

Modular Counting over 3-Element and Conservative Domains

Andrei A. Bulatov ✉

Simon Fraser University, Burnaby, Canada

Amirhossein Kazeminia ✉

University of Oxford, UK

Abstract

In the Constraint Satisfaction Problem (CSP for short) the goal is to decide the existence of a homomorphism from a given relational structure \mathcal{G} to a given relational structure \mathcal{H} . If the structure \mathcal{H} is fixed and \mathcal{G} is the only input, the problem is denoted $\text{CSP}(\mathcal{H})$. In its counting version, $\#\text{CSP}(\mathcal{H})$, the task is to find the number of such homomorphisms. The CSP and $\#\text{CSP}$ have been used to model a wide variety of combinatorial problems and have received a tremendous amount of attention from researchers from multiple disciplines.

In this paper we consider the modular version of the counting CSPs, that is, problems of the form $\#_p\text{CSP}(\mathcal{H})$ of counting the number of homomorphisms to \mathcal{H} modulo a fixed prime number p . Modular counting has been intensively studied during the last decade, although mainly in the case of graph homomorphisms. Here we continue the program of systematic research of modular counting of homomorphisms to general relational structures. The main results of the paper include a new way of reducing modular counting problems to smaller domains and a study of the complexity of such problems over 3-element domains and over conservative domains, that is, relational structures that allow to express (in a certain exact way) every possible unary predicate.

2012 ACM Subject Classification Theory of computation \rightarrow Problems, reductions and completeness; Theory of computation \rightarrow Constraint and logic programming

Keywords and phrases Constraint Satisfaction Problem, Modular Counting

Digital Object Identifier 10.4230/LIPIcs.STACS.2026.22

Related Version *Full Version*: <https://arxiv.org/abs/2510.09950>

Funding *Andrei A. Bulatov*: This research is supported by a NSERC Discovery Grant.

Amirhossein Kazeminia: UKRI EP/X024431/1.

1 Introduction

The Constraint Satisfaction Problem. In a Constraint Satisfaction Problem (CSP) the goal is to decide the existence of a homomorphism from a given relational structure \mathcal{G} to a given relational structure \mathcal{H} , [27]. If the target structure is fixed and only \mathcal{G} is the input, the problem called a *nonuniform CSP* and is denoted by $\text{CSP}(\mathcal{H})$, [27]. CSPs and nonuniform CSPs in particular have been used to model a wide variety of combinatorial problems across many disciplines. The complexity of problems of the form $\text{CSP}(\mathcal{H})$ for finite relational structures has been thoroughly investigated [37, 15, 2, 11, 36, 10, 48].

In the counting version of the CSP, denoted by $\#\text{CSP}$ (or $\#\text{CSP}(\mathcal{H})$ if we are dealing with a nonuniform problem), the goal is to find the number of homomorphisms from a given structure \mathcal{G} to a given structure \mathcal{H} , or in the case of a nonuniform problem to a fixed structure \mathcal{H} . Counting CSPs have received much attention starting from the seminal work of Valiant [46, 45], to the classification of Boolean [21] counting CSPs and those over graphs [22], to the discovery of the algebraic method for such problems [12], to a complexity classification of unweighted counting CSPs [8, 23], to a sequence of results on weighted counting CSPs [24, 13, 32, 18, 20, 19] culminating at a complete characterization of counting CSPs with



© Andrei A. Bulatov and Amirhossein Kazeminia;
licensed under Creative Commons License CC-BY 4.0

43rd International Symposium on Theoretical Aspects of Computer Science (STACS 2026).

Editors: Meena Mahajan, Florin Manea, Annabelle McIver, and Nguyễn Kim Thăng

Article No. 22; pp. 22:1–22:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



complex weights by Cai and Chen [17]. In all the research mentioned above the goal is “exact” counting, that is, finding the exact number of (weighted) homomorphisms between relational structures. Approximate counting is another vast area of research that we do not touch in this paper. Statistical Physics often concerns with the problem of computing partition functions that is closely related to counting CSPs. In this paper we consider another variation of counting problems, counting modulo an integer.

The complexity class the Counting CSP belongs to is $\#P$, the class of problems of counting accepting paths of polynomial time nondeterministic Turing machines. There are several ways to define reductions between counting problems, but the most widely used ones are parsimonious reductions and Turing (polynomial time) reductions. A *parsimonious reduction* from a counting problem $\#A$ to a counting problem $\#B$ is an algorithm that, given an instance I of $\#A$ produces (in polynomial time) an instance I' of $\#B$ such that the answers to I and I' are equal. A *Turing* (or *polynomial time*) *reduction* is a polynomial time algorithm that solves $\#A$ using $\#B$ as an oracle.

Modular counting. While the complexity of and algorithms for exact counting are now well understood, counting modulo an integer has received much attention in the past decade. For an integer k and a relational structure \mathcal{H} the problem of counting homomorphisms from a given relational structure to \mathcal{H} modulo k is denoted by $\#_k\text{CSP}(\mathcal{H})$. Here (and in almost all papers on the subject) the focus is on prime moduli p .

One of the most important features of relational structures affecting the complexity of the corresponding modular counting CSP is the automorphism group, $\text{Aut}(\mathcal{H})$, of \mathcal{H} . Indeed, it can be easily shown that if $\pi \in \text{Aut}(\mathcal{H})$ has order p (or is a *p-automorphism*), then $\#_p\text{CSP}(\mathcal{H})$ is equivalent, i.e., parsimoniously interreducible, with $\#_p\text{CSP}(\mathcal{H}^\pi)$, where \mathcal{H}^π denotes the substructure of \mathcal{H} induced by the set $\text{Fix}(\pi)$ of the fixed points of π . Clearly, such a reduction by p -automorphisms can be repeated until the resulting structure is *p-rigid*, that is, has no p -automorphisms. Such a derivative p -rigid structure is unique up to an isomorphism, as it is shown in [26], and will be denoted by \mathcal{H}^{*p} .

A related problem that will be important in this paper is computing partition functions. Let M be a $k \times k$ -matrix with entries from (in the most general case) a semiring \mathbb{S} . Let $\text{EVAL}(M)$ denote the problem of computing the function

$$Z_M(G) = \sum_{\varphi: V \rightarrow [k]} \prod_{(u,v) \in E} M[\varphi(u), \varphi(v)]$$

for a given (di)graph $G = (V, E)$, called the *partition function* of G . The complexity of $\text{EVAL}(M)$ has been extensively studied in Statistical Physics and Theoretical Computer Science, see [47, 4, 14] for some samples. In the case of real or complex matrices the complexity of exact computing of partition functions is well understood [14, 18]. The case of matrices over a finite field that arises from modular counting is wide open and presents challenges that can be glimpsed from the “cancellation phenomenon” observed in [32, 17] and produced by elements of the field of a finite multiplicative order.

Existing results. A systematic research on modular counting was initiated by Faben in [25], who introduced the complexity classes $\#_pP$ of problems of counting, modulo p , accepting paths of polynomial time Turing machines. In the same paper he also proved some key hardness results and gave a complexity classification of $\#_p\text{CSP}(\mathcal{H})$ for 2-element structures \mathcal{H} into polynomial time solvable and $\#_pP$ -complete ones. The notion of reduction between modular counting problems is similar to that for exact counting: it is either polynomial time reduction, in the same sense as before, or parsimonious reductions, where the answers to the two instances must be equal modulo p .

Since then most of the effort has been directed at problems $\#_p\text{CSP}(\mathcal{H})$ where \mathcal{H} is a graph. Faben and Jerrum [26] investigated the case when $p = 2$ and \mathcal{H} is a tree, and, apart from classifying the complexity of the problem in this case, they also posed a conjecture stating that if \mathcal{H} is a p -rigid graph, $\#_p\text{CSP}(\mathcal{H})$ has the same complexity as the exact counting problem $\#\text{CSP}(\mathcal{H})$. In a series of papers [31, 30, 33, 43, 28, 40] the Faben–Jerrum conjecture was confirmed for a number of graph classes and values of p , and then was proved in full generality by Bulatov and Kazeminia in [16].

While modular counting of graph homomorphisms provides some insight into the more general case, many of the complications do not occur in that case as it was demonstrated in [41]. As we will be using some of the results from this paper, we discuss it in some detail. The first observation is that the more general version of the Faben–Jerrum conjecture is not true for general relational structures: for any prime p there exists a p -rigid structure \mathcal{H} such that $\#\text{CSP}(\mathcal{H})$ is $\#\text{P}$ -complete, but $\#_p\text{CSP}(\mathcal{H})$ is polynomial time solvable. This can be fixed to some extent by introducing and exploiting multi-sorted relational structures with a richer structure of automorphisms. We will be using this framework as well, see Section 2.1. Although it is possible to modify the Faben–Jerrum conjecture for general multi-sorted structures in such a way that no counterexample is known so far, there is still little understanding of the general case.

There are two properties of graphs that made the result of [16] possible but fail for general relational structures. One is the structure of automorphisms of direct products of graphs. Due to the results of [35, 16] every automorphism of a direct product of graphs essentially boils down to a combination of automorphisms of the factors and a permutation of the factors. This is no longer true in the general case, even in the case of digraphs, as was demonstrated in [41]. This leads to a failure of the second important property: reductions through primitive positive (pp-) definitions. Let \mathcal{H} be a relational structure. Recall that a relation $\mathcal{R}(x_1, \dots, x_k)$ on H , the base set of \mathcal{H} , is said to be *pp-definable* in \mathcal{H} if there is a first order formula that represents \mathcal{R} and has the following form

$$\mathcal{R}(x_1, \dots, x_k) = \exists y_1, \dots, y_m \Phi(x_1, \dots, x_k, y_1, \dots, y_m),$$

where Φ is a conjunction of predicates from \mathcal{H} and equality predicates. If \mathcal{H} is a p -rigid graph, then the problem $\#_p\text{CSP}(\mathcal{H} + \mathcal{R})$, where $\mathcal{H} + \mathcal{R}$ denotes the *extension* of \mathcal{H} by adding the predicate \mathcal{R} , is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H})$. Such reductions are known to be a powerful tool in the CSP research, as they allow for the use of polymorphisms, that is, homomorphisms from powers \mathcal{H}^n to \mathcal{H} , to describe the complexity of constraint problems, see [15, 3]. In particular, the existence of a so-called *Mal'tsev polymorphism*, that is, a polymorphism $m : \mathcal{H}^3 \rightarrow \mathcal{H}$ satisfying the equations $m(x, y, y) = m(y, y, x) = x$, makes it possible to represent solutions to a CSP in a compact form that then in some cases can be used to find their number [11, 23]. In the case of general relational structures pp-definitions no longer give rise to reductions between problems and need to be replaced with their modular counterpart. A relation $\mathcal{R}(x_1, \dots, x_k)$ is said to be *p-mpp-definable* in \mathcal{H} for a prime p , it can be represented by a formula

$$\mathcal{R}(x_1, \dots, x_k) = \exists^{\neq p} y_1, \dots, y_m \Phi(x_1, \dots, x_k, y_1, \dots, y_m),$$

where the requirements on Φ are the same as before, and which is true for certain values of x_1, \dots, x_k if and only if the number of values of y_1, \dots, y_m that make $\Phi(x_1, \dots, x_k, y_1, \dots, y_m)$ true is not divisible by p . Then [41] proves that $\#_p\text{CSP}(\mathcal{H} + \mathcal{R})$ is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H})$ for any relational structure \mathcal{H} and any relation \mathcal{R} p -mpp-definable in \mathcal{H} .

Unfortunately, p -mpp-definitions cannot be captured by polymorphisms, and overcoming this difficulty will be an ongoing theme of this paper. In some cases, however, polymorphisms of p -mpp-definable relations play an important role. For instance, in [41] it was shown that if the set of 2-mpp-definable relations of a structure \mathcal{H} has a Mal'tsev polymorphism, then $\#_2\text{CSP}(\mathcal{H})$ can be solved in polynomial time.

Our contribution. In this paper we build upon the results of [41] to obtain a complexity classification of $\#_p\text{CSP}(\mathcal{H})$ for two classes of structures \mathcal{H} . We start with a new version of an automorphism and the corresponding rigidity condition.

A binary polymorphism f of a relational structure \mathcal{H} is said to be an *automorphic polynomial* if for every $a \in H$, $f(a, x)$ is a permutation of H . If, for a prime p , for any $a \in H$, $f(a, x)$ is the identity mapping or a permutation of order p , and $f(a, x)$ is a permutation of order p for at least one $a \in H$, f is said to be a *p -automorphic polynomial*. The existence of a p -automorphic polynomial allows one to reduce a modular counting CSP to a CSP over a smaller structure even when the structure is p -rigid, see Example 12. For a complete version of the following result see Proposition 10 and Corollary 11.

