The #CSP Dichotomy is Decidable

Martin Dyer[1] and David Richerby*

1 **School of Computing, University of Leeds, LS2 9JT, UK**

Abstract

Bulatov (2008) and Dyer and Richerby (2010) have established the following dichotomy for the counting constraint satisfaction problem (#CSP): for any constraint language Γ, the problem of computing the number of satisfying assignments to constraints drawn from Γ is either in FP or is #P-complete, depending on the structure of Γ. The principal question left open by this research was whether the criterion of the dichotomy is decidable. We show that it is; in fact, it is in NP.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Constraint satisfaction problem, counting problems, complexity dichotomy, decidability.

Digital Object Identifier 10.4230/LIPIcs.STACS.2011.261

1 Introduction

Many important and natural problems in areas such as graph theory, Boolean logic, databases, type inference, scheduling, artificial intelligence and even theoretical physics can be expressed naturally as constraint satisfaction problems (CSPs) [9, 13]. In such problems, we seek to assign values from some domain to variables, while simultaneously satisfying a collection of constraints on the values that may be taken by given combinations of the variables.

For example, graph three-colourability is the problem of deciding whether we can assign one of three colours (domain values) to each vertex of a graph (variables) such that no edge joins vertices with the same colour (constraints). Since it includes this well-known NP-complete problem, it is immediate that this general form of CSP, known as *uniform* CSP is, itself, NP-complete.

For this reason, attention has focused on the so-called *nonuniform* version of CSP. Here, we fix a domain D and a set Γ of relations over D, known as the *constraint language*. We write CSP(Γ) for the version of CSP where we only allow constraints of the form, "the values assigned to variables $v_1, \dots, v_r$ must form a tuple in the r-ary relation $H \in \Gamma$." Note that all the constraints needed to express three-colourability can be written by taking D to be any three-element set and letting Γ contain just the binary disequality relation on D.

It follows that, for some Γ, even the restricted problem CSP(Γ) is NP-complete. However, taking Γ to be the binary disequality relation on a two-element domain allows us to express graph two-colourability, which is in P. Feder and Vardi [13] conjectured that these are the only possibilities: that is, for all Γ, CSP(Γ) is in P or is NP-complete. To date, this conjecture remains open but it is known to hold in special cases [1, 14, 18]. Recent efforts to resolve the conjecture have focused on techniques from universal algebra [6].

It follows from Ladner's theorem that there can be no such dichotomy for the whole of NP, since either P = NP or there is an infinite, strict hierarchy of complexity classes between

* Supported by EPSRC grant EP/E062172/1 "The Complexity of Counting in Constraint Satisfaction Problems".

28th Symposium on Theoretical Aspects of Computer Science (STACS'11).
Editors: Thomas Schwentick, Christoph Dürr; pp. 261–272
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
SYMPOSIUM ON THEORETICAL ASPECTS OF COMPUTER SCIENCE

the two [15]. Therefore, if $\mathsf{P} \neq \mathsf{NP}$, there are problems in NP that are neither in P nor NP-complete. However, a dichotomy for CSP is still possible because the problems expressible as $\mathsf{CSP}(\Gamma)$ for some Γ are a proper subset of NP. In particular, there is no Γ such that $\mathsf{CSP}(\Gamma)$ defines graph Hamiltonicity or even graph connectivity (this follows from [11, 12]). Further, Ladner's proof is via a diagonalisation that does not seem to be expressible in CSP [13].

In the present paper, we consider the *counting constraint satisfaction problem*, #CSP. Here, we are interested in the number of satisfying assignments to CSP instances. For several restricted classes of constraint language Γ, it was known that $\#\mathsf{CSP}(\Gamma)$ is either in polynomial time or #P-complete [4, 5, 7–9].

Bulatov successfully proved the dichotomy for all Γ [2,3], showing that $\#\mathsf{CSP}(\Gamma)$ is always either computable in polynomial time or #P-complete. He made extensive use of techniques from universal algebra; the present authors gave an elementary proof of an equivalent dichotomy [10]. The principal question left open by this research was the decidability of the distinct but equivalent criteria: that is, whether there is an algorithm that determines for which Γ $\#\mathsf{CSP}(\Gamma)$ is tractable and for which it is #P-complete. In this paper, we demonstrate such an algorithm.

