{C})$, where V is the set of all variables appearing in the clauses F_i, and $\mathcal{C}$ is the set of constraints $\{\langle s_1, \varrho_1 \rangle, \ldots, \langle s_n, \varrho_n \rangle\}$, where each constraint $\langle s_l, \varrho_l \rangle$, $l = 1, \ldots, n$ is constructed as follows:

- $s_l = (x_1^l, \ldots, x_k^l)$ where $x_1^l, \ldots, x_k^l$ are the variables appearing in clause F_l;
- $\varrho_l = \{0,1\}^k \setminus \{(a_1, \ldots, a_k)\}$ where $a_i = 1$ if x_i^l is negated in F_l and $a_i = 0$ otherwise (i.e., ϱ_l contains exactly those k-tuples that make F_l true).

The solutions of this instance are exactly the assignments which make the formula Φ true.

It is well known [55,56] that $\#k$-SAT is #P-complete for $k \geq 2$. □

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.

Let $v_1, \ldots, v_k$ be variables. A first order formula with free variables $v_1, \ldots, v_k$ is said to be *conjunctive* if it is a finite conjunction of clauses $F_1 \wedge \cdots \wedge F_n$ such that each clause, F_i, is an *atomic formula* of the form $\varrho(v_{i_1}, \ldots, v_{i_r})$ where $v_{i_1}, \ldots, v_{i_r}$ are variables in $\{v_1, \ldots, v_k\}$ and ϱ is a predicate on A. An atomic formula $\varrho(v_1, ..., v_r)$ is *satisfied* by a variable assignment $\varphi : V \to A$ if and only if $(\varphi(v_1), ..., \varphi(v_r)) \in \varrho$, and a conjunctive formula is *satisfiable* if and only if there exists an assignment satisfying all its clauses. Sometimes another formulation of the #CSP, given in terms of conjunctive formulas, is more convenient.

Definition 2 *Let* A *be a finite set. An* instance *of the* #CSP *is a conjunctive formula* $F_1 \wedge \ldots \wedge F_n$ *where each* F_i *is an atomic formula. The* objective *is to* *find the number of satisfying assignments to the formula.*

Example 1 (continued) In the #k-SAT problem, a CNF can obviously be viewed as a conjunctive formula with predicate symbols interpreted as the corresponding Boolean predicates. □

Note: In this paper we use the notion of completeness based on Turing reduction, as in [41,51,55,56], rather than parsimonious reduction, as in [46]. In fact, this notion of reduction was used also in [14], as its #P-completeness results relies upon the results of [41,51,55,56]. Thus, throughout the paper 'reduction' always means 'Turing reduction'.

The general #CSP is known to be #P-complete, as follows from [56] and the example above. However, some restricted problems have been shown to be computable in polynomial time. One of the most natural and useful ways to restrict the CSP is to impose restrictions on the allowed constraint relations.

A *constraint language* on a set A is just a set of relations on A.

Definition 3 *For any constraint language* Γ*, the* $\#\mathrm{CSP}(\Gamma)$ *is defined to be the class of counting problems with:*

INSTANCE *: A constraint satisfaction problem instance* $\mathcal{P}$*, in which all constraint relations are elements of* Γ*.*
OBJECTIVE *: compute the number of solutions to* $\mathcal{P}$*.*

If the #CSP is defined in the logic form, that is as in Definition 2, then $\#\mathrm{CSP}(\Gamma)$ is defined similarly: the instances of $\#\mathrm{CSP}(\Gamma)$ are restricted to those instances of the #CSP which include predicates from Γ only.

Example 1 (continued) If we define $\Gamma_{k\text{-SAT}}$ to be the constraint language on $\{0,1\}$ consisting of all relations expressible by k-clauses, then any instance of #k-SAT can be expressed as an instance of $\#\mathrm{CSP}(\Gamma_{k\text{-SAT}})$ and vice versa. In other words, #k-SAT is equivalent to $\#\mathrm{CSP}(\Gamma_{k\text{-SAT}})$. □

Example 2 (ANTICHAIN**, [51])** In the problem ANTICHAIN we are given a finite poset $(P;\le)$, 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 relation that encodes the natural order on $A = \{0,1\}$, that is, $\varrho_\prec = \{(0,0),(0,1),(1,1)\}$. To each element $a \in P$, we assign a variable x_a. We shall denote by V the set of all such variables. Then the $\#\mathrm{CSP}(\{\varrho_\prec\})$ instance $\mathcal{P} = (V;\{0,1\};\{\langle(x_a,x_b),\varrho_\prec\rangle \mid a \le b\})$ is equivalent to the original ANTICHAIN instance.

To show this, notice that every solution φ to $\mathcal{P}$ satisfies the following condition: if $\varphi(x_a) = 1$ and $a \le b$ 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 $a \le b$ 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 $\{q \in P \mid \text{there is } p \in H \text{ such that } p \le q\}$ is a filter. Conversely, for a filter $F \subseteq P$, the set $\{q \in F \mid \text{if } p \in F \text{ and } p \le q \text{ then } p = q\}$ of *minimal* elements from F forms an antichain. Thus the solutions of $\mathcal{P}$ one-to-one correspond to the filters of P, and consequently, to the antichains of P.

On the other hand, any #CSP($\{\varrho_\prec\}$) instance is reducible to an ANTICHAIN instance, though not so straightforwardly. (The set of variables of the instance can be turned into a digraph, whose edges are the constraint scopes. Then the required ANTICHAIN instance is the poset of the strongly connected components of this digraph.) Thus ANTICHAIN is equivalent to #CSP($\{\varrho_\prec\}$). □

Example 3 (#H-COLORING, [21,30,40]) 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.

For every (directed) graph H we shall denote by V_H its set of nodes and by ϱ_H its set of edges.

Then every instance $G = (V_G, \varrho_G)$ of the #H-COLORING problem corresponds to the instance $\mathcal{P} = (V_G; V_H; \mathcal{C})$ of #CSP($\{\varrho_H\}$) constructed in the following way: The set of variables V_G of $\mathcal{P}$ is the set of nodes of G, the domain V_H of $\mathcal{P}$ is the set of nodes of H. Finally, for every edge (a, b) in ϱ_G, $\mathcal{C}$ contains the constraint $\langle (a, b), \varrho_H \rangle$. Therefore every homomorphism from G to H corresponds to a solution of $\mathcal{P}$.

Note that this framework does not allow us to study #H-COLORING problems with input graphs of restricted types, such as graphs of bounded degree, planar, and so on. To express problems of this type we need to impose restrictions on both relational structures involved in a #CSP. □

Definition 4 *A constraint language* Γ *is called* #-tractable *if for any finite* $\Gamma' \subseteq \Gamma$ *the problem* #CSP(Γ') *is solvable in polynomial time.*

A constraint language Γ *is called* #P-complete *if* #CSP(Γ') *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 in the Boolean case, that is when a constraint language is on a 2-element set.

Theorem 1 ([13,14]) *A Boolean constraint language Γ is #-tractable if and only if every relation from Γ is the solution space of a system of linear equations over a 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 which predicates can be added to a #-tractable constraint language so that the obtained language remains #-tractable. This idea was 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 in the logic form.

Let Γ be a (possibly infinite) constraint language on a finite domain A. The relation ϱ *defined* by a conjunctive formula $\Phi(v_1, \ldots, v_k)$ with free variables $v_1, \ldots, v_k$ is the k-ary relation that contains the tuple $(\varphi(v_1), \ldots, \varphi(v_k))$ for each satisfying assignment φ to Φ. If Φ involves only predicates from Γ we say that ϱ is *definable* by a conjunctive formula over Γ. Let $=_A$ denote the relation of equality on the set A.

Proposition 1 *Let Γ be a constraint language on a finite set A. If ϱ is definable by a conjunctive formula over Γ, then $\#\mathrm{CSP}(\Gamma \cup \{\varrho\})$ and $\#\mathrm{CSP}(\Gamma \cup \{=_A\})$ are reducible to $\#\mathrm{CSP}(\Gamma)$.*

Proof: Take an instance $\mathcal{P}$ from $\#\mathrm{CSP}(\Gamma \cup \{\varrho\})$ in the logic form, that is $\mathcal{P}$ is a conjunctive formula. For every constraint $\varrho(v_1, \ldots, v_m)$ from $\mathcal{P}$ we do the following: Rewrite the conjunctive formula Φ_ϱ that expresses ϱ so that its free variables are precisely $v_1, \ldots, v_m$. So we have that Φ_ϱ is of the form:

$$\varrho_1(v_{11}, \ldots, v_{1n_1}) \wedge \ldots \wedge \varrho_k(v_{k1}, \ldots, v_{kn_k}) \tag{1}$$

where $\varrho_1, \ldots, \varrho_k \in \Gamma$ depend only on the predicate ϱ, and $v_{11}, \ldots, v_{1n_1}$, $v_{21}, \ldots, v_{kn_k} \in \{v_1, \ldots, v_m\}$. Replace $\varrho(v_1, \ldots, v_m)$ with (1). We obtain a problem instance with the identical set of solutions.