► **Proposition 1.** *Let a relational structure \mathcal{H} have a p -automorphic polynomial f . Then there is \mathcal{H}^f such that the domain of \mathcal{H}^f has smaller cardinality than \mathcal{H} and $\#_p\text{CSP}(\mathcal{H})$ is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H}^f)$. Moreover, if \mathcal{H} satisfies some additional conditions, \mathcal{H}^f can be chosen such that $\#_p\text{CSP}(\mathcal{H})$ and $\#_p\text{CSP}(\mathcal{H}^f)$ are polynomial time equivalent.*

In order to prove Proposition 1, given an instance (V, \mathcal{C}) of $\#_p\text{CSP}(\mathcal{H})$ we find permutations of the domain that are specific for each variable from V , have order p , and such that they map every tuple of every constraint to a tuple from the same constraint (so-called *consistent permutations*). These permutations allow us to reduce the domain \mathcal{H} to a smaller structure (in terms of the cardinality of the domain), because similar to automorphisms the elements that are shifted by the permutations can be eliminated from the problem. On the other hand, let $a \in H$ be an element such that $f(a, x)$ is a permutation of order p . Then, if the permutations above do not exist, it implies that the element a cannot be a part of any solution, and therefore can be eliminated. Consistent permutations are found by constructing a derivative CSP, whose domain is the symmetric group on H , and the CSP itself is an instance of a Mal'tsev CSP over that group [11, 5].

Recall that a unary relation pp-definable in a structure \mathcal{H} is said to be a *subalgebra* of \mathcal{H} . In the case of modular counting we use *p -subalgebras*, that is, unary relations p -mpp-definable in \mathcal{H} . The two types of relational structures we consider here are

- 3-element structures, which are the next step after a similar result for 2-element structures from [25], and
- *p -conservative* structures, i.e., structures such that every possible unary relation is a p -subalgebra.

The same types of the CSP have been major milestones in the study of the decision problem, see [6, 7, 1, 9]. They introduced several key ideas that were then used to obtain a dichotomy theorem for the general CSP.

In this paper in both cases our complexity classification is incomplete. The remaining difficulty is the complexity of modular partition functions. When $p = 2$ every partition function has a 0-1 matrix, and therefore can be treated as the problem of counting homomorphism to a graph or a bipartite graph. The complexity of this problem is known, see, [16]. If $p \geq 3$ this is no longer possible and the structure of matrices of partition functions becomes more intricate, requiring methods from finite fields. To give the reader a better idea of the

difficulties with modular partition functions, recall [8, 23] that in the case of exact counting a necessary and sufficient condition for the problem $\#CSP(\mathcal{H})$ to be solvable in polynomial time is that the structure \mathcal{H} is strongly balanced. It means that there is a way to associate integer matrices to relations pp-definable in \mathcal{H} , and these matrices must have rank at most 1. In fact, the complexity of $\#CSP$ is determined by the complexity of $EVAL(M)$, for the matrices M defined as above. That balancedness is necessary for $\#_pCSP(\mathcal{H})$ being solvable in polynomial time is no longer true, and this is due to the fact that the modular version of $EVAL(M)$ can be polynomial time solvable even if M does not have rank 1. Some partial results have been obtained in [39].

Let $\langle \mathcal{H} \rangle$ and $\langle \mathcal{H} \rangle_p$ denote the set of all relations that are, respectively, pp-definable and p -mpp-definable in \mathcal{H} . In the case of exact counting a necessary condition for tractability of $\#CSP(\mathcal{H})$ is the existence of a Mal'tsev polymorphism of \mathcal{H} , mentioned above. No similar property has been known to be true for modular counting. The two main results of this paper claim that for 3-element and p -conservative structures \mathcal{H} if $\langle \mathcal{H} \rangle_p$ has no Mal'tsev polymorphism then $\#_pCSP(\mathcal{H})$ is $\#_pP$ -hard. If a Mal'tsev polymorphism exists then we identify certain cases, in which the problem is polynomial time solvable.

► **Theorem 2.** *Let \mathcal{H} be a p -conservative structure and p a prime. If $\langle \mathcal{H} \rangle_p$ does not have a Mal'tsev polymorphism then $\#_pCSP(\mathcal{H})$ is $\#_pP$ -hard. Otherwise, if $p = 2$ then $\#_2CSP(\mathcal{H})$ is solvable in polynomial time.*

In order to state the result for 3-element structures let \mathcal{H}^f be the structure from Proposition 1 constructed from \mathcal{H} using a p -automorphic polynomial f . Note that if \mathcal{H} is a 3-element structure, \mathcal{H}^f has at most 2 elements.

► **Theorem 3.** *Let \mathcal{H} be a 3-element structure and p a prime. If one of the following conditions holds:*

- (a) \mathcal{H} has a p -automorphic polynomial f and $\langle \mathcal{H}^f \rangle_p$ has a Mal'tsev polymorphism, or
- (b) \mathcal{H} does not have a p -automorphic polynomial, $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism, and $p = 2$,

then $\#_pCSP(\mathcal{H})$ is solvable in polynomial time. If

- (1) \mathcal{H} has a p -automorphic polynomial f and $\langle \mathcal{H}^f \rangle_p$ does not have a Mal'tsev polymorphism, or
- (2) \mathcal{H} does not have a p -automorphic polynomial and $\langle \mathcal{H} \rangle_p$ does not have a Mal'tsev polymorphism,

then $\#_pCSP(\mathcal{H})$ is $\#_pP$ -complete.

Since \mathcal{H}^f has at most 2 elements, the cases when \mathcal{H} has a p -automorphic polynomial can be dealt with using an extension of the results by Faben [25], see Proposition 20.

Apart from the introduction of p -automorphic polynomials, the main technical contribution of the paper is the following. The property of relations with a Mal'tsev polymorphism that makes Mal'tsev polymorphisms useful is rectangularity. A relation \mathcal{R} (at least binary) is said to be *rectangular* if for any partition of the coordinate set of \mathcal{R} into two nonempty parts for any tuples \mathbf{a}, \mathbf{b} from the first part and \mathbf{c}, \mathbf{d} from the second part, if $(\mathbf{a}, \mathbf{c}), (\mathbf{b}, \mathbf{c}), (\mathbf{b}, \mathbf{d}) \in \mathcal{R}$, then $(\mathbf{a}, \mathbf{d}) \in \mathcal{R}$. If a structure \mathcal{H} has a Mal'tsev polymorphism then any relation $\mathcal{R} \in \langle \mathcal{H} \rangle$ is rectangular, a property that is sometimes referred to as *strong rectangularity*. It is the lack of rectangularity that often makes a counting CSP hard, as it was shown in [12]. In the case of modular counting a similar result is also true, although highly nontrivial, provided a non-rectangular relation is in \mathcal{H} , see [16]. The major difficulty though has been to obtain hardness from non-rectangular relations in $\langle \mathcal{H} \rangle$. Here we found a way to use $\langle \mathcal{H} \rangle_p$ instead, while still using a Mal'tsev polymorphism, although in a different manner than in other versions of counting.

2 Preliminaries

Let $[n]$ denote the set $\{1, 2, \dots, n\}$. Let H^n be the direct product of the set H with itself n times and $H_1 \times \dots \times H_n$ the direct product of sets H_1, \dots, H_n . We denote the members of H^n and $H_1 \times \dots \times H_n$ using bold font, $\mathbf{a} \in H^n$, $\mathbf{a} \in H_1 \times \dots \times H_n$. The i -th entry of \mathbf{a} is denoted by $\mathbf{a}[i]$. For $I = \{i_1, \dots, i_k\} \subseteq [n]$, we write $\text{pr}_I \mathbf{a}$ for the tuple $(\mathbf{a}[i_1], \dots, \mathbf{a}[i_k])$, and for a relation $\mathcal{R} \subseteq H^n$ or $\mathcal{R} \subseteq H_1 \times \dots \times H_n$, we define $\text{pr}_I \mathcal{R} = \{\text{pr}_I \mathbf{a} \mid \mathbf{a} \in \mathcal{R}\}$. The *arity* of \mathcal{R} , denoted by $\text{ar}(\mathcal{R})$, is defined to be n , the length of the tuples in \mathcal{R} . For $\mathbf{a} \in H_1 \times \dots \times H_s$, $s \in [n]$, let $\text{Ext}_{\mathcal{R}}(\mathbf{a}) = \{\mathbf{b} \in H_{s+1} \times \dots \times H_n \mid (\mathbf{a}, \mathbf{b}) \in \mathcal{R}\}$, and by $\#\text{ext}_{\mathcal{R}}(\mathbf{a})$ we denote the cardinality of $\text{Ext}_{\mathcal{R}}(\mathbf{a})$. (Note that we will often use the notation Ext in a more general sense, when $\mathbf{a} \in \text{pr}_I \mathcal{R}$ for a set $I \subseteq [n]$ that is not necessarily of the form $I = [s]$.) For a prime p we denote the cardinality of $\text{Ext}_{\mathcal{R}}(\mathbf{a}) \bmod p$ by $\#_p \text{ext}_{\mathcal{R}}(\mathbf{a})$. Moreover, $\text{pr}_I^p \mathcal{R}$ denotes the set $\{\text{pr}_I \mathbf{a} \mid \mathbf{a} \in \mathcal{R} \text{ and } \#_p \text{ext}_{\mathcal{R}}(\text{pr}_I \mathbf{a}) \neq 0\}$. Often, we treat relations $\mathcal{R} \subseteq H_1 \times \dots \times H_n$ as predicates $\mathcal{R} : H_1 \times \dots \times H_n \rightarrow \{0, 1\}$.

2.1 Relational structures

We begin by introducing *multi-sorted* sets. Let $H = \{H_i\}_{i \in [k]}$ be a collection of sets. We assume that the sets H_1, \dots, H_k are disjoint. A mapping φ between two multi-sorted sets $G = \{G_i\}_{i \in [k]}$ and $H = \{H_i\}_{i \in [k]}$ is defined as a collection of mappings $\varphi = \{\varphi_i\}_{i \in [k]}$, where $\varphi_i : G_i \rightarrow H_i$, that is, φ_i maps elements of the sort i in G to elements of the sort i in H . A mapping $\varphi = \{\varphi_i\}_{i \in [k]}$ from $\{G_i\}_{i \in [k]}$ to $\{H_i\}_{i \in [k]}$ is injective (bijective), if for all $i \in [k]$, φ_i is injective (bijective).

A *multi-sorted relational signature* σ over a set of types $[k]$ is a collection of *relational symbols*. A symbol $\mathcal{R} \in \sigma$ is assigned a positive integer $\ell_{\mathcal{R}}$, the *arity* of the symbol denoted $\text{ar}(\mathcal{R})$, and a tuple $(i_1, \dots, i_{\ell_{\mathcal{R}}}) \in [k]^{\ell_{\mathcal{R}}}$, the *type* of the symbol. A *multi-sorted relational structure* \mathcal{H} with signature σ is a multi-sorted set $\{H_i\}_{i \in [k]}$ and an *interpretation* $\mathcal{R}^{\mathcal{H}}$ of each $\mathcal{R} \in \sigma$, where $\mathcal{R}^{\mathcal{H}}$ is a relation or a predicate on $H_{i_1} \times \dots \times H_{i_{\ell_{\mathcal{R}}}}$. The multi-sorted structure \mathcal{H} is finite if H contains finitely many finite domains, and σ is finite. All structures in this paper are finite. The set H is said to be the *base set* or the *universe* of \mathcal{H} . For the base set we will use the same letter as for the structure, only in the regular font. Multi-sorted structures with the same signature and types are called *similar*.

Let $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1, \dots, \mathcal{R}_{\ell})$ be a relational structure. We use $\text{Sp}(\mathcal{H})$ to denote its *spectrum*, that is, an infinite sequence (n_1, n_2, \dots) where n_j is the number of domains of cardinality j . If the structure is finite, $\text{Sp}(\mathcal{H})$ is essentially a finite sequence, and we will truncate it by removing the trailing zeroes. Let \preceq denote the colexicographic order on the set of such sequences: $(n_1, \dots, n_k) \preceq (n'_1, \dots, n'_\ell)$, $n_k, n_\ell \neq 0$ if and only if either the sequences are equal, or $k < \ell$, or for some $i \in [k]$, $n_i < n'_i$ and $n_j = n'_j$ for $j \in \{i+1, \dots, k\}$.