We first describe the dichotomy — formal definitions will be given later. A ternary relation R is *balanced* if the matrix $M(x, y) = |\{z : xyz \in R\}|$ decomposes into blocks of rank one. A relation $R \subseteq D^r$ of arity $r \geq 3$ can be considered as a ternary relation over $D^k \times D^\ell \times D^{r-k-\ell}$ for any $k, \ell \geq 1$ with $k + \ell < r$. We say that R is balanced if every such interpretation as a ternary relation is balanced.

A relation that can be defined from the relations in Γ using only existential quantification, conjunction and equalities between variables is said to be *pp-definable*. Γ is *strongly balanced* if all pp-definable relations of arity three or more are balanced. This gives the criterion of the dichotomy in [10].

▶ **Theorem 1** (Dichotomy Theorem). *If Γ is strongly balanced, then $\#\mathsf{CSP}(\Gamma)$ is in* FP*; otherwise, it is* #P*-complete.*

Note that infinitely many relations are pp-definable in Γ, which is why decidability is not obvious. Bulatov's criterion is equivalent but expressed in terms of an infinite algebra constructed from Γ so, again, is not obviously decidable.

In the remainder of the paper, we construct a nondeterministic, polynomial-time algorithm that determines whether a given constraint langauge Γ is strongly balanced.

1.1 Proof outline

Our proof of the Dichotomy Theorem [10] uses succinct representations, which we call "frames", of a class of relations we call *strongly rectangular*. We do not require frames in the present paper but strong rectangularity is useful as it imposes structure and because every strongly balanced relation is strongly rectangular. We first show that strong rectangularity is decidable in NP.

We next develop an alternative, equational characterisation of strong balance. We use this characterisation to translate the question of whether a constraint langauge Γ over domain D is strongly balanced to a property of homomorphisms to the relational structure $(D, \Gamma)^6$ (we use a standard definition of Cartesian products). Using a technique due to Lovász [16], we show that this property is equivalent to the existence of certain automorphisms of the product structure. It follows that strong balance is decidable in NP, since we can nondeterministically "guess" a suitable collection of functions and check, in deterministic polynomial time, that they are the desired automorphisms.

1.2 Organisation of the paper

The remainder of the paper is organised as follows. The necessary definitions and notation and some basic results appear in Section 2. In Section 3, we review the concept of strong rectangularity, which we introduced in [10] and, in Section 4, we formally define strong balance and present some necessary results on rank-one block matrices. The proof of the decidability of strong balance appears in Section 5 and some concluding remarks follow, in Section 6.

2 Definitions and notation

Given a set D, we write $\mathbf{a} = (a_1, \dots, a_r)$ for an r-ary tuple in D^r. We will sometimes omit the brackets and commas and just write $a_1 \dots a_r$.

For a natural number n, we write $[n]$ for the set $\{1, \dots, n\}$.

2.1 Relations and constraints

Let $D = \{d_1, d_2, \dots, d_q\}$ be a finite *domain* with $q = |D|$. A *constraint language* Γ is a finite set of named, finitary relations on D, including the binary equality relation $\{(d_i, d_i) : i \in [q]\}$, which we denote by $=$. We will call $\mathfrak{G} = (D, \Gamma)$ a *relational structure*. We may view an r-ary relation H on D with $\ell = |H|$ as an $\ell \times r$ matrix with elements in D. Then a tuple $\mathbf{t} \in H$ is any row of this matrix. We write H^Γ for the instantiation of the relation H in Γ.

We define the *size* of a relation H as $\|H\| = \ell r$, the number of elements in its matrix, and the size of Γ as $\|\Gamma\| = \sum_{H \in \Gamma} \|H\|$. To avoid trivialities, we will assume that every relation $H \in \Gamma$ is nonempty. We will also assume that every $d \in D$ appears in a tuple of some relation $H \in \Gamma$. If this is not so for some d, we can remove it from D. It then follows that $\|\Gamma\| \geq q$.