Now let $\mathcal{P}$ be an instance of #CSP($\Gamma \cup \{=_A\}$). In order to get rid of the relation of equality, $=_A$, we use the procedure introduced in [34]: For every constraint of the form $=_A (u, v)$ in $\mathcal{P}$, we remove it from $\mathcal{P}$ and replace every occurence of v with u.

The obtained problem instance $\mathcal{P}^*$ belongs to #CSP(Γ). Obviously, the reduction can be fulfilled in polynomial time. Furthermore, although the set of solutions to $\mathcal{P}^*$ differs from the set of solutions to $\mathcal{P}$ (since some variables are removed), both have the same cardinality. $\square$

Further constructions preserving #-tractability are much less easy and obvious. If we also allow existential quantification for conjunctive formulas then we obtain a larger class of formulas, called *primitive positive* or *pp-* formulas. The semantics and the relation expressed by a pp-formula is defined similarly to those for conjunctive formulas.

Proposition 2 *Let Γ be a constraint language, let ϱ be a relation in Γ and let σ be the relation defineded by $\exists x_m \varrho(x_1, \ldots, x_m)$. Then #CSP($\Gamma \cup \{\sigma\}$) is reducible to #CSP($\Gamma$).*

To prove Proposition 2 and some other results in this paper, we use the interpolation technique introduced in [56]. This technique is based on the following lemma that we borrow from [21].

Lemma 1 (Lemma 3.2, [21]) *Let $w_1, \ldots, w_r$ be known distinct nonzero constants. Suppose that we know values $f_1, \ldots, f_r$ such that*

$$f_s = \sum_{i=1}^{r} c_i w_i^s$$

for $1 \leq s \leq r$. The coefficients $c_1, \ldots, c_r$ can be evaluated in a time polynomial in r and $\max_{s=1,\ldots,r}\{\log w_s, \log f_s\}$.

Remark 1 *In most cases we are interested not in the individual values of the c_i, but rather in the sum of them. This allows us to deal with a situation when not all of $w_1, \ldots, w_r$ are distinct. Indeed, if $w_i = w_j$ then we replace c_i and c_j with their sum, and so shorten the sums above.*

Proof (of Proposition 2):

For each tuple $(a_1, \ldots, a_{n-1}) \in \sigma$ there are several b such that $(a_1, \ldots,$

$a_{n-1}, b) \in \varrho$. 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 $\mathbf{a}_j$. Clearly, $u_j > 0$ for all j.

Take a problem instance $\mathcal{P} = C_1 \wedge \ldots \wedge C_t$ in #CSP($\Gamma \cup \{\sigma\}$). Without loss of generality we may assume that $C_1, \ldots, C_s$, $s \leq t$, are the constraints containing σ. Then $n_i = n-1$ for $i \in \{1, \ldots, s\}$. Let $l \geq 1$ and let $\mathcal{P}^{(l)}$ be the problem instance from #CSP(Γ) in which each constraint $C_i = \sigma(x_{i_1}, \ldots, x_{i_{n-1}})$ with $1 \leq i \leq s$ is replaced with constrainsts $C_i^1, \ldots, C_i^l$, where each C_i^j, $1 \leq j \leq l$, is the constraint $\varrho(x_{i_1}, \ldots, x_{i_{n-1}}, y_{ij})$ such that the variables y_{ij} are all different and do not occur in $\mathcal{P}$.

For a solution φ to $\mathcal{P}$, let $a_\varphi(j)$ denote the number of constraints $C_i = \sigma(x_{i_1}, \ldots, x_{i_{n-1}})$, $i \leq s$, such that $(\varphi(x_{i_1}), \ldots, \varphi(x_{i_{n-1}})) = \mathbf{a}_j \in \sigma$. Clearly, $a_\varphi(1) + \ldots + a_\varphi(q) = s$. Furthermore, let $N(b_1, \ldots, b_q)$ denote the number of those solutions φ for which $a_\varphi(j) = b_j$, $1 \leq j \leq q$. To solve the problem $\mathcal{P}$ it suffices to find the sum of all numbers of this form. Each $N(b_1, \ldots, b_q)$ corresponds to a partition of s in q nonnegative summands $b_1 + \ldots + b_q$. 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 $\mathcal{P}$.

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

$$(u_1^l)^{a_\varphi(1)} \cdot (u_2^l)^{a_\varphi(2)} \cdot \ldots \cdot (u_q^l)^{a_\varphi(q)} = (u_1^{a_\varphi(1)} \cdot u_2^{a_\varphi(2)} \cdot \ldots \cdot u_q^{a_\varphi(q)})^l$$

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

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

By Remark 1 we may assume that all the numbers $u_1^{b_1} \cdot \ldots \cdot u_q^{b_q}$ are different. The determinant of the linear system

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

is Vandermonde, and therefore by Lemma 1 the system can be solved in polynomial time. □

Constraint languages containing the relation of equality and closed with respect to pp-formulas definability have been intensively studied (see e.g. [47–49]) and have provided strong assistance in the study of the decision constraint satisfaction problem [9,10,34,36].

Definition 5 *A constraint language Δ on a set A is said to be a* relational clone *if it contains $=_A$ and every relation expressible by a pp-formula over Δ.*

For a constraint language Γ, the relational clone, consisting of $=_A$ and all relations definable by a pp-formula over Γ is denoted by $\langle\Gamma\rangle$.

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 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 properties of relations [34,36,49]. 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 $\mathbf{a}_1, \ldots, \mathbf{a}_m \in A^n$, where $\mathbf{a}_i = (\mathbf{a}_{1i}, \ldots, \mathbf{a}_{ni})$ $(i = 1 \ldots m)$, define $f(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ to be $(f(\mathbf{a}_{11}, \ldots, \mathbf{a}_{1m}), \ldots, f(\mathbf{a}_{n1}, \ldots, \mathbf{a}_{nm}))$.

Definition 6 *An m-ary operation f* preserves *an n-ary relation ϱ (or ϱ is* invariant *under f, or f is a* polymorphism of *ϱ) if for any $\mathbf{a}_1, \ldots, \mathbf{a}_m \in \varrho$ the tuple $f(\mathbf{a}_1, \ldots, \mathbf{a}_m)$ 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 4 ([54]) Let ϱ be the solution space of a system of linear equations over a finite 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

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

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 [49,54].

Proposition 3 ([49,54]) *For any constraint languages Γ_1, Γ_2 on the same finite set, $\Gamma_2 \subseteq \langle \Gamma_1 \rangle$ if and only if $\mathsf{Pol}\, \Gamma_1 \subseteq \mathsf{Pol}\, \Gamma_2$.*

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 finite set such that Γ_2 is finite and $\mathsf{Pol}\ \Gamma_1 \subseteq \mathsf{Pol}\ \Gamma_2$. Then $\#\mathrm{CSP}(\Gamma_2)$ is reducible to $\#\mathrm{CSP}(\Gamma_1)$. Therefore, if $\#\mathrm{CSP}(\Gamma_1)$ is #-tractable then so is $\#\mathrm{CSP}(\Gamma_2)$, and if $\#\mathrm{CSP}(\Gamma_2)$ is #P-complete then so is $\#\mathrm{CSP}(\Gamma_1)$.*

Thus, 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 that generalizes operation $x-y+z$ is the operation $xy^{-1}z$ of a group.

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

Theorem 5 *If Γ is a constraint language which is invariant under no Mal't-sev operation then Γ is #P-complete.*

In the next section we develop an algebraic machinary sufficient to prove Theorem 5. Then, in Section 5, we apply Theorem 5 to some particular cases of #CSP, obtaining new algorithms, reproving and sometimes generalizing existing results.

4 Algebraic Structure of #CSP

4.1 Algebraic constructions and #CSP

In this subsection we give basic algebraic definitions. We also introduce the notion of a "#-tractable algebra" and show how it relates to the complexity of problem classes of the form #CSP(Γ). In our algebraic definitions we follow [12,42]. For algebraic notions and results concerning the decision CSP the reader is referred to [8,10].