Let \mathcal{G}, \mathcal{H} be similar multi-sorted structures with signature σ . A *homomorphism* φ from \mathcal{G} to \mathcal{H} is a mapping $\varphi : G \rightarrow H$ such that for any $\mathcal{R} \in \sigma$ with type $(i_1, \dots, i_{\ell_{\mathcal{R}}})$, if $\mathcal{R}^{\mathcal{G}}(a_1, \dots, a_{\ell_{\mathcal{R}}})$ is true for $(a_1, \dots, a_{\ell_{\mathcal{R}}}) \in G_{i_1} \times \dots \times G_{i_{\ell_{\mathcal{R}}}}$, then $\mathcal{R}^{\mathcal{H}}(\varphi_{i_1}(a_1), \dots, \varphi_{i_{\ell_{\mathcal{R}}}}(a_{\ell_{\mathcal{R}}}))$ is true as well. The set of all homomorphisms from \mathcal{G} to \mathcal{H} is denoted $\text{Hom}(\mathcal{G}, \mathcal{H})$. The cardinality of $\text{Hom}(\mathcal{G}, \mathcal{H})$ is denoted by $\text{hom}(\mathcal{G}, \mathcal{H})$. A homomorphism φ is an *isomorphism* if it is bijective and the inverse mapping φ^{-1} is a homomorphism from \mathcal{H} to \mathcal{G} . If \mathcal{H} and \mathcal{G} are isomorphic, we write $\mathcal{H} \cong \mathcal{G}$. A homomorphism of a structure to itself is called an *endomorphism*, and an isomorphism to itself is called an *automorphism*. As is easily seen, automorphisms of a structure \mathcal{H} form a group denoted $\text{Aut}(\mathcal{H})$. Let π be an automorphism of \mathcal{H} . By $\text{Fix}(\pi)$ we denote the collection $\{\text{Fix}(\pi_i)\}_{i \in [k]}$ of sets of fixed points of the π_i 's. For $\mathcal{Q} \subseteq H_{i_1} \times \dots \times H_{i_{\ell}}$, we use $\pi(\mathcal{Q})$ to denote the set $\{\pi(\mathbf{a}) \mid \mathbf{a} \in \mathcal{Q}\}$, where $\pi(\mathbf{a}) = (\pi_{i_1}(\mathbf{a}[1]), \dots, \pi_{i_{\ell}}(\mathbf{a}[\ell]))$. For a

prime number p we say that π has order p or is a p -automorphism if π has order p in $\text{Aut}(\mathcal{H})$. In other words, each of the π_i 's is either the identity mapping or has order p , and at least one of the π_i 's is not the identity mapping. The structure \mathcal{H} is said to be p -rigid if it has no p -automorphisms.

► **Proposition 4** ([41]). *Let \mathcal{H} be a multi-sorted structure and p a prime. Then up to an isomorphism there exists a unique p -rigid multi-sorted structure \mathcal{H}^{*p} such that for any structure \mathcal{G} similar to \mathcal{H} it holds that $\text{hom}(\mathcal{G}, \mathcal{H}) = \text{hom}(\mathcal{G}, \mathcal{H}^{*p})$.*

The direct product of multi-sorted σ -structures \mathcal{H}, \mathcal{G} , denoted $\mathcal{H} \times \mathcal{G}$ is the multi-sorted σ -structure with the base set $H \times G = \{H_i \times G_i\}_{i \in [k]}$, the interpretation of $\mathcal{R} \in \sigma$ is given by $\mathcal{R}^{\mathcal{H} \times \mathcal{G}}((a_1, b_1), \dots, (a_k, b_k)) = 1$ if and only if $\mathcal{R}^{\mathcal{H}}(a_1, \dots, a_k) = 1$ and $\mathcal{R}^{\mathcal{G}}(b_1, \dots, b_k) = 1$. By \mathcal{H}^ℓ we will denote the ℓ th power of \mathcal{H} , that is, the direct product of ℓ copies of \mathcal{H} . A substructure \mathcal{H}' of \mathcal{H} induced by a collection of sets $\{H'_i\}_{i \in [k]}$, where $H'_i \subseteq H_i$, is the relational structure given by $(\{H'_i\}_{i \in [k]}; \mathcal{R}'_1, \dots, \mathcal{R}'_m)$, where $\mathcal{R}'_j = \mathcal{R}_j \cap (H'_{i_1} \times \dots \times H'_{i_\ell})$ and (i_1, \dots, i_ℓ) is the type of \mathcal{R}_j .

2.2 The CSP and Modular Counting

There are two standard formulations of the Constraint Satisfaction Problem (CSP). Let $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1^{\mathcal{H}}, \dots, \mathcal{R}_\ell^{\mathcal{H}})$ be a multi-sorted relational structure. The problem $\text{CSP}(\mathcal{H})$ asks, given a relational structure $\mathcal{G} = (\{G_i\}_{i \in [k]}; \mathcal{R}_1^{\mathcal{G}}, \dots, \mathcal{R}_\ell^{\mathcal{G}})$ similar to \mathcal{H} whether there exists a homomorphism φ from \mathcal{G} to \mathcal{H} .

The other standard definition does not involve relational structures. Let $H = \{H_i\}_{i \in [k]}$ be a multi-sorted domain and Γ a set of multi-sorted relations over H , called a *constraint language*. An instance $\mathcal{P} = (V, \tau, \mathcal{C})$ of $\text{CSP}(\Gamma)$ consists of:

- a finite set of variables V ,
- a type function $\tau : V \rightarrow [k]$ assigning each variable a sort,
- a finite set of constraints \mathcal{C} , where each constraint is a pair $\langle \mathbf{s}, \mathcal{R} \rangle$ with $\mathcal{R} \in \Gamma$ of arity m and $\mathbf{s} = (v_1, \dots, v_m)$ a tuple of variables whose sorts match the type of \mathcal{R} .

A *solution* is a mapping $\varphi : V \rightarrow \bigcup_{i \in [k]} H_i$ such that for all $v \in V$, $\varphi(v) \in H_{\tau(v)}$, and for every constraint $\langle \mathbf{s}, \mathcal{R} \rangle$, the tuple $(\varphi(v_1), \dots, \varphi(v_m))$ belongs to \mathcal{R} .

For a structure $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1^{\mathcal{H}}, \dots, \mathcal{R}_\ell^{\mathcal{H}})$, let $\Gamma_{\mathcal{H}} = \{\mathcal{R}_1^{\mathcal{H}}, \dots, \mathcal{R}_\ell^{\mathcal{H}}\}$. It is well known, see e.g. [27] and [3], that the problems $\text{CSP}(\mathcal{H})$ and $\text{CSP}(\Gamma_{\mathcal{H}})$ can be easily translated into each other. The same holds for the counting versions $\#\text{CSP}(\mathcal{H})$ and $\#\text{CSP}(\Gamma_{\mathcal{H}})$, as well as the modular counting versions $\#_p\text{CSP}(\mathcal{H})$ and $\#_p\text{CSP}(\Gamma_{\mathcal{H}})$. The conversion procedure is as follows. Given an instance \mathcal{G} of $\text{CSP}(\mathcal{H})$ with signature σ , construct an instance $\mathcal{P} = (V, \tau, \mathcal{C})$ of $\text{CSP}(\Gamma_{\mathcal{H}})$ by setting $V = \bigcup_{i \in [k]} G_i$, with the type function τ induced by the sorts of \mathcal{G} , and for every relation $\mathcal{R} \in \sigma$ and tuple $\mathbf{s} \in \mathcal{R}^{\mathcal{G}}$, include the constraint $\langle \mathbf{s}, \mathcal{R}^{\mathcal{H}} \rangle$ in \mathcal{C} . Conversely, for a finite constraint language $\Gamma = \{\mathcal{R}_1, \dots, \mathcal{R}_\ell\}$ over H we can construct a structure $\mathcal{H}_\Gamma = (H, \mathcal{R}_1, \dots, \mathcal{R}_\ell)$ with an appropriately selected signature. Then any instance of $\text{CSP}(\Gamma)$ can be transformed into an instance of $\text{CSP}(\mathcal{H}_\Gamma)$ by reversing this construction. This correspondence naturally extends to the counting and modular counting versions.

2.3 Expansions of Relational Structures and the CSP

A (multi-sorted) relational structure $\mathcal{H}' = (H; \mathcal{R}'_1, \dots, \mathcal{R}'_{\ell'})$, $H = \{H_i\}_{i \in [k]}$, is an *expansion* of a relational structure $\mathcal{H} = (H; \mathcal{R}_1, \dots, \mathcal{R}_\ell)$, if $\ell' \geq \ell$ and $\mathcal{R}'_i = \mathcal{R}_i$ for $i \in [\ell]$. It will be convenient to denote \mathcal{H}' by $\mathcal{H} + \{\mathcal{R}_{\ell+1}, \dots, \mathcal{R}_{\ell'}\}$ or just $\mathcal{H} + \mathcal{R}_{\ell'}$ if $\ell' = \ell + 1$. Several types of expansions are often used in the context of the CSP.

Let $\mathcal{H} = (\{\mathcal{H}_i\}_{i \in [k]}; \mathcal{R}_1, \dots, \mathcal{R}_\ell)$ be a multi-sorted structure with signature σ and \mathcal{H}^- its expansion by adding a family of binary relational symbols $=_{H_i}$ (one for each type) interpreted as the equality relation on H_i , $i \in [k]$. A *constant relation* over a set $\{H_i\}_{i \in [k]}$ is a unary relation $C_{H_i, a} = \{a\}$, $a \in H_i$, $i \in [k]$ (such a predicate can only be applied to variables of type i). For a structure \mathcal{H} by \mathcal{H}^c we denote the expansion of \mathcal{H} by all the constant relations $C_{H_i, a}$, $a \in H_i$, $i \in [k]$.

► **Proposition 5** ([41]). *Let \mathcal{H} be a multi-sorted relational structure and p prime.*

- (1) $\#_p\text{CSP}(\mathcal{H}^-)$ is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H})$;
- (2) Let \mathcal{H} be p -rigid. Then $\#_p\text{CSP}(\mathcal{H}^c)$ is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H})$.

Observe that if a relational structure contains all the constant relations, in particular, any structure of the form \mathcal{H}^c , it is obviously rigid, as every automorphism should respect every $C_{H_i, a}$ and map a to a , and therefore also p -rigid for any p .

Using Proposition 4 we may assume that all the structures we are dealing with are p -rigid. Then, by Proposition 5(2) we may assume they contain all the constants, that is, $\mathcal{H} = \mathcal{H}^c$. We will use this assumption from now on.

Primitive-positive definitions play a central role in the CSP research. It has been proved in multiple circumstances that expanding a relational structure with pp-definable relations does not change the complexity of the corresponding CSP. This has been proved for the decision CSP in [37, 15], the exact counting CSP [12], and in certain cases (where relational structures are expansions of graphs) [16] of modular counting CSP. The reader is referred to [3] for details about primitive positive definitions and their use in the study of the CSP.

Let \mathcal{H} be a multi-sorted relational structure with the base set H . A *primitive positive* (pp-) formula in \mathcal{H} is a first-order formula

$$\exists y_1, \dots, y_s \Phi(x_1, \dots, x_k, y_1, \dots, y_s),$$

where Φ is a conjunction of atomic formulas of the form $z_1 =_H z_2$ or $\mathcal{R}(z_1, \dots, z_\ell)$, $z_1, \dots, z_\ell \in \{x_1, \dots, x_k, y_1, \dots, y_s\}$, and \mathcal{R} is a predicate of \mathcal{H} . Every variable in Φ is assigned a type $\tau(x_i), \tau(y_j)$ in such a way that for every atomic formula $z_1 =_H z_2$ it holds that $\tau(z_1) = \tau(z_2)$, and for any atomic formula $\mathcal{R}(z_1, \dots, z_\ell)$ the sequence $(\tau(z_1), \dots, \tau(z_\ell))$ matches the type of \mathcal{R} . We say that \mathcal{H} *pp-defines* a predicate \mathcal{Q} if there exists a pp-formula such that

$$\mathcal{Q}(x_1, \dots, x_k) = \exists y_1, \dots, y_s \Phi(x_1, \dots, x_k, y_1, \dots, y_s).$$

The set of all pp-definable relations in the (multi-sorted) relational structure \mathcal{H} is denoted by $\langle \mathcal{H} \rangle$.

For $\mathbf{a} \in \mathcal{Q}$ by $\#\text{ext}_\Phi(\mathbf{a})$ we denote the number of assignments $\mathbf{b} \in H_{\tau(y_1)} \times \dots \times H_{\tau(y_s)}$ to y_1, \dots, y_s such that $\Phi(\mathbf{a}, \mathbf{b})$ is true. We denote the number of such assignments modulo p by $\#_p\text{ext}_\Phi(\mathbf{a})$.