Let $V = \{\nu_1, \nu_2, \dots, \nu_n\}$ be a finite *codomain*. An *assignment* is a function $\mathbf{x} \colon V \to D$. We will abbreviate $\mathbf{x}(\nu_i)$ to x_i. If $\{i_1, i_2, \dots, i_r\} \subseteq [n]$, we write $H(x_{i_1}, x_{i_2}, \dots, x_{i_r})$ for the relation $\Theta = \{\mathbf{x} : (x_{i_1}, x_{i_2}, \dots, x_{i_r}) \in H\}$ and we refer to this as a *constraint*. Then $(\nu_{i_1}, \nu_{i_2}, \dots, \nu_{i_r})$ is the *scope* of the constraint and we say that $\mathbf{x}$ is a *satisfying* assignment for the constraint if $\mathbf{x} \in \Theta$.

A Γ-*formula* Φ in a set of variables $\{x_1, x_2, \dots, x_n\}$ is a conjunction of constraints $\Theta_1 \wedge \dots \wedge \Theta_m$. We will identify the variables with the x_i above, although strictly the latter are only a *model* of the formula. The precise labelling of the variables is of no significance and a formula remains the same if its variables are bijectively renamed.

A Γ-formula Φ describes an instance of the *constraint satisfaction problem* (CSP) with constraint language Γ. A satisfying assignment for Φ is an assignment that satisfies all Θ_i ($i \in [m]$). The set of all satisfying assignments for Φ is the Γ-*definable* relation R_Φ over D. We will make no distinction between Φ and R_Φ, unless this could cause confusion.

If $H \subseteq D^r$ and $I = \{i_1, \dots, i_k\} \subseteq [r]$, with $i_1 < \dots < i_k$, we write $\mathsf{pr}_I H$ for the *projection* of H given by $\{(a_{i_1}, \dots, a_{i_k}) : a_1 \dots a_r \in H\}$.

2.2 Definability

A *primitive positive* (pp) formula Ψ is a Γ-formula Φ with existential quantification over some subset of the variables. A satisfying assignment for Ψ is any satisfying assignment for Φ. The unquantified (free) variables then determine the *pp-definable* relation R_Ψ, a projection of R_Φ. Again, we make no distinction between Ψ and R_Ψ.

The set of all Γ-definable relations is denoted by $\mathsf{CSP}(\Gamma)$ and the set of all relations pp-definable in Γ is the *relational clone* $\langle\Gamma\rangle$.

2.3 Polymorphisms

A *Mal'tsev polymorphism* of a relation $H \subseteq D^r$ is a function $\varphi\colon D^3 \to D$ with the following properties:

1. whenever $\mathbf{a}, \mathbf{b}, \mathbf{c} \in H$, we have $\varphi(\mathbf{a}, \mathbf{b}, \mathbf{c}) := \big(\varphi(a_1, b_1, c_1), \dots, \varphi(a_r, b_r, c_r)\big) \in H$;
2. for any $a, b \in D$, $\varphi(a, b, b) = \varphi(b, b, a) = a$.

The first condition describes a (ternary) polymorphism; the second is known as the Mal'tsev property. Note that the first condition can be extended to functions of arbitrary arity but we only require ternary polymorphisms here.

A function φ is a polymorphism of a constraint langauge Γ if it is a polymorphism of every relation in Γ. The following lemma is well known from the folklore and is easy to prove.

▶ **Lemma 2.** *φ is a polymorphism of Γ if, and only if, it is a polymorphism of $\langle\Gamma\rangle$.*

2.4 Homomorphisms and monomorphisms

A different, but equivalent, view of $\mathsf{CSP}(\Gamma)$ is often taken in the literature. This is to regard Φ as a finite structure with domain V and relations determined by the scopes of the constraints. Thus, we have relations $\tilde{H}$, where $(i_1, i_2, \dots, i_r) \in \tilde{H}$ if, and only if, $H(x_{i_1}, x_{i_2}, \dots, x_{i_r})$ is a constraint. A satisfying assignment $\mathbf{x}$ corresponds to a *homomorphism* from Φ to Γ.