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

We shall slightly abuse the notation and denote by #CSP($\mathbb{A}$) the problem class #CSP($\mathsf{Inv}\ F$).

Making use of Definition 7 we reformulate Problem 1.

Problem 2 (Classification problem) *Characterise the #-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 6 ([13,14]) *A two-element algebra $\mathbb{A} = (\{0,1\}; F)$ is #-tractable if and only if $x - y + z \pmod 2$ 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 8 (1) *Let $\mathbb{A} = (A; F)$ be an algebra. The k-*th direct power *of $\mathbb{A}$ is the algebra $\mathbb{A}^k = (A^k; F)$ where we treat each (n-ary) operation $f \in F$ as acting on A^k.*

(2) *Let $\mathbb{A} = (A; F)$ be an algebra, and let B be 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 $\mathbb{A}$.*

(3) *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 [32,42]. Therefore, the next theorem allows us to apply the methods of modern algebra to the study of the complexity of the counting CSP.

Theorem 7 *Let $\mathbb{A} = (A; F)$ 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.*

Proof: We show that for each of the mentioned constructions the resulting problem is reducible to #CSP($\mathbb{A}$). Thus we prove both parts (i) and (ii). The proof is subdivided in accordance with the construction considered.

(1) Let $\mathbb{B} = (B; F\big|_B)$ be a subalgebra of $\mathbb{A}$. This means that any relation $\varrho \in \mathsf{Inv}\, F\big|_B$ is also invariant with respect to F. Therefore $\mathsf{Inv}\, F\big|_B \subseteq \mathsf{Inv}\, F$, and #CSP($\mathbb{B}$) is trivially reducible to #CSP($\mathbb{A}$).

(2) Let $\mathbb{A}^k = (A^k; F)$ be a direct power of $\mathbb{A}$. Then any (n-ary) $\varrho \in \mathsf{Inv}\ F$ on A^k can be encoded in the form

$$e(\varrho) = \{(a_{11}, \dots, a_{1k}, a_{21}, \dots, a_{nk}) \mid (\mathbf{a}_1, \dots, \mathbf{a}_n) \in \varrho, \mathbf{a}_i = (a_{i1}, \dots, a_{ik})\},$$

and it is well known and easy to check that $e(\varrho) \in \mathsf{Inv}\ F$.

Take a problem instance $\mathcal{P} = (V; A^k; \mathcal{C})$ from #CSP($\mathbb{A}^k$), and transform it to $\mathcal{P}' = (V'; A; \mathcal{C}')$ where

- $V^{(k)} = \{v^1, \dots, v^k \mid v \in V\}$ is a disjouint union of k copies of V;
- every $\langle s, \varrho\rangle \in \mathcal{C}$ where $s = (v_1, \dots, v_l)$ is replaced with $\langle s', e(\varrho)\rangle$, $s' = (v_1^1, \dots, v_1^k, v_2^1, \dots, v_l^k)$ and $e(\varrho)$ is constructed as above.

The instance $\mathcal{P}'$ has the same number of solutions as $\mathcal{P}$, and $\mathcal{P}' \in$ #CSP($\mathbb{A}$).

(3) Let $\mathbb{B} = (B; F')$ be a homomorphic image of $\mathbb{A} = (A; F)$ under a homomorphism φ. We prove that for any finite $\Gamma \subseteq \mathsf{Inv}\ F'$, there is a finite $\Delta \subseteq \mathsf{Inv}\ F$ such that #CSP(Γ) is reducible to #CSP(Δ). The result then follows straightforwardly. Take a finite constraint language $\Gamma \subseteq \mathsf{Inv}\ F'$. By Proposition 1, we may assume that $=_B \in \Gamma$. Denote by $\Delta = \varphi^{-1}(\Gamma)$ the set consisting of full preimages of relations from Γ. In particular, $\varphi^{-1}(=_B)$ is an equivalence relation on A, the kernel of φ. It is easy to check that $\Delta \subseteq \mathsf{Inv}\ F$. We solve the problem #CSP(Γ) with oracle #CSP(Δ) in polynomial time.

Let $\mathcal{P} = (V; B; \mathcal{C})$ be a problem instance from #CSP(Γ); and, for any $k \geq 1$, let $\mathcal{P}^{(k)} = (V^{(k)}; A; \mathcal{C}^{(k)}) \in$ #CSP(Δ) be defined as follows

- $V^{(k)} = \{v^1, \dots, v^k \mid v \in V\}$ is a disjoint union of k copies of V;
- for any $\langle s, \varrho\rangle \in \mathcal{C}$, $s = (v_1, \dots, v_n)$, we include in $\mathcal{C}^{(k)}$ the constraints $\langle s^1, \varphi^{-1}(\varrho)\rangle, \dots, \langle s^k, \varphi^{-1}(\varrho)\rangle$ where $s^i = (v_1^i, \dots, v_n^i)$;
- for every variable $v \in V$ and every pair $1 \leq i, j \leq k$ we include in $\mathcal{C}^{(k)}$ the constraint $\langle (v^i, v^j), \varphi^{-1}(=_B)\rangle$.

Notice that for a solution ψ to $\mathcal{P}^{(k)}$ and any i, $1 \leq i \leq k$, the mapping $\psi^i: V \to B$ defined through the rule $\psi^i(v) = \varphi(\psi(v^i))$, is a solution to $\mathcal{P}$, and moreover, $\psi^i = \psi^j$ for any $1 \leq i, j \leq k$.

Let the classes of $\varphi^{-1}(=_B)$ be $U_1, \dots, U_m$ and their sizes $u_1, \dots, u_m$ respectively. For any natural numbers $t_1, \dots, t_m$ with $t_1 + t_2 + \dots + t_m = |V|$ let us denote by $N(t_1, t_2, \dots, t_m)$ the number of solutions ψ to $\mathcal{P}$ such that the number of variables that take value in U_i is t_i for $i = 1, \dots, n$. We shall obtain the number of solutions of $\mathcal{P}$ as the sum of all numbers of the form $N(t_1, \dots, t_m)$. Observe that the number of such numbers does not exceed $(n+1)^m$, which is polynomial in the size of $\mathcal{P}$ as m is fixed.

In order to compute the numbers $N(t_1, \dots, t_m)$, we shall construct and solve a system of linear equations.

First, observe that the number of solutions to $\mathcal{P}^{(k)}$ is

$$N_k = \sum_{t_1+t_2+\ldots+t_m=|V|} N(t_1,\ldots,t_m)(u_1^k)^{t_1}(u_2^k)^{t_2}\ldots(u_m^k)^{t_m}.$$

Since each $\mathcal{P}^{(k)}$ belongs to #CSP(Δ), the numbers N_k can be found in polynomial time.

By Remark 1, we may assume all the numbers $u_1^{t_1}u_2^{t_2}\ldots u_m^{t_m}$ to be different for different tuples $(t_1,\ldots,t_m)$.

The determinant of the system constructed is Vandermonde, and hence we can use Lemma 1 to solve the system in polynomial time. □

An operation f on a set A is said to be *idempotent* if the equality $f(x,\ldots,x) = x$ holds for all x from A. 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}: $F_{\mathrm{id}} = \mathsf{Pol}\ (\mathsf{Inv}\ F \cup \{C_a \mid a \in A\})$ where $C_a = \{(a)\}$ means the unary relation containing only one tuple, namely (a). Such a relation is sometimes called *constant*

Theorem 8 *A finite algebra* $\mathbb{A}$ *is #-tractable [#P-complete] if and only if so is* $\mathsf{Id}(\mathbb{A})$*.*

Proof: Let $\mathbb{A} = (A; F)$ be a finite algebra. We show that, for every finite constraint language Γ on A, the problem #CSP($\Gamma \cup \{C_a \mid a \in A\}$) is reducible to #CSP($\Gamma$). Consequently, if $\mathsf{Inv}\ F$ is #-tractable then so is $\mathsf{Inv}\ F \cup \{C_a \mid a \in A\}$. Finally, since $F_{\mathrm{id}} = \mathsf{Pol}\ (\mathsf{Inv}\ F \cup \{C_a \mid a \in A\})$, the result follows.

Let $A = \{a_1,\ldots,a_n\}$ (assuming $a_1,\ldots,a_n$ are different) be a finite set and let Γ be a finite constraint language on A. It is known [49,54] that the n-ary relation $\varrho_1 = \{(h(a_1),\ldots,h(a_n)) \mid h : A \to A, h \in \mathsf{Pol}\ \Gamma\}$ is in $\langle\Gamma\rangle$.