As was shown in [41], pp-definitions are not always compatible with modular counting in the sense that will be made precise later. The concept of pp-definitions adapted to modular counting is that of *modular pp-definitions*, or *p-mpp-definitions*, where p is the modulus for counting. For a prime number p the *p-modular* quantifier $\exists^{\neq p}$ is interpreted as follows

$$\begin{aligned} \mathcal{R}(x_1, \dots, x_k) = \exists^{\neq p} y_1, \dots, y_s \Phi(x_1, \dots, x_k, y_1, \dots, y_s) \text{ is true} \\ \Leftrightarrow \#_p\text{ext}_\Phi(x_1, \dots, x_k) \not\equiv 0 \pmod{p}. \end{aligned} \tag{1}$$

A primitive positive formula that uses p -modular quantifiers instead of regular ones is called a *p-modular primitive positive* (or *p-mpp* for short). The relation it defines is said to be *p-mpp-definable*, and the definition itself is called a *p-mpp-definition*. The set of all *p-mpp-definable* relations in \mathcal{H} is denoted by $\langle \mathcal{H} \rangle_p$. Note that modular quantifiers are not as robust

as regular ones. In particular, $\exists^{\neq p} x, y$ and $\exists^{\neq p} x \exists^{\neq p} y$ may result in a different predicate. The same is of course true for more complicated applications of modular quantifiers, so, one needs to be extra careful with p -mpp-definitions.

If for the p -mpp-definition (1) $\#_p \text{ext}_{\mathbb{F}}(\mathbf{a}) = 1$ for all $\mathbf{a} \in \mathcal{R}$, the p -mpp definition is said to be *strict*. By Proposition 5.6 from [16] if \mathcal{R} has a p -mpp-definition, it has a strict p -mpp definition.

► **Proposition 6** ([41]). *Let \mathcal{H} be a σ -structure (single- or multi-sorted), and p a prime. Let \mathcal{R} be a relation that is p -mpp-definable in \mathcal{H} . Then $\#_p \text{CSP}(\mathcal{H} + \mathcal{R})$ is polynomial time reducible to $\#_p \text{CSP}(\mathcal{H})$. In particular, if \mathcal{R} is conjunctive definable in \mathcal{H} (that is, $s = 0$ in (1) and the specific value of p is irrelevant), $\#_p \text{CSP}(\mathcal{H} + \mathcal{R})$ is polynomial time reducible to $\#_p \text{CSP}(\mathcal{H})$.*

Let $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1, \dots, \mathcal{R}_m)$ be a multi-sorted relational structure. A subset $A \subseteq H_i$ is called a *subalgebra* (of sort i) if it is pp-definable in \mathcal{H} as a unary relation. If A is p -mpp-definable in \mathcal{H} , it is said to be *p -subalgebra*. In other words, A is a p -subalgebra if there exists a relation $\mathcal{Q}(x)$ p -mpp-definable in \mathcal{H} , such that $a \in A$ if and only if $\mathcal{Q}(a)$.

2.4 Polymorphisms and Automorphisms of Products

We will also use polymorphisms of relational structures and constraint languages. Let $\mathcal{H} = (H : \mathcal{R}_1, \dots, \mathcal{R}_\ell)$, $H = \{H_i\}_{i \in [k]}$ be a multi-sorted relational structure. An r -ary *polymorphism* f of \mathcal{H} is a homomorphism $f : \mathcal{H}^r \rightarrow \mathcal{H}$. In other words, a collection of mappings $f = \{f_i\}_{i \in [k]}$, $f_i : H_i^r \rightarrow H_i$, such that for every \mathcal{R}_j , $j \in [\ell]$, with type (i_1, \dots, i_s) , for any $\mathbf{a}_1, \dots, \mathbf{a}_r \in \mathcal{R}_j$

$$(f_{i_1}(\mathbf{a}_1[1], \mathbf{a}_2[1], \dots, \mathbf{a}_r[1]), \dots, f_{i_s}(\mathbf{a}_1[s], \mathbf{a}_2[s], \dots, \mathbf{a}_r[s])) \in \mathcal{R}_j.$$

If this condition for \mathcal{R}_j holds, f is also said to be a polymorphism of \mathcal{R}_j . For a constraint language Γ over H the definition of a polymorphism is essentially the same: f is a polymorphism of Γ if it is a polymorphism of every relation from Γ .

Let $H = \{H_i\}_{i \in [k]}$ be a multi-sorted set. A collection of mappings $\{e_i^{s,r}\}_{i \in [k]}$, $s \in [r]$, is said to be the s 'th r -ary *projection* if $e_i^{s,r}(a_1, \dots, a_r) = a_s$ for all $a_1, \dots, a_r \in H_i$ and $i \in [k]$. Projections are polymorphisms of any relation, structure, or constraint language.

A particular type of polymorphisms plays an outsized role in counting CSP research. A ternary polymorphism m of a multi-sorted relational structure $\mathcal{H} = (H; \mathcal{R}_1, \dots, \mathcal{R}_\ell)$, $H = \{H_i\}_{i \in [k]}$, (a constraint language Γ over H) is said to be *Mal'tsev* if for every $i \in [k]$ the mapping m_i satisfies the equations $m_i(x, y, y) = m_i(y, y, x) = x$. If \mathcal{H} has a Mal'tsev polymorphism, the set of solutions of any instance \mathcal{G} of $\text{CSP}(\mathcal{H})$ admits a compact representation by a set of solutions whose number is linear in the size of \mathcal{G} , see, [11, 23]. Under certain conditions this allows to solve $\# \text{CSP}(\mathcal{H})$ in polynomial time.

One other concept that we will refer to in this paper is the *rectangularity*. A binary relation $\mathcal{R} \subseteq H_1 \times H_2$ is called *rectangular* if $(a, c), (a, d), (b, c) \in \mathcal{R}$ implies $(b, d) \in \mathcal{R}$ for any $a, b \in H_1, c, d \in H_2$. A relation $\mathcal{R} \subseteq H_{i_1} \times \dots \times H_{i_n}$ for $n \geq 2$ is rectangular if for every $\emptyset \neq I \subsetneq [n]$, the relation \mathcal{R} is rectangular when viewed as a binary relation, a subset of $\text{pr}_I \mathcal{R} \times \text{pr}_{[n]-I} \mathcal{R}$. A relational structure \mathcal{H} is *strongly rectangular* if every relation $\mathcal{R} \in \langle \mathcal{H} \rangle$ of arity at least 2 is rectangular. The following lemma provides a connection between strong rectangularity and Mal'tsev polymorphisms and was first observed in [34] although in a different language.

► **Lemma 7** ([34], see also [23]). *A relational structure is strongly rectangular if and only if it has a Mal'tsev polymorphism.*

We introduce a modular version of this concept, strongly p -rectangular structures. A relational structure \mathcal{H} is said to be strongly p -rectangular, if every $\mathcal{R} \in \langle \mathcal{H} \rangle_p$ is rectangular. It is shown in [41] that a relational structure can be strongly rectangular, but not strongly p -rectangular.

Also, it was shown in [41] that if a relational structure is strongly 2-rectangular and admits a Mal'tsev polymorphism, then the corresponding modular counting CSP problem modulo 2 is tractable. This result highlights the algorithmic significance of Mal'tsev polymorphisms: their presence enables efficient algorithms, while their absence plays a central role in establishing hardness.

► **Theorem 8** ([41, 42]). *Let \mathcal{H} be a 2-rigid, strongly 2-rectangular multi-sorted relational structure, and $\langle \mathcal{H} \rangle_2$ has a Mal'tsev polymorphism. Then $\#_2\text{CSP}(\mathcal{H})$ can be solved in time $O(n^5)$.*

Another part polymorphisms will be playing in this paper is their connection to automorphisms of powers of structures. Let \mathcal{H} be a multi-sorted structure with the base set $H = \{H_i\}_{i \in [k]}$ and \mathcal{H}^r its direct power. A mapping $f : \mathcal{H}^r \rightarrow \mathcal{H}^r$ of \mathcal{H}^r to itself is a collection of mappings $\{f_i\}_{i \in [k]} : H_i^r \rightarrow H_i^r$ each of which can also be represented as $f_i = (f_i^1, \dots, f_i^r)$, where $f_i^j : H_i^r \rightarrow H_i$. Rearranging these mapping we obtain collections $f^j = \{f_i^j\}_{i \in [k]}$, $j \in [r]$, where each f^j is a mapping of H^r to H . The following lemma is straightforward from the definitions.

► **Lemma 9.** *In the notation above f is a homomorphism of \mathcal{H}^r to itself if and only if f^j is a polymorphism of \mathcal{H} for every $j \in [r]$.*

Clearly, f is an automorphism of \mathcal{H}^r if and only if the f^j are such that f is bijective.

3 Automorphic Polynomials

In this section we introduce a new automorphism-like construction that allows one to reduce $\#_p\text{CSP}(\mathcal{H})$ to a CSP over a smaller structure that we will extensively use in the future. A binary polymorphism f of a (multi-sorted) relational structure $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1, \dots, \mathcal{R}_\ell)$ is said to be an *automorphic polynomial* if for any domain H_i of \mathcal{H} and any $a \in H_i$, $f_i(a, x)$ viewed as a mapping from H_i to H_i is a permutation of H_i . If, for a prime p , for any domain H_i and any $a \in H_i$, $f_i(a, x)$ is the identity mapping or a permutation of order p , and $f_i(a, x)$ is a permutation of order p for at least one domain H_i and at least one $a \in H_i$, f is said to be a *p -automorphic polynomial*.

Let $\mathcal{H} = (\{H_i\}_{i \in [k]}, \mathcal{R}_1, \dots, \mathcal{R}_\ell)$ and f its p -automorphic polynomial. Let also $i \in [k]$ and $a \in H_i$ be such that $f_i(a, x)$ is a permutation of H_i of order p . Then a structure \mathcal{H}' is constructed as follows:

- The domains are the same as those of \mathcal{H} , except H_i . The domain H_i is replaced with two domains: $H'_i = H_i - \{a\}$ and $H''_i = H_i - B$, where B is the union of all nontrivial orbits of $f_i(a, x)$.
- Every relation \mathcal{R}_j containing i in its type is replaced with \mathcal{R}'_j and \mathcal{R}''_j that are obtained from \mathcal{R}_j by replacing H_i with H'_i (respectively H''_i) restricting to tuples that do not contain a (respectively, do not contain elements from B).

We repeat this construction for every H_i and $a \in H_i$ such that $f_i(a, x)$ is a permutation of H_i of order p . The resulting structure is denoted \mathcal{H}^f .

► **Proposition 10.** *Suppose that a relational structure \mathcal{H} has a p -automorphic polynomial f and \mathcal{H}^f is as above. Then $\text{Sp}(\mathcal{H}^f) \preceq \text{Sp}(\mathcal{H})$ and $\#_p\text{CSP}(\mathcal{H})$ is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H}^f)$.*

Proof (sketch). Let $\mathcal{H} = (\{H_i\}_{i \in [k]}, \mathcal{R}_1, \dots, \mathcal{R}_\ell)$. The first claim of the proposition is obvious, so we focus on the second claim. We will construct an auxiliary problem from an instance $\mathcal{P} = (V, \tau, \mathcal{C})$ of $\#_p\text{CSP}(\mathcal{H})$. Let $C = \langle \mathbf{s}_C, \mathcal{R}_{j_C} \rangle$ be a constraint in \mathcal{P} and $\text{Sym}(\mathcal{R}_{j_C})$ the symmetric group on \mathcal{R}_{j_C} as a set. We introduce an instance $s(\mathcal{P})$ with domains $\text{Sym}(\mathcal{R}_{j_C})$, constraints that are derivative from the constraints of \mathcal{P} and will allow us to conclude that either the p -automorphic polynomial f can be used to reduce \mathcal{P} to a problem with smaller domains that are parts of \mathcal{H}^f , or that \mathcal{P} does not have solutions that are equal to certain elements of the domains and therefore, again, can be reduced to a problem with smaller domains.

More precisely, $s(\mathcal{P}) = (V', \tau', \mathcal{C}')$ is constructed as follows.