The following definitions and notation are used only in Section 5. Let $[D_1 \to D_2]$ denote the set of functions from D_1 to D_2. A homomorphism between two relational structures $\mathfrak{S}_1 = (D_1, \Gamma_1)$ and $\mathfrak{S}_2 = (D_2, \Gamma_2)$ is a function $\sigma \in [D_1 \to D_2]$ that preserves relations. Thus, for each r-ary relation H and each tuple $\mathbf{u} = (u_1, \dots, u_r) \in H^{\Gamma_1}$, we have $\sigma(\mathbf{u}) = (\sigma(u_1), \dots, \sigma(u_r)) \in H^{\Gamma_2}$. We write $\sigma\colon \mathfrak{S}_1 \to \mathfrak{S}_2$ to indicate that σ is a homomorphism.

Let $[V \hookrightarrow D]$ and $[V \leftrightarrow D]$ denote the sets of all *injective* and *bijective* functions $V \to D$, respectively. An injective homomorphism is called a *monomorphism* and we will write $\sigma\colon \mathfrak{S}_1 \hookrightarrow \mathfrak{S}_2$. An *endomorphism* of a relational structure $\mathfrak{S}$ is a homomorphism $\sigma\colon \mathfrak{S} \to \mathfrak{S}$ (such a function is also a unary polymorphism). An *automorphism* is a bijective endomorphism whose inverse is also an endomorphism. Note that $[\mathfrak{S} \hookrightarrow \mathfrak{S}] = [\mathfrak{S} \leftrightarrow \mathfrak{S}]$, since D is finite, so an injective endomorphism is always an automorphism. Clearly, the identity function is always an automorphism, for any relational structure $\mathfrak{S}$.

2.5 Powers of structures

We use the following construction of *powers of* $\mathfrak{S}$ (see, for example, [17, p. 282]). For any relational structure $\mathfrak{S} = (D, \Gamma)$ and $k \in \mathbb{N}$, the relational structure $\mathfrak{S}^k = (D^k, \Gamma^k)$ is defined as follows. The domain is the Cartesian power D^k. The constraint language Γ^k is such that, each r-ary relation $H \in \Gamma$, corresponds to an r-ary $H^k \in \Gamma^k$, which is defined as follows. If $\mathbf{u}_i = (u_{i,1}, u_{i,2}, \dots, u_{i,k}) \in D^k$ $(i \in [r])$, then $(\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_r) \in H^k$ if, and only if, $(u_{1,j}, u_{2,j}, \dots, u_{r,j}) \in H$ for all $j \in [k]$. Now, if Ψ is a formula pp-definable in Γ, we define the corresponding formula Ψ^k to be identical to Ψ, except that each occurrence of $H \in \Gamma$ is replaced by the corresponding relation $H^k \in \Gamma^k$. Observe that Ψ^k is actually pp-definable in Γ, since $\Psi^k(\mathbf{x}) = \Psi(\mathbf{x}_1) \wedge \Psi(\mathbf{x}_2) \wedge \cdots \wedge \Psi(\mathbf{x}_k)$, where the $\mathbf{x}_i$ $(i \in [k])$ are disjoint n-tuples of variables. In particular, we have $|\Psi^k| = |\Psi|^k$.

Using this construction, the definition of a polymorphism can be reformulated. In this view of $\mathsf{CSP}(\Gamma)$, it follows directly that a k-ary polymorphism is just a homomorphism $\varphi\colon \mathfrak{S}^k \to \mathfrak{S}$.

3 Rectangularity

The key tool in our proof of the Dichotomy Theorem [10] is the use of succint representations (which we call “frames”) for a class of relations that we call strongly rectangular. Frames allow us to efficiently count solutions in the polynomial-time cases but we do not require them here. However, the concept of strong rectangularity does play a role in our analysis.

A binary relation $H \subseteq A_1 \times A_2$ is *rectangular* if, for all $a, b \in A_1$ and $c, d \in A_2$,

$$ac, ad, bc \in H \text{ implies } bd \in H\,.$$

For $r \geq 2$, a relation $H \subseteq D^r$ can be considered as a binary relation in $D^k \times D^{r-k}$ for any k with $1 \leq k < r$. We say that a relation of arity $r \geq 2$ is rectangular if every such expression of it as a binary relation is rectangular.