Let $\mathcal{P} = (V; A; \mathcal{C})$ be a problem instance in #CSP($\Gamma \cup \{C_a \mid a \in A\}$) and let $\mathcal{P}'$ be the problem instance $(V'; A; \mathcal{C}')$ in #CSP($\Gamma \cup \{=_A, \varrho_1\}$), where $V' = V \cup \{v_a \mid a \in A\}$, ($v_a$ for $a \in A$ are variables not in V), and $\mathcal{C}'$ contains every constraint $C = \langle s, \varrho\rangle$ in $\mathcal{C}$, such that $\varrho \in \Gamma$. Furthermore, for every constraint $\langle v, \{a\}\rangle$ in $\mathcal{C}$, the set $\mathcal{C}'$ contains the constraint $\langle (v, v_a), =_A\rangle$, and also $\mathcal{C}'$ contains the constraint $\langle (v_{a_1},\ldots,v_{a_n}), \varrho_1\rangle$.

The number of solutions to $\mathcal{P}$ equals the number of solutions φ to $\mathcal{P}'$ such that

$\varphi(v_a) = a$ for all $a \in A$. Let $\mathcal{N}$ be the set of all such solutions. The cardinality of $\mathcal{N}$ can be computed in two stages. For the first stage, we consider the set $\mathsf{Part}(A)$ of all partitions of A and the partial order $\leq$ on $\mathsf{Part}(A)$: for partitions $\eta, \theta \in \mathsf{Part}(A)$, we have $\eta \leq \theta$ if and only if every class of η is a subset of a certain class of θ. The least element $\underline{0}$ of $\mathsf{Part}(A)$ is the partition every class of which is a singleton. The partitions that *cover* the least element (such elements are sometimes called *atoms* of $\mathsf{Part}(A)$) have one 2-element class and the other classes are singletons. For a partition θ, let $I(\theta)$ denote the *principal ideal* generated by θ, that is the set $\{\eta \in \mathsf{Part}(A) \mid \eta \leq \theta\}$.

For every partition $\theta \in \mathsf{Part}(A)$ we define $\mathcal{P}'_\theta$ as the problem $(V', A, \mathcal{C}' \cup \{\langle (v_a, v_{a'}), =_A\rangle \mid a, a' \text{ belong to the same class of } \theta\})$. Notice that any function φ is a solution of $\mathcal{P}'_\theta$ if φ is a solution of $\mathcal{P}'$ and, for every a, a' from the same class of θ, $\varphi(v_a) = \varphi(v_{a'})$. Let us denote N_θ the number of solutions to $\mathcal{P}'_\theta$. The number N_θ can be computed with the oracle #CSP(Γ) since $\{=_A, \varrho_1\} \subseteq \langle\Gamma\rangle$.

In the first stage we compute how many solutions φ of $\mathcal{P}'$ assign v_a, $a \in A$, pairwise different values. Let us denote by $\mathcal{M}$ the set of all such solutions. The cardinality of $\mathcal{M}$ can be obtained using Möbius inversion formula for poset $\mathsf{Part}(A)$ as follows. We define a function $w\colon \mathsf{Part}(A) \to \mathbb{Z}$ inductively:

– $w(\underline{0}) = 1$;

– for any partition $\theta \neq \underline{0}$, $w(\theta) = -\sum\limits_{\eta \in I(\theta) - \{\theta\}} w(\eta)$. We claim that

$$|\mathcal{M}| = \sum_{\theta \in \mathsf{Part}(A)} w(\theta) N_\theta.$$

Indeed, for any solution φ of $\mathcal{P}'$, we can relate a partition $\theta(\varphi)$ such that a, a' belong to the same class of $\theta(\varphi)$ if and only if $\varphi(v_a) = \varphi(v_{a'})$. Then φ is a solution of ${\mathcal{P}'}_\eta$ if and only if $\eta \in I(\theta(\varphi))$. Therefore, φ is counted in the sum above

$$\sum_{\eta \in I(\theta(\varphi))} w(\eta)$$

times. As is easily seen, this number equals 1 if the values $\varphi(v_a)$ are all different and equals 0 otherwise.

In the second stage we express the cardinality of $\mathcal{N}$ via the cardinality of $\mathcal{M}$. Let G be the set of all permutations in $\mathsf{Pol}\ \Gamma$. It is well known [49,54] that, since A is finite, G constitutes a permutation group.

We show that $\mathcal{M} = \{g\varphi \mid g \in G, \varphi \in \mathcal{N}\}$. For every solution φ in $\mathcal{N}$ and every $g \in G$, $g\varphi$ is also a solution of $\mathcal{P}'$ [36] and, since g is one-to-one, $g\varphi$ is in $\mathcal{M}$. Conversely, for every $\psi \in \mathcal{M}$, there exists some $g \in G$ such that $g(a) = \psi(v_a), a \in A$. Notice that $g^{-1} \in G$ implies $\varphi = g^{-1}\psi \in \mathcal{N}$, and therefore $\psi = g\varphi$.

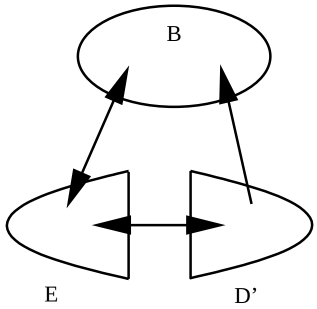

Fig. 1. The structure of the relation obtained in Lemma 2

Finally, for every $\varphi, \varphi' \in \mathcal{N}$ and every $g, g' \in G$, if $\varphi \neq \varphi'$ or $g \neq g'$ then $g\varphi \neq g'\varphi'$. In consequence, $|\mathcal{M}| = |G| \cdot |\mathcal{N}|$. □

4.2 Hard cases of #CSP

In this section we prove the main hardness result that later will allow us to obtain a necessary condition for tractability. Such a benchmark #P-complete problem arises from binary reflexive, but not symmetric relations.

Theorem 9 *If σ is a binary reflexive but not symmetric relation on a finite set then* $\#\mathrm{CSP}(\{\sigma\})$ *is #P-complete.*

Let σ be reflexive but not symmetric binary relation on a set A. We split a proof of Theorem 9 into three lemmas. The first one shows that relation σ can be 'improved', that is it suffices to prove the theorem for relations of a certain restricted form. The second lemma reduces ANTICHAIN to $\#\mathrm{CSP}(\sigma)$ in the case when σ is a relation on a 3-element set. Finally, the third lemma reduces a problem on such a small domain to $\#\mathrm{CSP}(\sigma)$ in the general case.

Lemma 2 *There exists a relation σ' on a set $A' \subseteq A$ such that* $\#\mathrm{CSP}(\sigma')$ *is reducible to* $\#\mathrm{CSP}(\sigma)$*, relation σ' is reflexive but no symmetric, and A' consists of 3 disjoint parts: B, E, and D' such that $(c, d) \notin \sigma$ if and only if $c \in B$ and $d \in D'$, see Fig 1.*

Proof: By Theorem 2 and Theorem 8, for any relation ϱ from the relational clone R generated by σ and the constant relations C_a, $a \in A$, the problem $\#\mathrm{CSP}(\varrho)$ is reducible to $\#\mathrm{CSP}(\sigma)$. Therefore it is enough to find a required relation σ' in R. As σ is no symmetric, there are $a, b \in A$ such that $(a, b) \in \sigma$ and $(b, a) \notin \sigma$. Such a pair of elements will be called *antisymmetric* for σ.

We shall prove that there exist a relation $\alpha \in R$ on some set $F \subseteq A$ containing some some antisymmetric pair $\{a, b\}$ for α satisfying the following conditions:
(P1) $(a, c), (c, b) \in \alpha$ for any $c \in F$,

(P2) $(c,a) \in \sigma$ implies $(c,d) \in \sigma$ for any $c,d \in F$, and
(P3) $(c,a) \notin \sigma$ implies $(d,c) \in \sigma$ for any $c,d \in F$;

Let α be a minimal (with respect to inclusion) relation in R containing an antisymmetric pair. We shall show that α contains an antisymmetric pair satisfying properties (P1)-(P3) (indeed, we shall prove that *any* antisymmetric pair satisfies conditions (P1)-(P3) although we do not need this stronger fact in our proof).

Let $\{a,b\}$ be an antisymmetric pair for α. Let us assume that $\{a,b\}$ does not satisfy (P1). Let F' be the set of all $c \in F$ such that $(a,c),(c,b) \in \alpha$. Observe that since the pair $\{a,b\}$ does not satisfy (P1), $F' \neq F$. Notice also that $a,b \in F'$. The unary relation F' belongs to R, as the following formula shows:

$$F'(x) = \exists y,z\ \sigma(y,x) \wedge C_a(y) \wedge \sigma(x,z) \wedge C_b(z)$$

where C_d is the predicate corresponding to the relation C_d, $d \in A$. Let us define α' to be $\alpha \cap F'^2$. Relation α' contradicts the minimality of α since, by its construction α' belongs to R, $\alpha' \subsetneq \alpha$, and α' contains the antisymmetric pair $\{a,b\}$.