- $V' = \{v_C \mid C = \langle \mathbf{s}_C, \mathcal{R}_{j_C} \rangle \in \mathcal{C}\}$ and the domain of $v_C \in V'$ is $\text{Sym}(\mathcal{R}_{j_C})$, that is, $\tau'(v_C) = j_C$;
- for any constraints $C_1, C_2 \in \mathcal{C}$, $C_q = \langle \mathbf{s}_{C_q}, \mathcal{R}_{j_{C_q}} \rangle$ such that $\mathbf{s}_{C_1}[s] = \mathbf{s}_{C_2}[t]$ for some s, t , introduce the constraint $\langle (v_{C_1}, v_{C_2}), \mathcal{S}_{st}^{uv} \rangle \in \mathcal{C}'$, where $u = j_{C_1}, v = j_{C_2}$,

$$\mathcal{S}_{st}^{uv} = \{(\varphi_1, \varphi_2) \mid \varphi_q \in \text{Sym}(\mathcal{R}_{j_{C_q}}), q \in \{1, 2\}, \text{ and } \varphi_1(\mathbf{a}_1)[s] = \varphi_2(\mathbf{a}_2)[t] \\ \text{for any } \mathbf{a}_1 \in \mathcal{R}_{j_{C_1}}, \mathbf{a}_2 \in \mathcal{R}_{j_{C_2}} \text{ such that } \mathbf{a}_1[s] = \mathbf{a}_2[t]\}.$$

For an instance $\mathcal{P} = (V, \tau, \mathcal{C})$ of $\text{CSP}(\mathcal{H})$, a collection $\{\varphi_C \mid C \in \mathcal{C}\}$ of permutations $\varphi_C : \mathcal{R}_{j_C} \rightarrow \mathcal{R}_{j_C}$ is said to be a *consistent collection of permutations* if for any $C_1, C_2, C_q = \langle \mathbf{s}_{C_q}, \mathcal{R}_{j_{C_q}} \rangle$, $q \in \{1, 2\}$ and any $\mathbf{a}_1 \in \mathcal{R}_{j_{C_1}}, \mathbf{a}_2 \in \mathcal{R}_{j_{C_2}}$ such that $\text{pr}_{\mathbf{s}_{C_1} \cap \mathbf{s}_{C_2}} \mathbf{a}_1 = \text{pr}_{\mathbf{s}_{C_1} \cap \mathbf{s}_{C_2}} \mathbf{a}_2$ it holds that $\text{pr}_{\mathbf{s}_{C_1} \cap \mathbf{s}_{C_2}} \varphi_{C_1}(\mathbf{a}_1) = \text{pr}_{\mathbf{s}_{C_1} \cap \mathbf{s}_{C_2}} \varphi_{C_2}(\mathbf{a}_2)$. It can be shown that each solution of $s(\mathcal{P})$ provides a collection $\varphi = \{\varphi_C\}_{C \in \mathcal{C}}$ of consistent permutations for \mathcal{P} .

Finding solutions to $s(\mathcal{P})$ can also be done in polynomial time, because as every $\text{Sym}(\mathcal{R}_j)$ is a group, we can introduce a Mal'tsev operation $m = \{m_j\}_{j \in [\ell]}$ on the collection of $\text{Sym}(\mathcal{R}_j)$, $j \in [\ell]$, as follows: $m_j(x, y, z) = xy^{-1}z$, where multiplication and inverse are as in the group $\text{Sym}(\mathcal{R}_j)$. Then for any $u, v \in [\ell]$ and any $s \in [\text{ar}(\mathcal{R}_u)], t \in [\text{ar}(\mathcal{R}_v)]$ such that the types of the corresponding coordinates match, the mapping m is a polymorphism of \mathcal{S}_{st}^{uv} . By the results of [11] for any instance \mathcal{P} of $\#_p\text{CSP}(\mathcal{H})$ the problem $s(\mathcal{P})$ can be solved in polynomial time. Moreover, for any $C_0 = \langle \mathbf{s}_0, \mathcal{R}_{j_0} \rangle \in \mathcal{C}$ and $\varphi_0 \in \text{Sym}(\mathcal{R}_{j_0})$, we can verify in polynomial time whether or not there is a consistent collection of mappings $\varphi = \{\varphi_C\}_{C \in \mathcal{C}}$ such that $\varphi_{C_0} = \varphi_0$. With that in mind the following transformation of \mathcal{P} can be performed in polynomial time.

Now, let \mathcal{H}' be constructed as explained before the proposition. Let $C_0 = \langle \mathbf{s}_0, \mathcal{R}_{j_0} \rangle \in \mathcal{C}$ be such that for some $v \in \mathbf{s}_0$ the type of v is H_i , say, $\mathbf{s}_0[s] = v$, and there is $\mathbf{a} \in \mathcal{R}_{j_0}$ with $\mathbf{a}[s] = a$. If there are no such C_0 and \mathbf{a} , then H_i can be either completely eliminated or it can be reduced by removing a from it. Thus, \mathcal{P} itself can be considered as an instance of $\#_p\text{CSP}(\mathcal{H}')$. Otherwise let $\varphi_0 \in \text{Sym}(\mathcal{R}_{j_0})$ be given by $\varphi_0(\mathbf{x}) = f(\mathbf{a}, \mathbf{x})$. It is possible to verify whether or not there exists a solution φ of $s(\mathcal{P})$ such that $\varphi_{C_0} = \varphi_0$. If such a solution exists, H_i can be reduced by eliminating all the nontrivial orbits of $f_i(a, x)$, since for any solution ψ of \mathcal{P} with $\psi(v)$ belonging to such an orbit, the mappings $\varphi(\psi), \varphi^2(\psi), \dots, \varphi^{p-1}(\psi)$ are also different solutions of \mathcal{P} and together they contribute 0 modulo p into the number of solutions of \mathcal{P} . Suppose such a solution does not exist. As is easily seen, if there is a solution ψ of \mathcal{P} such that $\psi(\mathbf{s}_0) = \mathbf{a}$, then the mappings $\varphi_C \in \text{Sym}(\mathcal{R}_{j_C})$, $C = \langle \mathbf{s}, \mathcal{R}_{j_C} \rangle \in \mathcal{C}$, given by $\varphi_C(\mathbf{x}) = f(\psi(\mathbf{s}), \mathbf{x})$, form a solution of $s(\mathcal{P})$ with $\varphi_{C_0} = \varphi_0$. Therefore, there is no such solution ψ and the tuple \mathbf{a} can be removed from \mathcal{R}_{j_0} . Repeating this procedure for every $\mathbf{a} \in \mathcal{R}_{j_0}$ with $\mathbf{a}[s] = a$ we can either reduce the domain of v by eliminating nontrivial orbits of $f_i(a, x)$, or by eliminating a . In both cases after applying this transformation to all constraints of \mathcal{P} we obtain an instance of $\#_p\text{CSP}(\mathcal{H}')$.

Finally, repeating this construction for every H_i and $a \in H_i$ such that $f_i(a, x)$ is a permutation of H_i of order p we arrive to an instance the required problem $\#_p\text{CSP}(\mathcal{H}^f)$. ◀

While Proposition 10 may help to prove that $\#_p\text{CSP}(\mathcal{H})$ is solvable in polynomial time, in order to prove hardness we also need the reverse reduction, which exists only under additional assumptions. The case we need is the following.

► **Corollary 11.** *Let \mathcal{H} be a 3-element single-sorted structure such that there is a 2-automorphic polynomial f of \mathcal{H} and for $a \in H$ the mapping $f(a, x)$ is a permutation of H of order 2. Then, if $\{a\}$ and $H - \{a\}$ are 2-subalgebras of \mathcal{H} , the structure \mathcal{H}^f from Proposition 10 can be chosen such that $\#_p\text{CSP}(\mathcal{H})$ and $\#_p\text{CSP}(\mathcal{H}^f)$ are polynomial time interreducible.*

Proof. In order to prove Corollary 11 we only need to show that the problem $\#_p\text{CSP}(\mathcal{H}^f)$ is polynomial time reducible to $\#_p\text{CSP}(\mathcal{H})$. Let \mathcal{R}_j be a relation of \mathcal{H} of arity r . Then

$$\mathcal{R}'_j = \mathcal{R}_j \cap (\{a\} \times \cdots \times \{a\}) \quad \text{and} \quad \mathcal{R}''_j = \mathcal{R}_j \cap ((H - \{a\}) \times \cdots \times (H - \{a\})).$$

Since $\{a\}$ and $H - \{a\}$ are 2-subalgebras of \mathcal{H} , the relations $\mathcal{R}'_j, \mathcal{R}''_j$ are p -mpp-definable in \mathcal{H} . The result follows from Proposition 6. ◀

► **Example 12.** Let $\mathcal{H} = (H; \mathcal{R}, C_{H,a}, a \in H)$, where $H = \{0, \dots, p-1, p, p+1\}$ for some prime p , \mathcal{R} is the binary relation $H^2 - \{(0, p), \dots, (p-1, p)\}$, and $C_{H,a}$ is the constant relation $\{a\}$. Since any automorphism of \mathcal{H} must preserve $C_{H,a}$ for each $a \in H$, the structure is rigid. However, it has a p -automorphic polynomial f given by

$$\begin{aligned} f(p, i) &= f(p+1, i) = i+1 \pmod{p}, & \text{for } i \in \{0, \dots, p-1\}, \text{ and} \\ f(x, y) &= y & \text{otherwise.} \end{aligned}$$

If we now apply Proposition 10, the problem $\#_p\text{CSP}(\mathcal{H})$ can be reduced to $\#_p\text{CSP}(\mathcal{H}^f)$, where \mathcal{H}^f is a multisorted structure with domains $\{p, p+1\}$, $\{0, \dots, p-1\}$, and the 1-element domains. The relations of \mathcal{H}^f are all trivial, and so $\#_p\text{CSP}(\mathcal{H}^f)$ is polynomial time solvable.

4 Tools for Reduction

This section introduces the main technical tools used in our reductions. First, we define indicator problems and explain how the existence of a Mal'tsev polymorphism relates to the presence of a specific predicate in the set of all (multi-sorted) pp-definable relations. Then, we apply all the constructions and tools that we developed in this section to two special classes of relational structures: p -conservative and three-element structures.

4.1 Indicator Problem and Indicator Obstruction

In this section, we show that in many cases, if a relational structure \mathcal{H} is not strongly p -rectangular, then there is a rectangularity obstruction of a very specific kind. Recall that ([38]) for a (multi-sorted) relational structure $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1, \dots, \mathcal{R}_\ell)$, the *ternary indicator problem* $\mathcal{I}_3(\mathcal{H})$ is an instance of $\text{CSP}(\mathcal{H})$ that characterizes the ternary polymorphisms of \mathcal{H} .

- The set of variables V is the set $H^{(3)} = H_1^3 \cup \cdots \cup H_k^3$ of all triples over the domains of \mathcal{H} , i.e., for each triple $\mathbf{a} = (\mathbf{a}[1], \mathbf{a}[2], \mathbf{a}[3]) \in H_i^3$, the set V contains a variable $x_{\mathbf{a}}$ with domain H_i .
- For every s -ary relation \mathcal{R}_j of type (i_1, \dots, i_s) from \mathcal{H} and for any tuples $\mathbf{a}_1, \dots, \mathbf{a}_s \in H^{(3)}$, where each $\mathbf{a}_q = (\mathbf{a}_q[1], \mathbf{a}_q[2], \mathbf{a}_q[3]) \in H_{i_q}^3$, we include the constraint $\langle (x_{\mathbf{a}_1}, \dots, x_{\mathbf{a}_s}), \mathcal{R}_j \rangle$ in the instance $\mathcal{I}_3(\mathcal{H})$ if for every $t \in [3]$, the tuple $(\mathbf{a}_1[t], \dots, \mathbf{a}_s[t])$ belongs to \mathcal{R}_j . That is, the projection of the sequence $(\mathbf{a}_1, \dots, \mathbf{a}_s)$ onto each coordinate $t \in [3]$ is a tuple in \mathcal{R}_j .

As is easily seen, every solution of $\mathcal{I}_3(\mathcal{H})$ defines a ternary multi-sorted operation $\{f_i\}_{i \in [k]}$ with $f_i : H_i^3 \rightarrow H_i$.