▶ **Definition 3.** A constraint langauge Γ is strongly rectangular if every relation in $\langle\Gamma\rangle$ of arity ≥ 2 is rectangular.

We consider the following computational problem.

STRONG RECTANGULARITY

Instance : A relational structure $\mathfrak{G} = (D, \Gamma)$.

Question : Is Γ strongly rectangular?

As $\langle\Gamma\rangle$ is an infinite set, it is not immediate whether STRONG RECTANGULARITY is decidable. However, it turns out that strong rectangularity is equivalent to the existence of a Mal’tsev polymorphism.

▶ **Lemma 4.** *Γ is strongly rectangular if, and only if, it has a Mal’tsev polymorphism.*

We defer the proof of this lemma for a moment. We require the lemma to prove the following result.

▶ **Lemma 5.** STRONG RECTANGULARITY *is in* NP.

Proof. By Lemma 4, Γ is strongly rectangular if, and only if, it has a Mal’tsev polymorphism. Thus, we nondeterministically guess a function $\varphi\colon D^3 \to D$ in time $O(q^3)$. We can verify that φ is a Mal’tsev polymorphism, deterministically in time $O(\|\Gamma\|^4)$ just by checking that all relevant inputs to φ produce appropriate outputs. ◀

Lemma 4 is usually proved in an algebraic setting. That proof is not difficult, but requires an understanding of concepts from universal algebra, such *free algebras* and *varieties* [6]. Therefore, we will give a proof in the relational setting which, we believe, provides more insight for the reader whose primary interest is in relations.

Proof of Lemma 4. Suppose Γ has a Mal’tsev polymorphism φ. Consider any pp-definable binary relation $B \subseteq D^r \times D^s$. By Lemma 2, φ is also a polymorphism of B. If $(\mathbf{a}, \mathbf{c})$, $(\mathbf{a}, \mathbf{d})$, $(\mathbf{b}, \mathbf{d}) \in B$ then we have $(\varphi(\mathbf{a}, \mathbf{a}, \mathbf{b}), \varphi(\mathbf{c}, \mathbf{d}, \mathbf{d})) = (\mathbf{b}, \mathbf{c}) \in B$, from the definition of a Mal’tsev polymorphism. Thus, B is rectangular and, hence, Γ is strongly rectangular.

Conversely, suppose Γ is strongly rectangular. Denote the relation $H \in \Gamma$ by $H = \{\mathbf{u}_i^H : i \in [\ell_H]\}$, where $\mathbf{u}_i^H \in D^{r_H}$. Consider the Γ-formula

$$\Phi(\mathbf{x}) \;=\; \bigwedge_{H\in\Gamma} \bigwedge_{i_1\in[\ell_H]} \bigwedge_{i_2\in[\ell_H]} \bigwedge_{i_3\in[\ell_H]} H\big(\mathbf{x}^H_{i_1,i_2,i_3}\big)\,,$$

where $\mathbf{x}^H_{i_1,i_2,i_3}$ is an r_H-tuple of variables, distinct for all $H \in \Gamma$, $i_1, i_2, i_3 \in [\ell_H]$. Thus, the relation R_Φ has arity $r_\Phi = \sum_{H\in\Gamma} r_H \ell_H^3$ and $|R_\Phi| = \prod_{H\in\Gamma} {\ell_H}^{\ell_H^3}$.

Clearly R_Φ has three tuples $\mathbf{u}_1$, $\mathbf{u}_2$, $\mathbf{u}_3$ such that the sub-tuple corresponding to $\mathbf{x}^H_{i_1,i_2,i_3}$ in $\mathbf{u}_j$ is $\mathbf{u}^H_{i_j}$ for each $j \in \{1, 2, 3\}$. Then $U = \{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$ has the following universality property for Γ. For all $H \in \Gamma$ and every triple of (not necessarily distinct) tuples $\mathbf{t}_1$, $\mathbf{t}_2$, $\mathbf{t}_3 \in H$, there is a set $I(\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3)$ with $I \subseteq [r_\Phi]$, $|I| = r_H$ such that $\mathsf{pr}_I R_\Phi = H$ and $\mathsf{pr}_I \mathbf{u}_j = \mathbf{t}_j$ $(j = 1, 2, 3)$.