Assume now that $\{a,b\}$ does not satisfy (P2). Consequently, there exist some $c,d \in F$ such that $(c,a) \in \alpha$ and $(c,d) \notin \alpha$. In this case, set $F' = \{d \in F \mid (c,d) \in \alpha\}$. Relation F' belongs to R:

$$F'(x) = \exists y\ \alpha(y,x) \wedge C_c(y),$$

Since $F' \neq F$, relation $\alpha' = \alpha \cap F'^2$ contradicts again the minimality of α.

Finally, if $\{a,b\}$ does not satisfy (P3) there exist some $c,d \in F$ such that $(c,a) \notin \alpha$ and $(d,c) \notin \alpha$. We set $F' = \{d \in F \mid (d,c) \in \alpha\}$. The relation F' belongs to R:

$$F'(x) = \exists y\ \alpha(x,y) \wedge C_c(y),$$

and $a,c \in F'$. Note that $\{a,c\}$ is an antisymmetric pair for $\alpha' = \alpha \cap F'$. Again α' contradicts the minimality of α.

To proceed further we need some additional notation. Let a,b be an antisymmetric pair for α. Set $D = \{c \in F \mid (c,a) \in \alpha\}$, $B = F - D$, and $D_d = \{c \in D \mid (d,c) \in \alpha\}$ for $d \in B$. Then $a \in D$, $b \in B$, and $a \in D_c$ for no $c \in B$. The only thing that remains to prove is that α is such that for any $d \in B$ we have $D_d = D_b$

Choose an element $d \in B$ such that D_d is maximal, and set $B' = \{c \in B \mid D_c = D_d\}$. It is straightforward to see that relation $B' \cup D$ belongs to R:

$$(B' \cup D)(x) = \exists y_1, \ldots, y_k \bigwedge_{i=1}^{k} (\alpha(x, y_i) \wedge C_{a_i}(y_i))$$

where $D_d = \{a_1, \ldots, a_k\}$ (the conjunction in the brackets tells that $(x, a_i) \in \sigma_3$). The relation $\sigma' = \alpha \cap (B' \cup D)^2$ on the set $A' = B' \cup D$ satisfies the conditions of Theorem 9 for B chosen as above, $E = D_b$, and $D' = D - E$, because $a, d \in B' \cup D$ is an antisymmetric pair for σ'. $\square$

Lemma 3 *The problem* ANTICHAIN *is reducible to* #CSP(σ')*, where* σ' *is such that* $|B| = |D'| = 1$ *and* $|E| \leq 1$.

Proof: Notice first that if $E = \varnothing$ then #CSP(σ') is equivalent to ANTICHAIN by Example 2. So we assume $|E| = 1$. Let us denote the only element from D' by 0, the element from B by 1, and the element from E by 2. Let $(P; \leq)$ be a problem instance of ANTICHAIN. As is observed in Example 2, there is a one-to-one correspondence between antichains of the poset P and its filters.

Consider the problem instance $\mathcal{P}^{(2)} = (V^{(2)}; A; \mathcal{C}^{(2)})$ defined as follows.

- $V^{(2)} = P_1 \cup P_2$ where P_l, P_2 are disjoint copies of P, and if $P = \{p_1, \ldots, p_n\}$ then $P_i = \{p_1^i, \ldots, p_n^i\}$.
- $\mathcal{C}^{(2)}$ comprises the constraints of the form
  - $\langle (p_j^1, p_j^2), \sigma' \rangle, \langle (p_j^2, p_j^1), \sigma' \rangle$ for $j \in \{1, \ldots, n\}$, and
  - $\langle (p_{j_1}^i, p_{j_2}^{i'}), \sigma' \rangle$ for $p_{j_1} \leq p_{j_2}$, $j_1 \neq j_2$, $i, i' \in \{1, 2\}$.

We observe some properties of a solution φ of $\mathcal{P}^{(2)}$.

(1) If $p_{j_1} \leq p_{j_2}$ and $\varphi(p_{j_1}^{i_1}) = 1$ then $\varphi(p_{j_2}^{i_2}) \in \{1, 2\}$ for any i_2 if $j_1 \neq j_2$, and for $i_2 \geq i_1$ if $j_1 = j_2$.

(2) If i is the least number such that $\varphi(p_j^i) = 1$, then for any $i' < i$, we have $\varphi(p_j^{i'}) \in \{0, 2\}$.

Furthermore, let H_φ be the set of elements $p_j \in P$ that are minimal amongst the elements with the property that $\{\varphi(p_j^1), \varphi(p_j^2)\}$ contains elements equal to 1. Clearly H_φ is an antichain, so, let F_φ denote the corresponding filter. By properties (1),(2), for any p_j^i such that $p_j \in F_\varphi$, and $p_j \notin H_\varphi$ or $p_j \in H_\varphi$ and $i \geq i'$ for some i' such that $\varphi(p_j^{i'}) = 1$, we have $\varphi(p_j^i) \in \{1, 2\}$, and for any p_j^i with $p_j \notin F\varphi$, or $p_j \in H_\varphi$ and $\varphi(p_j^{i'}) = 1$ for no $i' \leq i$, we have $\varphi(p_j^i) \in \{0, 2\}$. Moreover, if $p_j \notin H_\varphi$ then there are no further restrictions on $\varphi(p_j^i)$; however, if $p_j \in H_\varphi$ then at least one of $\varphi(p_j^1), \varphi(p_j^2)$ must be equal to 1. Therefore, for an antichain H and the corresponding filter F, the number of solutions φ to

$\mathcal{P}^{(2)}$ such that $H_\varphi = H$, $F_\varphi = F$, equals

$$(2^{|P-F|})^2 \cdot (2 \cdot 2)^{|H|} \cdot (2^{|F-H|})^2,$$

where 2 is the size of both $\{0, 2\}$ and $\{1, 2\}$.

Let $M(x, y, z)$ denote the number of antichains H of P such that $|P - F| = x$, $|H| = y$, $|F - H| = z$. Obviously, the sum of all numbers of this form equals the number N of antichains in $(P, \le)$. The number N_2 of solutions to $\mathcal{P}^{(2)}$ satisfies the identity

$$\sum_{x+y+z=|P|} M(x, y, z) 2^{2x} \cdot 2^{2y} \cdot 2^{2z} = N_2,$$

and thus

$$N \cdot 2^{2|P|} = N_2,$$

that completes the proof. $\square$

Let us fix a relation σ'' satisfying the conditions of Lemma 3; and as before we let the only element from D' be denoted by 0, the element from B by 1, and the element from E (if any) by 2. We now reduce the problem #CSP(σ'') to #CSP(σ'), where σ' is on a set A', σ' satisfies the conditions of Lemma 2, and the sizes of the sets B, E, D' for σ' are unconstrained. Let us denote $a = |D'|$, $b = |B|$, and $c = |E|$.

Lemma 4 *The problem* #CSP(σ'') *is reducible to* #CSP(σ').

Proof: Let $\mathcal{P} = (V; \{0, 1, 2\}; \mathcal{C})$ be a problem instance of #CSP(σ''). Let $N(x, y, z)$ denote the number of solutions φ of $\mathcal{P}$ such that x, y, and z are the sizes of the preimages of 0, 1, and 2 correspondingly. We show that these numbers can be computed in polynomial time with oracle #CSP(σ').

Consider the problem instance $\mathcal{P}^{(l)} = (V^{(l)}; A; \mathcal{C}^{(l)})$, for a natural number l, defined as follows.

- $V^{(l)} = V_0 \cup V_1 \cup \ldots \cup V_l$ where $V_0, V_1, \ldots, V_l$ are disjoint copies of V, and if $V = \{v_1, \ldots, v_n\}$ then $V_i = \{v^i_1, \ldots, v^i_n\}$.
- $\mathcal{C}^{(l)}$ comprises the constraints of the form
  - $\langle (v^0_j, v^i_j), \sigma' \rangle, \langle (v^i_j, v^0_j), \sigma' \rangle$ for $i \in \{1, \ldots, l\}$, $j \in \{1, \ldots, n\}$, and
  - $\langle (v^0_{j_1}, v^0_{j_2}), \sigma' \rangle$ for $\langle (v_{j_1}, v_{j_2}), \sigma'' \rangle \in \mathcal{C}$.