► **Example 13.**

(a) Let \mathcal{H}_1 be a relational structure with only one domain $H = \{0, 1\}$ and one (binary) relation $\mathcal{R} = \{(0, 1), (1, 0)\}$. Then $\mathcal{I}_3(\mathcal{H}_1)$ contains 8 variables $x_{000}, x_{001}, x_{010}, x_{011}, x_{100}, x_{101}, x_{110}, x_{111}$, that is, all possible triples of elements of \mathcal{H} , and the constraints are imposed on every pair (x_{abc}, x_{def}) such that $(a, d), (b, e), (c, f) \in \mathcal{R}$. In this case this means all pairs of “dual” variables of the form $(x_{abc}, x_{\neg a \neg b \neg c})$. Thus, the constraints are

$$\langle\langle x_{000}, x_{111} \rangle, \mathcal{R}\rangle, \langle\langle x_{001}, x_{110} \rangle, \mathcal{R}\rangle, \langle\langle x_{010}, x_{101} \rangle, \mathcal{R}\rangle, \langle\langle x_{011}, x_{100} \rangle, \mathcal{R}\rangle.$$

(b) For a structure \mathcal{H}_2 let the domain H be the same set $\{0, 1\}$, but the only binary relation is given by $\mathcal{Q} = \{(0, 0), (0, 1), (1, 0)\}$. The set of variables of $\mathcal{I}_3(\mathcal{H}_2)$ is the same as before, but the constraints are very different. As is easily seen, the constraints in this case are

$$\begin{array}{cccccc} \langle\langle x_{000}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{000}, x_{001} \rangle, \mathcal{Q}\rangle & \langle\langle x_{000}, x_{010} \rangle, \mathcal{Q}\rangle & \langle\langle x_{000}, x_{011} \rangle, \mathcal{Q}\rangle & \langle\langle x_{000}, x_{100} \rangle, \mathcal{Q}\rangle & \langle\langle x_{000}, x_{101} \rangle, \mathcal{Q}\rangle \\ \langle\langle x_{000}, x_{110} \rangle, \mathcal{Q}\rangle & \langle\langle x_{000}, x_{111} \rangle, \mathcal{Q}\rangle & \langle\langle x_{001}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{001}, x_{010} \rangle, \mathcal{Q}\rangle & \langle\langle x_{001}, x_{100} \rangle, \mathcal{Q}\rangle & \langle\langle x_{001}, x_{110} \rangle, \mathcal{Q}\rangle \\ \langle\langle x_{010}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{010}, x_{001} \rangle, \mathcal{Q}\rangle & \langle\langle x_{010}, x_{100} \rangle, \mathcal{Q}\rangle & \langle\langle x_{010}, x_{101} \rangle, \mathcal{Q}\rangle & \langle\langle x_{100}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{100}, x_{001} \rangle, \mathcal{Q}\rangle \\ \langle\langle x_{100}, x_{010} \rangle, \mathcal{Q}\rangle & \langle\langle x_{100}, x_{011} \rangle, \mathcal{Q}\rangle & \langle\langle x_{011}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{011}, x_{100} \rangle, \mathcal{Q}\rangle & \langle\langle x_{101}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{101}, x_{010} \rangle, \mathcal{Q}\rangle \\ \langle\langle x_{110}, x_{000} \rangle, \mathcal{Q}\rangle & \langle\langle x_{110}, x_{001} \rangle, \mathcal{Q}\rangle & \langle\langle x_{111}, x_{000} \rangle, \mathcal{Q}\rangle & & & \end{array}$$

┘

► **Lemma 14** ([38]). *For a (multi-sorted) relational structure \mathcal{H} , a ternary (multi-sorted) operation on the domains of \mathcal{H} is a solution of $\mathcal{I}_3(\mathcal{H})$ if and only if it is a ternary polymorphism of \mathcal{H} .*

We will call the set of all solutions of $\mathcal{I}_3(\mathcal{H})$ the *indicator predicate* $\Upsilon_3(\mathcal{H})$. It is not difficult to see that $\Upsilon_3(\mathcal{H})$ is conjunctive-definable in \mathcal{H} . Indeed, let \mathcal{C} be the set of all constraints as defined above in $\mathcal{I}_3(\mathcal{H})$. Then we define the indicator predicate as the conjunction of all those constraints

$$\Upsilon_3(\mathcal{H})(\mathbf{x}) = \bigwedge_{\langle \mathbf{s}, \mathcal{R}_j \rangle \in \mathcal{C}} \mathcal{R}_j(\mathbf{s}),$$

where \mathbf{x} is the tuple consisting of all variables $x_{(a,b,c)}$ indexed by triples $(a, b, c) \in H^{(3)}$.

We now define certain coordinate sets used to analyze $\Upsilon_3(\mathcal{H})$. Let

$$I_{\mathcal{H}} = \{(a, b, b) \mid a, b \in H_i, i \in [k]\}, \quad J_{\mathcal{H}} = \{(b, b, a) \mid a, b \in H_i, i \in [k], a \neq b\}.$$

Note that the diagonal triples (a, a, a) are included in $I_{\mathcal{H}}$. Enumerate these sets as $I_{\mathcal{H}} = \{(a_1, b_1, b_1), \dots, (a_N, b_N, b_N)\}$, $J_{\mathcal{H}} = \{(c_1, c_1, d_1), \dots, (c_M, c_M, d_M)\}$, and define tuples

$$\mathbf{a}_{\mathcal{H}} = (a_1, \dots, a_N), \quad \mathbf{b}_{\mathcal{H}} = (b_1, \dots, b_N), \quad \mathbf{c}_{\mathcal{H}} = (c_1, \dots, c_M), \quad \mathbf{d}_{\mathcal{H}} = (d_1, \dots, d_M).$$

Then $\mathbf{a}_{\mathcal{H}}, \mathbf{b}_{\mathcal{H}} \in \text{pr}_{I_{\mathcal{H}}} \Upsilon_3(\mathcal{H})$, and $\mathbf{c}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}} \in \text{pr}_{J_{\mathcal{H}}} \Upsilon_3(\mathcal{H})$. Also,

$$(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H}).$$

Moreover, \mathcal{H} admits a Mal'tsev polymorphism if and only if $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H})$.

Indeed, this condition tests whether the tuple $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}})$, which represents applying the Mal'tsev term coordinate-wise to the elements of each triple from $I_{\mathcal{H}} \cup J_{\mathcal{H}}$, belongs to the projection of the relation $\Upsilon_3(\mathcal{H})$. That is, since $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H})$, the closure under a Mal'tsev polymorphism would imply that the corresponding output tuple, $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}})$, must also be in the relation. So, its membership characterizes whether such a polymorphism exists.

► **Example 15.** Let us reconsider the structures $\mathcal{H}_1, \mathcal{H}_2$ from Example 13 and their indicator problems. The following tables contain all the solutions of $\mathcal{I}_3(\mathcal{H}_1)$ and $\mathcal{I}_3(\mathcal{H}_2)$, that is, the tuples of $\Upsilon_3(\mathcal{H}_1), \Upsilon_3(\mathcal{H}_2)$. The rows are labeled with the variables of $\mathcal{I}_3(\mathcal{H}_1), \mathcal{I}_3(\mathcal{H}_2)$.

$\Upsilon_3(\mathcal{H}_1) :$	$\Upsilon_3(\mathcal{H}_2) :$																																																																																																																																																																																																																																																																																																																
<table style="border-collapse: collapse;"> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{000}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{001}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{010}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{011}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{100}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{101}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{110}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{111}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td></tr> </table>	x_{000}	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	x_{001}	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	1	1	x_{010}	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	x_{011}	0	1	1	0	1	0	0	1	0	0	0	1	1	1	1	0	0	0	x_{100}	1	0	0	1	0	1	1	0	1	1	1	0	0	0	0	0	1	0	x_{101}	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	x_{110}	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	0	x_{111}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	<table style="border-collapse: collapse;"> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{000}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{001}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{010}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{011}</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{100}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{101}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{110}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td></tr> <tr><td style="border-right: 1px solid black; padding: 2px 5px;">x_{111}</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">0</td><td style="padding: 2px 5px;">1</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td><td style="padding: 2px 5px;">*</td></tr> </table>	x_{000}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x_{001}	0	0	1	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0	x_{010}	0	1	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	x_{011}	0	1	1	0	0	*	*	0	*	0	*	*	0	*	*	*	*	*	x_{100}	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	x_{101}	1	0	1	0	0	*	0	*	0	*	*	0	*	*	*	*	*	*	x_{110}	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	*	*	*	x_{111}	1	1	1	0	1	*	*	*	*	*	*	*	*	*	*	*	*	*
x_{000}	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1																																																																																																																																																																																																																																																																																															
x_{001}	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	1	1																																																																																																																																																																																																																																																																																															
x_{010}	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1																																																																																																																																																																																																																																																																																															
x_{011}	0	1	1	0	1	0	0	1	0	0	0	1	1	1	1	0	0	0																																																																																																																																																																																																																																																																																															
x_{100}	1	0	0	1	0	1	1	0	1	1	1	0	0	0	0	0	1	0																																																																																																																																																																																																																																																																																															
x_{101}	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0																																																																																																																																																																																																																																																																																															
x_{110}	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	0																																																																																																																																																																																																																																																																																															
x_{111}	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0																																																																																																																																																																																																																																																																																															
x_{000}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																																																																																																																																																																																																																																																																																															
x_{001}	0	0	1	0	0	1	0	0	1	0	0	1	0	1	0	0	0	0																																																																																																																																																																																																																																																																																															
x_{010}	0	1	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0																																																																																																																																																																																																																																																																																															
x_{011}	0	1	1	0	0	*	*	0	*	0	*	*	0	*	*	*	*	*																																																																																																																																																																																																																																																																																															
x_{100}	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0																																																																																																																																																																																																																																																																																															
x_{101}	1	0	1	0	0	*	0	*	0	*	*	0	*	*	*	*	*	*																																																																																																																																																																																																																																																																																															
x_{110}	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	*	*	*																																																																																																																																																																																																																																																																																															
x_{111}	1	1	1	0	1	*	*	*	*	*	*	*	*	*	*	*	*	*																																																																																																																																																																																																																																																																																															

Every column in these tables represents a ternary polymorphism of $\mathcal{H}_1, \mathcal{H}_2$, respectively. Note that the first three polymorphisms in both cases are the projections, their values equal the first, second, and third component of the triple representing the label of variables of the indicator problem. Projections are polymorphisms of every structure, and therefore are always solutions of the indicator problem.

The relation $\Upsilon_3(\mathcal{H}_1)$ contains every polymorphism $f(x, y, z)$ satisfying the condition $f(\neg x, \neg y, \neg z) = \neg f(x, y, z)$, also known as the condition of *self-duality*. The relation $\Upsilon_3(\mathcal{H}_2)$ cannot be described by such a simple condition. The stars in the table for $\Upsilon_3(\mathcal{H}_2)$ mean that no matter what values are in these positions, the tuple belongs to $\Upsilon_3(\mathcal{H}_2)$.

The sets $I_{\mathcal{H}_1} = I_{\mathcal{H}_2} = \{000, 011, 100, 111\}$ are equal in this case and so are $J_{\mathcal{H}_1} = J_{\mathcal{H}_2} = \{001, 110\}$. The special tuples $\mathbf{a}_{\mathcal{H}_1}, \mathbf{b}_{\mathcal{H}_1}, \mathbf{c}_{\mathcal{H}_1}, \mathbf{d}_{\mathcal{H}_1}$ and $\mathbf{a}_{\mathcal{H}_2}, \mathbf{b}_{\mathcal{H}_2}, \mathbf{c}_{\mathcal{H}_2}, \mathbf{d}_{\mathcal{H}_2}$ are also equal, as they are restrictions of the tuples corresponding to the same projections. Specifically,

$$\mathbf{a}_{\mathcal{H}_1} = \mathbf{a}_{\mathcal{H}_2} = (0, 0, 1, 1), \quad \mathbf{b}_{\mathcal{H}_1} = \mathbf{b}_{\mathcal{H}_2} = (0, 1, 0, 1), \quad \mathbf{c}_{\mathcal{H}_1} = \mathbf{c}_{\mathcal{H}_2} = (0, 1), \quad \mathbf{d}_{\mathcal{H}_1} = \mathbf{d}_{\mathcal{H}_2} = (1, 0).$$

As was observed, the presence of a Mal'tsev polymorphism is equivalent to the conditions

$$(\mathbf{a}_{\mathcal{H}_1}, \mathbf{d}_{\mathcal{H}_1}) \in \text{pr}_{I_{\mathcal{H}_1} \cup J_{\mathcal{H}_1}} \Upsilon_3(\mathcal{H}_1), \quad (\mathbf{a}_{\mathcal{H}_2}, \mathbf{d}_{\mathcal{H}_2}) \in \text{pr}_{I_{\mathcal{H}_2} \cup J_{\mathcal{H}_2}} \Upsilon_3(\mathcal{H}_2),$$

that is $(0, 0, 1, 1, 1, 0) \in \text{pr}_{I_{\mathcal{H}_1} \cup J_{\mathcal{H}_1}} \Upsilon_3(\mathcal{H}_1), \text{pr}_{I_{\mathcal{H}_2} \cup J_{\mathcal{H}_2}} \Upsilon_3(\mathcal{H}_2)$. It is not difficult to see that such a tuple exists in $\text{pr}_{I_{\mathcal{H}_1} \cup J_{\mathcal{H}_1}} \Upsilon_3(\mathcal{H}_1)$, but not in $\text{pr}_{I_{\mathcal{H}_2} \cup J_{\mathcal{H}_2}} \Upsilon_3(\mathcal{H}_2)$. \square

While $\Upsilon_3(\mathcal{H})$ characterizes the ternary polymorphisms of \mathcal{H} , we now define a construction that characterizes the ternary polymorphisms of $\langle \mathcal{H} \rangle_p$.