We observe some properties of a solution φ to $\mathcal{P}^{(l)}$. Recall that D denotes the set $D' \cup E$.

(1) If $\langle (v_{j_1}, v_{j_2}), \sigma'' \rangle$ is a constraint in $\mathcal{P}$, then $(\varphi(v^0_{j_1}), \varphi(v^0_{j_2})) \in \sigma'$.

(2) If $\varphi(v_j^0) \in D'$, then, for any i, we have $\varphi(v_j^i) \in D$.
(3) If $\varphi(v_j^0) \in B$, then, for any i, we have $\varphi(v_j^i) \in B \cup E$.
(4) If $\varphi(v_j^0) \in E$, then, for any i, the value $\varphi(v_j^i)$ can be any.

Every solution ψ to $\mathcal{P}$ is associated with a set of solutions φ to $\mathcal{P}^{(l)}$ such that if $\psi(v) = 0$ then $\varphi(v^0) \in D'$, if $\psi(v) = 1$ then $\varphi(v^0) \in B$, and if $\psi(v) = 2$ then $\varphi(v^0) \in E$. Obviously, the sets associated with distinct solutions to $\mathcal{P}$ are disjoint, and every solution of $\mathcal{P}^{(l)}$ is associated with some solution of $\mathcal{P}$.

The number of solutions of $\mathcal{P}^{(l)}$ associated with solutions ψ of $\mathcal{P}$ such that $|\{v \in V \mid \psi(v) = 0\}| = x$, $|\{v \in V \mid \psi(v) = 1\}| = y$, and $|\{v \in V \mid \psi(v) = 2\}| = z$, can now be computed:

$$|D'|^x(|D|^l)^x \cdot |B|^y(|B \cup E|^l)^y \cdot |E|^z(|A|^l)^z = a^x(a+c)^{lx} \cdot b^y(b+c)^{ly} \cdot c^z(a+b+c)^{lz}.$$

The number N_l of solutions to $\mathcal{P}^{(l)}$ satisfies the identity

$$\sum_{x+y+z=|V|} N(x,y,z)a^x b^y c^z \cdot ((a+c)^x(b+c)^y(a+b+c)^z)^l = N_l.$$

Let p denote the number of triples (x,y,z) with $x+y+z = |V|$. The number of solutions of $\mathcal{P}$ can be found as the sum of solutions to the following system of linear equations

$$\begin{aligned}
\sum_{x+y+z=|V|} N(x,y,z)a^x b^y c^z \cdot ((a+c)^x(b+c)^y(a+b+c)^z) &= N_1 \\
\sum_{x+y+z=|V|} N(x,y,z)a^x b^y c^z \cdot ((a+c)^x(b+c)^y(a+b+c)^z)^2 &= N_2 \\
&\vdots \\
\sum_{x+y+z=|V|} N(x,y,z)a^x b^y c^z \cdot ((a+c)^x(b+c)^y(a+b+c)^z)^p &= N_p.
\end{aligned}$$

Note that inside each column of the determinant of this system the factors of the form $a^x b^y c^z$ are all equal. Moreover, by Remark 1, we may assume all the numbers $(a+c)^x(b+c)^y(a+b+c)^z$ to be different. Therefore the determinant of this system can easily be transformed to a Vandermonde determinant. By Lemma 1, we can find the numbers $N(x,y,z)$. This completes the proof. □

Now Theorem 9 follows from Lemmas 3 and 4, and the #P-completeness of the problem Antichain.

Theorems 7 and 9, and results from [32] provide a link between the complexity of #CSP and Mal'tsev operations. The next statement follows from the results of [28] (see also Lemma 9.13 of [32]).

Theorem 10 ([28]) *For a finite algebra $\mathbb{A}$ the following conditions are equivalent.*

1. $\mathbb{A}$ does not have 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 9, the algebra $\mathbb{B}$ from Theorem 10(2) is #P-complete. Furthermore, Theorem 7 implies that $\mathbb{A}$ is also #P-complete.

Corollary 2 *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 2 expressed in terms of constraint languages.

5 Applications

5.1 2-element 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 [50], 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 a basis of this space in polynomial time, and furthermore, the number of solutions equals 2^n where n is the number of vectors in the basis. Thus, we have obtained another equivalent characterization of tractable Boolean #CSPs.

Theorem 11 *A constraint language Γ over a 2-element set is #-tractable if and only if it has a Mal'tsev polymorphism. Otherwise it is #P-complete.*

5.2 Rectangularity and permutability

Relations invariant with respect to a Mal'tsev operation satisfy strong restrictions on their form. One of them is especially useful. Let ϱ be an (n-ary) relation and $I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$. Then $\mathrm{pr}_I \varrho$ denotes the k-ary relation $\{(a_{i_1}, \ldots, a_{i_k}) \mid (a_1, \ldots, a_n) \in \varrho\}$. The 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. 2 (here $(\mathbf{a},\mathbf{c})$ denotes the tuple $\mathbf{e}$ such that $e_i=a_i$ if $i\in I$ and $e_i=c_i$ if $i\in J$).

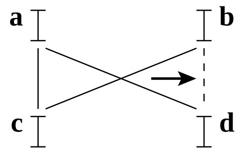

Fig. 2. The property of rectangularity

If ϱ is invariant under a Mal'tsev operation m then ϱ is rectangular. 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 3 *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 the 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 the fact that 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 [12,42].

*5.3 #H-***Coloring** *problem*

The problem Graph homomorphism [25] 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 graph H, and also for restrictions on the class of source graphs [1,2,18–20,27,29,31,57].

In particular, undirected graphs H for which H-Coloring is tractable have been characterized in [30]. An analogous result for $\#H$-Coloring has been obtained by Dyer and Greenhill.

Theorem 12 ([21]) *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.*

This theorem can be easily derived from Theorem 5. Note that in [21] a stronger version of Theorem 12 was proved. In that stronger version #P-complete problems remain #P-complete even when restricted to simple graphs with a constant degree bound. We prove a weaker version stated above, in which the degree of vertices is not bounded and loops are allowed.

Proof: Undirected graphs correspond to symmetric binary relations. Observe that graphs of rectangular symmetric binary relations are precisely those specified in Theorem 12. 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 classes of the bipartition and therefore the edge (b, d) presents.

Conversely, let H be a connected graph such that ϱ_H (see Example 3) 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 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 odd length then it also has a loop, a contradiction. Thus H is bipartite, and completeness follows straightforwardly from rectangularity.

Making use of Theorem 5, one may easily derive the #P-completeness part of Theorem 12. The tractability part of this theorem is fairly simple. $\square$

As an easy implication of the proof above and an observation that any graph satisfying the conditions of Theorem 12 has a Mal'tsev polymorphism we get the following:

Corollary 4 *For an undirected graph H, the #H-*Coloring *problem is tractable if and only if H has a Mal'tsev polymorphism. Otherwise it is #P-complete.*

In what follows we shall apply Theorem 5 in order to get new results. We start with identifying a simple condition that guarantees #P-completeness.

A digraph is said to be *N-free* if it does not contain a subgraph shown on Fig. 3 (not necessarily induced) such that edge (c, b) does not belong to the digraph.

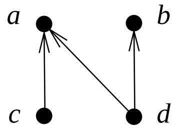

Fig. 3.

Lemma 5 *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. 3.

Our second application is a complete characterization of the oriented cycles H that give rise to tractable $\#H$-COLORING problems.

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. 4). Otherwise it is $\#P$-complete.*

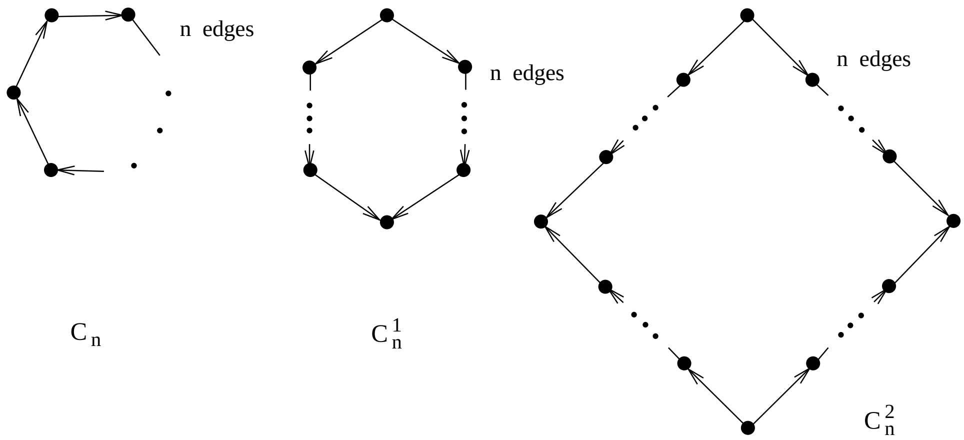

Fig. 4.