Note that every polymorphism of $\langle \mathcal{H} \rangle_p$ is also a polymorphism of \mathcal{H} . For any ternary polymorphism f of \mathcal{H} that is not a polymorphism of $\langle \mathcal{H} \rangle_p$, there exists a relation $\mathcal{Q}_f \in \langle \mathcal{H} \rangle_p$ such that f fails to preserve \mathcal{Q}_f . Let $\mathcal{H}^{\dagger p}$ be the expansion of \mathcal{H} by all such relations $\mathcal{H}^{\dagger p} = \mathcal{H} + \{\mathcal{Q}_f \mid f \in \text{Pol}_3(\mathcal{H}) - \text{Pol}_3(\langle \mathcal{H} \rangle_p)\}$. The following lemma is straightforward.

► **Lemma 16.** *Let \mathcal{H} be a (multi-sorted) relational structure. Then*

1. $\Upsilon_3(\mathcal{H})$ is conjunctive-definable in \mathcal{H} ,
2. $\Upsilon_3(\mathcal{H}^{\dagger p})$ is p -mpp-definable in \mathcal{H} .

If $\langle \mathcal{H} \rangle_p$ does not have a Mal'tsev polymorphism, then $\mathcal{H}^{\dagger p}$ also does not have a Mal'tsev polymorphism. Therefore, $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H}^{\dagger p})$ and $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \notin \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H}^{\dagger p})$.

A relation $\mathcal{R} \subseteq \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H}^{\dagger p})$ such that $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \mathcal{R}$ and $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \notin \mathcal{R}$ will be called a p -indicator rectangularity obstruction. Clearly, if $\langle \mathcal{H} \rangle_p$ has no Mal'tsev polymorphism, $\text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H}^{\dagger p})$ itself is a p -indicator rectangularity obstruction.

However, it may not be p -mpp-definable in \mathcal{H} . A relational structure \mathcal{H} is said to *admit a p -indicator rectangularity obstruction* if there exists a p -indicator rectangularity obstruction \mathcal{R} that is p -mpp-definable in \mathcal{H} .

Recall that for a multisorted structure \mathcal{H} with domains H_1, \dots, H_k we use $H^{(3)}$ to denote $H_1^3 \cup \dots \cup H_k^3$. An important ingredient in our construction is the study of automorphisms of $\mathcal{H}^{(3)}$ that fix certain structured tuples. Specifically, those of the form (a, b, b) and (b, b, a) . We refer to such automorphisms as *M -automorphisms*, with “ M ” standing for Mal’tsev.

The motivation for working with $\mathcal{H}^{(3)}$ comes from the need to apply our reduction tools in a higher-dimensional setting. While every automorphism of $\mathcal{H}^{(3)}$ induces a polymorphism of \mathcal{H} via its coordinate projections, the converse does not hold: not every collection of polymorphisms arises as projections of a single automorphism of $\mathcal{H}^{(3)}$. The notion of M -automorphisms captures precisely those symmetries of $\mathcal{H}^{(3)}$ that preserve the key tuples required for our connection to Mal’tsev polymorphisms. This connection will become essential later.

Let $\mathbf{x} = (x_i)_{i \in \mathcal{H}^{(3)}}$ be the tuple of variables indexed by the elements of $H^{(3)}$. For a subset $K \subseteq \mathcal{H}^{(3)}$, we use the notation $\mathbf{x}|_K$ to denote the subtuple $(x_i)_{i \in K}$, that is, the variables whose indices belong to K . The next lemma identifies conditions sufficient for \mathcal{H} to admit a p -indicator rectangularity obstruction. Define $E_{\mathcal{H}} = H^{(3)} - (I_{\mathcal{H}} \cup J_{\mathcal{H}})$ and $m = \text{ar}(\Upsilon_3(\mathcal{H}))$.

► **Lemma 17.** *Let $\mathcal{R}(\mathbf{x}|_{H^{(3)}-K}) = \exists \mathbf{x}|_K \Upsilon_3(\mathcal{H}^{\dagger p})(\mathbf{x}) \wedge \Phi(\mathbf{x}|_K)$, where $K \subseteq E_{\mathcal{H}}$, and Φ is a conjunction of unary predicates on variables from K such that $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \mathcal{R}$. Let also $y \in E_{\mathcal{H}} - K$ and let*

$$B_1 = \text{pr}_y \text{Ext}_{\mathcal{R}}(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), \quad B_2 = \text{pr}_y \text{Ext}_{\mathcal{R}}(\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), \quad B_3 = \text{pr}_y \text{Ext}_{\mathcal{R}}(\mathbf{c}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}).$$

If $B_1 \times B_2 \times B_3$ contains a triple that is a fixed point of every M -automorphism of $(\mathcal{H}^{\dagger p})^3$ then there is a p -mpp-definable relation $\mathcal{Q} \subseteq \text{pr}_{H^{(3)}-(K \cup \{y\})} \mathcal{R}$ in $\langle \mathcal{H} + \{\mathcal{R}\} \rangle_p$ such that $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \mathcal{Q}$, and

$$|\text{pr}_y(\text{Ext}_{\mathcal{Q}}((\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}})))|, |\text{pr}_y(\text{Ext}_{\mathcal{Q}}((\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}})))|, |\text{pr}_y(\text{Ext}_{\mathcal{Q}}((\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}})))| \not\equiv 0 \pmod{p}.$$

Starting from $\Upsilon_3(\mathcal{H}^{\dagger p})$ by repeated application of Lemma 17 we can obtain a relation $\mathcal{R} \subseteq \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \Upsilon_3(\mathcal{H}^{\dagger p})$, which is a p -indicator rectangularity obstruction.

4.2 Conservative and 3-Element Structures

In this section we use the results of Sections 3 and 4.1 to essentially show that in the case of a p -conservative and 3-element structure \mathcal{H} (single-sorted in the latter case) either there is a Mal’tsev polymorphism of $\langle \mathcal{H} \rangle_p$ or \mathcal{H} admits a p -indicator rectangularity obstruction. The latter will later be used to prove the hardness of the corresponding modular counting problem.

Recall that a multi-sorted relational structure $\mathcal{H} = (\{H_i\}_{i \in [k]}; \mathcal{R}_1, \dots, \mathcal{R}_m)$ is said to be *p -conservative* if for every $i \in [k]$ and every subset $A \subseteq H_i$, the set A is p -mpp-definable in \mathcal{H} (as a unary relation).

► **Theorem 18.** *Let \mathcal{H} be a p -conservative structure. Then either $\langle \mathcal{H} \rangle_p$ has a Mal’tsev polymorphism, or \mathcal{H} admits a p -indicator rectangularity obstruction and therefore is not strongly p -rectangular.*

Proof (sketch). We assume that $\langle \mathcal{H} \rangle_p$ has no Mal’tsev polymorphism. By Lemma 16, it suffices to show that $\Upsilon_3(\mathcal{H}^{\dagger p})$ p -mpp-defines a p -indicator rectangularity obstruction.

Let z_1, \dots, z_L be an ordering of the variables from the set $E_{\mathcal{H}}$. We prove by induction that for any $q \in [L]$, there exists a relation $\mathcal{Q}_q \subseteq \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}} \cup E_q} \Upsilon_3(\mathcal{H}^{\dagger p})$, where $E_q = \{z_q, \dots, z_L\}$, such that $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \text{pr}_{I_{\mathcal{H}} \cup J_{\mathcal{H}}} \mathcal{Q}_q$. Note that \mathcal{Q}_{L+1} is the desired p -indicator rectangularity obstruction. For the base case, let $\mathcal{Q}_1 = \Upsilon_3(\mathcal{H}^{\dagger p})$, which satisfies the required condition, since $E_1 = \{z_1, \dots, z_L\}$.

Now suppose the statement holds for some q and that \mathcal{Q}_q is the corresponding relation. Consider the variable $z_q \in E_q$, and let its domain be H_i . Define the sets $B_1, B_2, B_3 \subseteq H_i$ as the sets of values appearing at coordinate z_q in the extensions of $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}})$, $(\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}})$, and $(\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}})$, respectively, within \mathcal{Q}_q , i.e.,

$$B_1 = \text{pr}_{z_q} \text{Ext}_{\mathcal{Q}_q}(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), \quad B_2 = \text{pr}_{z_q} \text{Ext}_{\mathcal{Q}_q}(\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), \quad B_3 = \text{pr}_{z_q} \text{Ext}_{\mathcal{Q}_q}(\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}).$$

Let D be a minimal set of representatives for B_1, B_2 , and B_3 ; clearly, $|D| \leq 3$. Let $\mathcal{Q}' = \mathcal{Q}_q(\mathbf{x}_{|I_{\mathcal{H}} \cup J_{\mathcal{H}} \cup E_q}) \wedge D(z_q)$. We then consider several cases depending on the cardinality of D and use Lemma 17 to show that z_q can be quantified away from \mathcal{Q}' via applying a p -modular quantifier to it. \blacktriangleleft

Next, we assume \mathcal{H} to be a 3-element (single-sorted) structure $(H, \mathcal{R}_1, \dots, \mathcal{R}_\ell)$, where $H = \{0, 1, 2\}$. The main result of this section is the following theorem.

► **Theorem 19.** *Let \mathcal{H} be a 3-element structure. Then one of the following three options holds*

- (1) $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism and therefore \mathcal{H} is strongly p -rectangular;
- (2) \mathcal{H} admits a p -indicator rectangularity obstruction;
- (3) $p = 2$ and \mathcal{H} has a p -automorphic polynomial f , there is $a \in H$ such that $f(a, x)$ is a permutation of order 2, and $H - \{a\}$ is a p -subalgebra of \mathcal{H} .

Proof (idea). In this case we follow the same approach as in the proof of Theorem 18, except in some cases it is not possible to show that z_q can be quantified away from \mathcal{Q}' . The reason for that is M-automorphisms of $(\mathcal{H}^{\dagger p})^3$ that act nontrivially on $B_1 \times B_2 \times B_3$. So, we prove that whenever such an M-automorphism is present, \mathcal{H} has a p -automorphic polynomial. This is done by a large case analysis using the Universal Algebra Calculator [29]. \blacktriangleleft

5 Complexity Classifications

We start this section with an (incomplete) complexity classification of $\#_p \text{CSP}(\mathcal{H})$, where \mathcal{H} is a p -conservative structure. The missing case is $p > 2$ and $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism. The gap is due the lack of understanding of the complexity of partition functions over nontrivial finite fields. It is a highly nontrivial problem that requires a separate study, see [39] for some partial results.

We start with proving hardness for structures whose domains are 2-element. The proposition below generalizes the results of Faben [25] to multisorted CSPs.

► **Proposition 20.** *Let \mathcal{H} be a multi-sorted relational structure, all of whose domains contain at most 2 elements. The problem $\#_p \text{CSP}(\mathcal{H})$ is solvable in polynomial time if and only if $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism. Otherwise it is $\#_p P$ -hard.*

Proof (idea). If $\langle \mathcal{H} \rangle_p$ does not have a Mal'tsev polymorphism, by Theorem 18 a p -indicator rectangularity obstruction \mathcal{R} is p -mpp-definable in \mathcal{H} . We start with the obstruction \mathcal{R} and then recursively quantify away coordinates of \mathcal{R} by applying p -modular quantifiers (which requires significant care) until we obtain a binary non-rectangular relation, for which hardness is known by [25].

Now, suppose that $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism. It is known that if a single-sorted 2-element relational structure has a Mal'tsev polymorphism then it also has the operation $x \oplus y \oplus z$ as a polymorphism [44], where \oplus is addition modulo 2. This means that for every domain H_i of \mathcal{H} there is a polymorphism m^i of $\langle \mathcal{H} \rangle_p$ that acts like $x \oplus y \oplus z$ if H_i is equipped with addition \oplus modulo 2. The main step of the proof is to show that all the m^i can be chosen the same. Therefore $\text{CSP}(\mathcal{H})$ can be represented by a system of linear equations modulo 2, for which the exact number of solutions can be found. ◀

► **Theorem 21.** *Let \mathcal{H} be a conservative structure and p a prime. If $\langle \mathcal{H} \rangle_p$ does not have a Mal'tsev polymorphism then $\#_p \text{CSP}(\mathcal{H})$ is $\#_p P$ -hard. If $\langle \mathcal{H} \rangle_2$ has a Mal'tsev polymorphism then $\#_2 \text{CSP}(\mathcal{H})$ is solvable in polynomial time.*

Proof. The second statement of the theorem follows from Theorem 8. For the first statement, by Theorem 18 a p -indicator rectangularity obstruction \mathcal{R} is p -mpp-definable in \mathcal{H} . Recall that $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \in \mathcal{R}$, while $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}}) \notin \mathcal{R}$. As \mathcal{H} is conservative, for every $v \in I_{\mathcal{H}}$ ($v \in J_{\mathcal{H}}$), $H_{i_v} = \{\mathbf{a}[v], \mathbf{b}[v]\}$ ($H_{i_v} = \{\mathbf{c}[v], \mathbf{d}[v]\}$) is a p -subalgebra of \mathcal{H} . Set

$$\mathcal{R}'(\mathbf{x}_{|I_{\mathcal{H}} \cup J_{\mathcal{H}}}) = \mathcal{R}(\mathbf{x}_{|I_{\mathcal{H}} \cup J_{\mathcal{H}}}) \wedge \bigwedge_{v \in I_{\mathcal{H}} \cup J_{\mathcal{H}}} H_{i_v}(\mathbf{x}_v).$$

The relation \mathcal{R}' still contains $(\mathbf{a}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{c}_{\mathcal{H}}), (\mathbf{b}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}})$ and does not contain $(\mathbf{a}_{\mathcal{H}}, \mathbf{d}_{\mathcal{H}})$. By Proposition 20 $\#_p \text{CSP}(\mathcal{R}')$ is $\#_p P$ -hard. ◀

Next, we consider the case when \mathcal{H} is a 3-element structure.

► **Theorem 22.** *Let \mathcal{H} be a 3-element structure and p a prime. If one of the following conditions holds:*

- (a) \mathcal{H} has a p -automorphic polynomial f and $\langle \mathcal{H}^f \rangle_p$ has a Mal'tsev polymorphism, or
- (b) \mathcal{H} does not have a p -automorphic polynomial, $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism, and $p = 2$,

then $\#_p \text{CSP}(\mathcal{H})$ is solvable in polynomial time. Also, if

- (1) \mathcal{H} has a p -automorphic polynomial f and $\langle \mathcal{H}^f \rangle_p$ does not have a Mal'tsev polymorphism, or
- (2) \mathcal{H} does not have a p -automorphic polynomial and $\langle \mathcal{H} \rangle_p$ does not have a Mal'tsev polymorphism,

then $\#_p \text{CSP}(\mathcal{H})$ is $\#_p P$ -complete.

Proof (idea). If \mathcal{H} has a p -automorphic polynomial then $p = 2$, and the result follows from Corollary 11 and Proposition 20. If $\langle \mathcal{H} \rangle_p$ has a Mal'tsev polymorphism and $p = 2$, the result follows from Theorem 8. Therefore, we assume that $\langle \mathcal{H} \rangle_p$ has no Mal'tsev polymorphism and \mathcal{H} does not have a p -automorphic polynomial. By Theorem 19 a p -indicator rectangularity obstruction \mathcal{R} is p -mpp-definable in \mathcal{H} . We prove that it is possible to p -mpp-define a binary non-rectangular relation showing that $\#_p \text{CSP}(\mathcal{H})$ is $\#_p P$ -complete. Unlike the case of p -conservative structures, it is not possible to restrict \mathcal{R} with arbitrary subsets of H , because they may not be p -subalgebras. We therefore use an approach similar to that in the proof of Theorem 21, except every possible combination of p -subalgebras of \mathcal{H} is considered separately. ◀

References

- 1 Libor Barto. The dichotomy for conservative constraint satisfaction problems revisited. In *LICS 2011*, pages 301–310, 2011. doi:10.1109/LICS.2011.25.
- 2 Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *J. ACM*, 61(1):3:1–3:19, 2014. doi:10.1145/2556646.
- 3 Libor Barto, Andrei Krokhin, and Ross Willard. Polymorphisms, and how to use them. In *Dagstuhl Follow-Ups*, volume 7. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/DFU.VOL7.15301.1.
- 4 Alexander I. Barvinok. *Combinatorics and Complexity of Partition Functions*, volume 30 of *Algorithms and Combinatorics*. Springer, 2016. doi:10.1007/978-3-319-51829-9.
- 5 Christoph Berkholz and Martin Grohe. Limitations of algebraic approaches to graph isomorphism testing. In *ICALP 2015*, volume 9134 of *LNCS*, pages 155–166. Springer, 2015. doi:10.1007/978-3-662-47672-7_13.
- 6 Andrei A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *J. ACM*, 53(1):66–120, 2006. doi:10.1145/1120582.1120584.
- 7 Andrei A. Bulatov. Complexity of conservative constraint satisfaction problems. *ACM Trans. Comput. Log.*, 12(4):24:1–24:66, 2011. doi:10.1145/1970398.1970400.
- 8 Andrei A. Bulatov. The complexity of the counting constraint satisfaction problem. *J. ACM*, 60(5):34:1–34:41, 2013. doi:10.1145/2528400.
- 9 Andrei A. Bulatov. Conservative constraint satisfaction re-visited. *J. Comput. Syst. Sci.*, 82(2):347–356, 2016. doi:10.1016/J.JCSS.2015.07.004.
- 10 Andrei A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *FOCS*, pages 319–330, 2017. doi:10.1109/FOCS.2017.37.
- 11 Andrei A. Bulatov and Víctor Dalmau. A simple algorithm for Mal’tsev constraints. *SIAM J. Comput.*, 36(1):16–27, 2006. doi:10.1137/050628957.
- 12 Andrei A. Bulatov and Víctor Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. *Information and Computation*, 205(5):651–678, 2007. doi:10.1016/J.IC.2006.09.005.
- 13 Andrei A. Bulatov, Martin E. Dyer, Leslie Ann Goldberg, Markus Jalsenius, Mark Jerrum, and David Richerby. The complexity of weighted and unweighted $\#CSP$. *J. Comp. Syst. Sci.*, 78(2):681–688, 2012. doi:10.1016/J.JCSS.2011.12.002.
- 14 Andrei A. Bulatov and Martin Grohe. The complexity of partition functions. *Theor. Comput. Sci.*, 348(2-3):148–186, 2005. doi:10.1016/J.TCS.2005.09.011.
- 15 Andrei A. Bulatov, Peter Jeavons, and Andrei A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM J. Comput.*, 34(3):720–742, 2005. doi:10.1137/S0097539700376676.
- 16 Andrei A. Bulatov and Amirhossein Kazeminia. Complexity classification of counting graph homomorphisms modulo a prime number. In *STOC*, pages 1024–1037, 2022. doi:10.1145/3519935.3520075.
- 17 Jin-Yi Cai and Xi Chen. Complexity of counting CSP with complex weights. *J. ACM*, 64(3), 2017. doi:10.1145/2822891.
- 18 Jin-yi Cai, Xi Chen, and Pinyan Lu. Graph homomorphisms with complex values: A dichotomy theorem. In *ICALP 2010*, volume 6198 of *LNCS*, pages 275–286. Springer, 2010. doi:10.1007/978-3-642-14165-2_24.
- 19 Jin-Yi Cai, Xi Chen, and Pinyan Lu. Nonnegative weighted $\#CSP$: An effective complexity dichotomy. *SIAM J. Comput.*, 45(6):2177–2198, 2016. doi:10.1137/15M1032314.
- 20 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. The complexity of complex weighted Boolean $\#CSP$. *J. Comput. Syst. Sci.*, 80(1):217–236, 2014. doi:10.1016/J.JCSS.2013.07.003.
- 21 Nadia Creignou and Miki Hermann. Complexity of generalized satisfiability counting problems. *Inf. Comput.*, 125(1):1–12, 1996. doi:10.1006/INCO.1996.0016.

- 22 Martin Dyer and Catherine Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17:260–289, 2000. doi:10.1002/1098-2418(200010/12)17:3/4%3C260::AID-RSA5%3E3.O.CO;2-W.
- 23 Martin Dyer and David Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM J on Comp*, 42(3):1245–1274, 2013. doi:10.1137/100811258.
- 24 Martin E. Dyer, Leslie Ann Goldberg, and Mark Jerrum. The complexity of weighted Boolean $\#$ CSP. *SIAM J. Comput.*, 38(5):1970–1986, 2009. doi:10.1137/070690201.
- 25 John Faben. The complexity of counting solutions to generalised satisfiability problems modulo k , 2008. arXiv:0809.1836.
- 26 John Faben and Mark Jerrum. The complexity of parity graph homomorphism: an initial investigation. *arXiv preprint arXiv:1309.4033*, 2013. arXiv:1309.4033.
- 27 Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.
- 28 Jacob Focke, Leslie Ann Goldberg, Marc Roth, and Stanislav Zivny. Counting homomorphisms to K_4 -minor-free graphs, modulo 2. *CoRR*, abs/2006.16632, 2020. arXiv:2006.16632.
- 29 Ralph Freese and Emil Kiss. Universal algebra calculator. Available at <http://uacalc.org/>.
- 30 Andreas Göbel, Leslie Ann Goldberg, and David Richerby. Counting homomorphisms to square-free graphs, modulo 2. *ACM Transactions on Computation Theory (TOCT)*, 8(3):1–29, 2016. doi:10.1145/2898441.
- 31 Andreas Göbel, J. A. Gregor Lagodzinski, and Karen Seidel. Counting homomorphisms to trees modulo a prime. In *MFCS*, volume 117, pages 49:1–49:13, 2018. doi:10.4230/LIPIcs.MFCS.2018.49.
- 32 Leslie Ann Goldberg, Martin Grohe, Mark Jerrum, and Marc Thurley. A complexity dichotomy for partition functions with mixed signs. *SIAM J. Comput.*, 39(7):3336–3402, 2010. doi:10.1137/090757496.
- 33 Andreas Göbel, Leslie Ann Goldberg, and David Richerby. The complexity of counting homomorphisms to cactus graphs modulo 2. *ACM Trans on Comp Th*, 6(4):1–29, 2014. doi:10.1145/2635825.
- 34 J. Hagemann and A. Mitschke. On n -permutable congruences. *Algebra Universalis*, 3:8–12, 1973.
- 35 Richard Hammack, Wilfried Imrich, and Sandi Klavzar. *Handbook of Product Graphs, Second Edition*. CRC Press, Inc., USA, 2nd edition, 2011.
- 36 Pawel M. Idziak, Petar Markovic, Ralph McKenzie, Matthew Valeriote, and Ross Willard. Tractability and learnability arising from algebras with few subpowers. *SIAM J. Comput.*, 39(7):3023–3037, 2010. doi:10.1137/090775646.
- 37 Peter Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1-2):185–204, 1998. doi:10.1016/S0304-3975(97)00230-2.
- 38 P.G. Jeavons, D.A. Cohen, and M. Gyssens. How to determine the expressive power of constraints. *Constraints*, 4:113–131, 1999. doi:10.1023/A:1009890709297.
- 39 Amirhossein Kazeminia. *Complexity of Modular Counting*. PhD thesis, Simon Fraser University, 2025.
- 40 Amirhossein Kazeminia and Andrei A Bulatov. Counting homomorphisms modulo a prime number. In *MFCS*, volume 138 of *LIPIcs*, pages 59:1–59:13, 2019. doi:10.4230/LIPIcs.MFCS.2019.59.
- 41 Amirhossein Kazeminia and Andrei A. Bulatov. Modular counting CSP: reductions and algorithms. In *STACS 2025*, volume 327 of *LIPIcs*, pages 60:1–60:18, 2025. doi:10.4230/LIPIcs.STACS.2025.60.
- 42 Amirhossein Kazeminia and Andrei A. Bulatov. Modular counting CSP: reductions and algorithms. *CoRR*, abs/2501.04224, 2025. doi:10.48550/arXiv.2501.04224.

22:20 Modular Counting over 3-Element and Conservative Domains

- 43 J. A. Gregor Lagodzinski, Andreas Göbel, Katrin Casel, and Tobias Friedrich. On counting (quantum-) graph homomorphisms in finite fields of prime order. *CoRR*, abs/2011.04827, 2021. [arXiv:2011.04827](https://arxiv.org/abs/2011.04827).
- 44 Emil L. Post. *The two-valued iterative systems of mathematical logic*, volume 5 of *Annals Mathematical Studies*. Princeton University Press, 1941.
- 45 Leslie Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8:189–201, 1979. doi:10.1016/0304-3975(79)90044-6.
- 46 Leslie Valiant. The complexity of enumeration and reliability problems. *SIAM J. on Comput.*, 8(3):410–421, 1979. doi:10.1137/0208032.
- 47 Dominic J.A. Welsh. *The computational complexity of some classical problems from statistical physics*. Oxford Univ. Press, 1990.
- 48 Dmitriy Zhuk. A proof of the CSP dichotomy conjecture. *J. ACM*, 67(5):30:1–30:78, 2020. doi:10.1145/3402029.