Proof: Let $H = (W; E)$ 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 proposition.

If H contains no vertex of zero indegree then $H = C_{|W|}$. 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.

There exists a vertex d different from b_{l-1} such that $(d, c) \in \varrho_H$ (see Fig. 5).

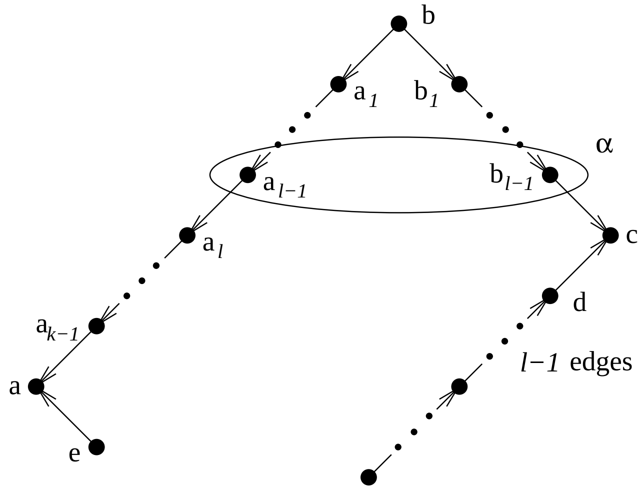

Fig. 5.

Let m be the Mal'tsev polymorphism of ϱ_H and let g be $m(a_{l-1}, b_{l-1}, d)$. We shall prove some basic facts about g.

First, we can infer that

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

Therefore, $g = a_{l-1}$ or $k = l$ and $g = e$ (see Fig. 5).

Let α be the binary relation on W defined in the following way: $(u, v) \in \alpha$ iff there exists a $w \in W$ and directed paths of lenght $l - 1$ connecting w with u and w with v.

Observe that α is definable by means of the formula

$$\begin{aligned} \alpha(u, v) = \exists x_1, y_1, x_2, y_2, \ldots, x_l, y_l \quad & (x_1 = y_1) \wedge (x_l = u) \wedge (y_l = v) \wedge \\ & \wedge \varrho_H(x_1, x_2) \wedge \varrho_H(y_1, y_2) \wedge \cdots \wedge \varrho_H(x_{l-1}, x_l) \wedge \varrho_H(y_{l-1}, y_l) \end{aligned}$$

and hence, m is a polymorphism of α. Notice that, since l is the minimal length of a maximal path, the pair (d, d) is in α. Then

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

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

We sketch counting algorithms for all three types of graphs.

Assume that we want to compute the number of homomorphisms from a given graph $G = (W', E')$ to H where H is C_n, C_n^1, or C_n^2. Let G_i, $i = 1, \ldots, r$, be the connected components of G and let m_i, $i = 1, \ldots, r$, be the number of homomorphisms from G_i to H. It is easy to see that the number of homomorphisms from G to H is $m_1 \cdot m_2 \cdot \ldots \cdot m_r$. Hence we can assume that G is connected.

If $H = C_n$, then the algorithm is trivial: a homomorphism is completely determined by the image of any vertex of G.

Suppose that $H = C_n^1$, and let $a_0^*, a_1^0, \ldots, a_n^*$ and $a_0^*, a_1^1, \ldots, a_n^*$ denote the maximal paths.

Let L_n be the graph with vertex set $0, \ldots, n$ and with set of edges $\{(0,1), (1,2), \ldots, (n-1, n)\}$. It is easy to observe that there exists a unique homomorphism ψ from C_n^1 to L_n, given by $\psi(a_j^i) = j$, $i = 0, 1, *$, $j = 0, \ldots, n$. Consequently, for every homorphism φ from G to C_n^1, we have that $\psi \cdot \varphi$ is an homomorphism from G to L_n.

In order to count the number of homomorphisms from G to C_n^1 we shall do the following. First we shall enumerate all homomorphisms from G to L_n. Observe that every such homomorphism ϕ is completely determined by the image of any vertex of G, so this can be easily done in polynomial time and, moreover, the number of such homomorphisms does not exceed the number of vertices in G multiplied by $n + 1$.

Then, for every such ϕ we shall compute how many different homomorphisms φ from G to C_n^1 satisfy $\psi \cdot \varphi = \phi$. We shall denote this number by N_ϕ.

In order to compute N_ϕ we shall do the following: Let G_ϕ be the subgraph of G induced by those vertices v such that $\phi(v) \notin \{0, n\}$. Then $N_\phi = 2^m$ where m is the number of connected components of G_ϕ. Let us prove it. Let f be any mapping from W' to $\{0, 1, *\}$ such that (1) $f(v) = *$ iff $\phi(v) \in \{0, n\}$ and such that (2) $f(v) = f(w)$ if v, w belong to the same connected component in G_φ. Then we have that the mapping φ defined by $\varphi(v) = a_{\phi(v)}^{f(v)}$ is a homomorphism from G to C_n^1. Conversely, for each homomorphism φ from G to C_n^1 we have that there exists a mapping f satisfying (1) and (2) such that $\varphi(v) = a_{\phi(v)}^{f(v)}$. Thus N_ϕ is equal to the number of mappings f satisfying (1) and (2). An easy computation shows that this number is 2^m.

Finally, consider the case $H = C_n^2$. The algorithm here is similar to the one for C_n^1. Let $a_0^{*,0}, a_1^{0,0}, a_2^{0,0}, \ldots, a_n^{0,*}$, $a_0^{*,0}, a_1^{1,0}, a_2^{1,0}, \ldots, a^{1,*}$, $a_0^{*,1}, a_1^{0,1}, a_2^{0,1}, \ldots, a_n^{0,*}$, and $a_0^{*,1}, a_1^{1,1}, a_2^{1,1}, \ldots, a_n^{1,*}$ be the maximal paths of C_n^2.

Again it is easy to observe that the mapping ψ given by $\psi(a_k^{i,j}) = k$ is the unique homomorphism from C_n^2 to L_n. In order to count the number of homomorphism from G to C_n^2 we again enumerate all homomorphisms from G to L_n. Then for every such homomorphism ϕ we compute the number N_ϕ of homomorphisms φ from G to C_n^2 such that $\psi \cdot \varphi = \phi$.

In order to compute N_ϕ we do the following. Let G_ϕ^0 be the subgraph of G induced by those vertices v such that $\phi(v) \neq 0$ and let G_ϕ^n be the subgraph of G induced by those vertices v such that $\phi(v) \neq n$. Using a line of reasoning similar to the previous case it is easy to see that N_ϕ is equal to $2^{m_0+m_n}$ where m_i is the number of connected components of G_ϕ^i. $\square$

As in the previous cases, the proof of Proposition 4 implies

Corollary 5 *For an oriented cycle H, the #H-Coloring problem is tractable if and only if H has a Mal'tsev polymorphism. Otherwise it is #P-complete.*

5.4 **List #H-Coloring** *problem*

Let H be a directed graph. In the List #H-Coloring problem we are given as input a (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 #CSP(Γ_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 [19] and [29]. The criteria happened to be the same as that for #H-Coloring.

Theorem 13 ([19,29]) *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.*

Similarily to Theorem 12, Theorem 13 implies

Corollary 6 *For an undirected graph H, the* List #H-Coloring *problem is tractable if and only if H fas a Mal'tsev polymorphism. Otherwise it is #P-complete.*

However, in the case of the List #H-Coloring problem, we can impose stronger restrictions onto a Mal'tsev polymorphism required. If List #H-

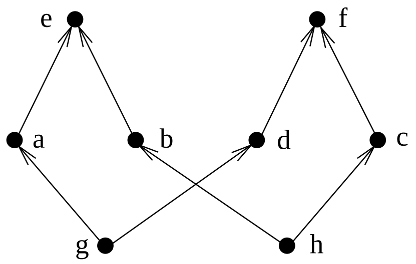

Fig. 6.

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*. This observation allows us to give an example of a digraph H, for which #H-Coloring problem is tractable while the List #H-Coloring problem is #P-complete.

Let us consider the digraph $H = C_2^2$ (Fig. 6). 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

$$m\left(\begin{pmatrix} a \\ b \end{pmatrix}, \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} c \\ c \end{pmatrix}\right) = \begin{pmatrix} c \\ c \end{pmatrix} \in \delta_1 \quad m\left(\begin{pmatrix} a \\ a \end{pmatrix}, \begin{pmatrix} b \\ b \end{pmatrix}, \begin{pmatrix} c \\ b \end{pmatrix}\right) = \begin{pmatrix} a \\ a \end{pmatrix} \in \delta_2.$$

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

6 Towards a Dichotomy Theorem

As we saw in Section 5, in all the studied cases the tractability of a #CSP can be explained by the presence of a Mal'tsev polymorphism. It is therefore very natural to conjecture that the existence of such a polymorphism is a sufficient condition for tractability, and so we did in the conference version of this paper [5].

In this paper we provide an example that shows that the presence of a Mal'tsev polymorphism does not guarantee the #-tractability of a constraint language.

Example 5 Let us consider the #H-Coloring problem, where H is the graph shown on Fig. 7. Notice that the vertices of H are divided into three levels; we refer to these levels as to the bottom, intermediate and top levels.

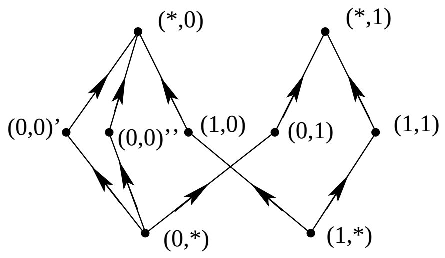

Fig. 7.

We define a polymorphism of H in two stages. First, we "collapse" vertices $(0,0)'$ and $(0,0)''$ into one vertex $(0,0)$ and define a ternary operation g on the resulting 8-element set. For any $x_1 = (i_1, j_1), x_2 = (i_2, j_2), x_3 = (i_3, j_3)$ where $i_1, i_2, i_3, j_1, j_2, j_3 \in \{*, 0, 1\}$, set $k_1 = i_1 - i_2 + i_3 \pmod 2$ (if $i_1, i_2, i_3 \in \{0, 1\}$), $k_2 = j_1 - j_2 + j_3 \pmod 2$ (if $j_1, j_2, j_3 \in \{0, 1\}$), and

$$g(x_1, x_2, x_3) = \begin{cases} (k_1, k_2) & \text{if } x_1, x_2, x_3 \text{ are from the intermediate level,} \\ x_\ell, & \text{if two of } x_1, x_2, x_3 \text{ are from the same level} \\ & \text{and } x_\ell \text{ is from another level,} \\ (*, k_2), & \text{if } x_1, x_2, x_3 \text{ are from the top level,} \\ (k_1, *), & \text{if } x_1, x_2, x_3 \text{ are from the bottom level,} \\ x_1 & \text{otherwise.} \end{cases}$$

Now, to define a polymorphism of H, we set $m(x_1, x_2, x_3) = g(x_1, x_2, x_3)$ whenever $g(x_1, x_2, x_3) \neq (0,0)$; otherwise $m(x_1, x_2, x_3) = (0,0)''$ if $x_1 = x_2 = x_3 = (0,0)''$ or exactly one of x_1, x_2, x_3 is $(0,0)''$ and the two others are equal; in all other cases $m(x_1, x_2, x_3) = (0,0)'$.

As is easily seen, this operation is Mal'tsev. We also show that m is a polymorphism of H. Indeed, it is enough to show that if x_1, x_2, x_3 and y_1, y_2, y_3 are such that $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ are edges of H, then $(m(x_1, x_2, x_3), m(y_1, y_2, y_3))$ is an edge of H. We consider 5 cases. As before, $x_1 = (i_1, j_1), x_2 = (i_2, j_2), x_3 = (i_3, j_3)$, $y_1 = (i'_1, j'_1), y_2 = (i'_2, j'_2), y_3 = (i'_3, j'_3)$, if say $i_1 = j_1 = 0$ then x_1 may be equal $(0,0)'$ or $(0,0)''$.

Case 1. x_1, x_2, x_3 are on the intermediate level.
In this case, $y_1 = (*, j_1), y_2 = (*, j_2), y_3 = (*, j_3)$, and $m(x_1, x_2, x_3) = (i_1 - i_2 + i_3, j_1 - j_2 + j_3)$ or $m(x_1, x_2, x_3) = (0,0)', (0,0)''$ ($j_1 - j_2 + j_3 = 0$ in this case) and $m(y_1, y_2, y_3) = (*, j_1 - j_2 + j_3)$. Thus $(m(x_1, x_2, x_3), m(y_1, y_2, y_3))$ is an edge.

Case 2. One of x_1, x_2, x_3 is on the bottom level and the others are on the intermediate level.
Let first x_1 be on the bottom level. Then y_1 is on the intermediate level and y_2, y_3 are on the top level. We have $m(x_1, x_2, x_3) = x_1$ and $m(y_1, y_2, y_3) = y_1$.

Obviosly, $(m(x_1,x_2,x_3), m(y_1,y_2,y_3))$ is an edge. If x_2 or x_3 are on the bottom level, the proof is essentially the same.

Case 3. One of x_1, x_2, x_3 is on the intermediate level and the others are on the bottom level.
As in Case 2, we have $m(x_1,x_2,x_3) = x_i$, $m(y_1,y_2,y_3) = y_i$, where x_i is the vertex on the intermediate level.

Case 4. x_1, x_2, x_3 are on the bottom level.
Then y_1, y_2, y_3 are on the intermediate level and $i'_1 = i_1, i'_2 = i_2, i'_3 = i_3$. We have $m(x_1,x_2,x_3) = (i_1 - i_2 + i_3, *)$ and $m(y_1,y_2,y_3) = (i_1 - i_2 + i_3, j'_1 - j'_2 + j'_3)$ (or $(i_1 - i_2 + i_3, j'_1 - j'_2 + j'_3)'$, or $(i_1 - i_2 + i_3, j'_1 - j'_2 + j'_3)''$), which constitute an edge of H.

Case 5. One of x_1, x_2, x_3 are on the top level.
This case is impossible, because there is no choice for the correspooning y.

However, the #MAXCUT problem can be reduced to #H-Coloring. It follows from the results of [21] that #MAXCUT is #P-complete, therefore #H-Coloring is also #P-complete. For a proof the reader is referred to [6,7].

7 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. This allows us, by making use of advanced and deep algebraic results (Theorem 10), to obtain a strong necessary condition on tractable cases of the counting constraint satisfaction problem (Theorem 5).

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 future research. This approach and the results of the conference version of this paper have already been used in, e.g., [6,7,22,38,44].

The applicability of the algorithm solving Mal'tsev decision constraint satisfaction problems to #CSP is limited. However, often those limitations can be overcome by using structural properties of relations invariant under a Mal'tsev operation. We strongly believe that future developments in algebraic theory 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.R. 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 and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. In *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science, FOCS'03*, pages 562–571, Cambridge, MA, USA, October 2003. IEEE Computer Society.

[6] A. Bulatov and M. Grohe. The complexity of partition functions. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming, ICALP'04*, pages 294–306, Turku, Finland, July 2004.

[7] A.A. Bulatov and M. Grohe. The complexity of partition functions. Technical Report PRG-RR-04-04, Computing Laboratory, University of Oxford, Oxford, UK, 2004.

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

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

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

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

[12] P.M. 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] V. Dalmau and P. Jonsson. The complexity of counting homomorphisms seen from the other side. *Theoretical Computer Science*, 329(1-3):315–323, 2004.

[17] J. Diaz, M. Serna, and D.M. 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.

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

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

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

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

[22] M. Dyer, L. Goldberg, M. Paterson. On counting homomorphisms to directed acyclic graphs. ECCC, TR05-121, 2005.

[23] T. Feder. Constraint satisfaction on finite groups with near subgroups. Electronic Colloquium on Computational Complexity (ECCC), TR05-005, 2005.

[24] T. Feder and M.Y. 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.

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

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

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

[28] J. Hagemann and A. Mitschke. On n-permutable congruences. *Algebra Universalis*, 3:8–12, 1972.

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

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

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

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

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

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

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

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

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

[38] B. Larose, O. Klima, P. Tesson. Systems of equations over finite semigroups and the #CSP dichotomy conjecture. In *Proceedings of MFCS'06*. To appear.

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

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

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

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

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

[44] G. Nordh and P. Jonsson. The Complexity of Counting Solutions to Systems of Equations over Finite Semigroups. In *Proceedings of COCOON'04*, pages 370-379, 2004.

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

[46] C. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

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

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

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

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

[51] J.S. Provan and M.O. 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.

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

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

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

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

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

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