Now, for each set of identical columns in U, we impose equality on the corresponding variables in Φ, to give a Γ-formula Φ'. Let U' be the resulting submatrix of U, with rows $\mathbf{u}'_1$, $\mathbf{u}'_2$, $\mathbf{u}'_3$. Observe that U' is obtained by deleting copies of columns in U. Therefore U' has no identical columns and has a column (a, b, c) for all $a, b, c \in \mathsf{pr}_k H$ with $H \in \Gamma$ and $k \in [r_H]$.

Next, for all columns (a, b, c) of U' such that $b \notin \{a, c\}$, we impose existential quantification on the corresponding variables in Φ', to give a pp-formula Φ''. Let U'' be the submatrix of U' with rows $\mathbf{u}''_1$, $\mathbf{u}''_2$, $\mathbf{u}''_3$ corresponding to $\mathbf{u}'_1$, $\mathbf{u}'_2$, $\mathbf{u}'_3$. Then U'' results from deleting columns in U' and U'' has columns of the form (a, a, b) or (c, d, d). Thus, after rearranging columns (relabelling variables), we will have

$$U'' = \begin{bmatrix} \mathbf{u}''_1 \\ \mathbf{u}''_2 \\ \mathbf{u}''_3 \end{bmatrix} = \begin{bmatrix} \mathbf{a} & \mathbf{c} \\ \mathbf{a} & \mathbf{d} \\ \mathbf{b} & \mathbf{d} \end{bmatrix},$$

for some nonempty tuples $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{d}$. By strong rectangularity, this implies $\mathbf{u}'' = \begin{bmatrix} \mathbf{b} & \mathbf{c} \end{bmatrix} \in R_{\Phi''}$.

Removing the existential quantification in Φ'', $\mathbf{u}''$ can be extended to $\mathbf{u}' \in R_{\Phi'}$. Now, if column k of U' is (a, b, c) say, we define $\varphi(a, b, c) = u'_k$. This is unambiguous, since U' has no identical columns. Thus, $\mathbf{u}' = \varphi(\mathbf{u}'_1, \mathbf{u}'_2, \mathbf{u}'_3) \in R_{\Phi'}$. If, for any $a, b, c \in D$, $\varphi(a, b, c)$ remains undefined, we will set $\varphi(a, b, c) = a$ unless $a = b$, in which case $\varphi(a, b, c) = c$. Clearly φ satisfies $\varphi(a, b, b) = \varphi(b, b, a) = a$, for all $a, b \in D$, and so has the Mal'tsev property.

Removing the equalities between variables in Φ', $\mathbf{u}'$ can be further extended to $\mathbf{u} = \varphi(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) \in R_\Phi$. This is consistent since $\mathbf{u}$ satisfies the equalities imposed on Φ to give Φ'. Now, for any $\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3 \in H$, the universality property of U implies that $\mathsf{pr}_I \mathbf{u} = \varphi(\mathbf{t}_1, \mathbf{t}_2, \mathbf{t}_3) \in H$. Thus, φ preserves all $H \in \Gamma$, so it is a polymorphism and hence a Mal'tsev polymorphism. ◀

4 Strong balance

Recall the Dichotomy Theorem (Theorem 1): $\#\mathsf{CSP}(\Gamma)$ is computable in polynomial time if Γ is strongly balanced, and is $\#\mathsf{P}$-complete, otherwise. In this section, we formally define strong balance and investigate its properties.

A *rank-one block matrix* is a $k \times k$ matrix M whose rows and columns can be permuted to give a block-diagonal matrix whose non-zero blocks have rank 1. (It is equivalent to say that the relation $\{xy : M(x, y) \neq 0\}$ is rectangular and there are functions $\alpha, \beta\colon [k] \to \mathbb{N}$ such that $M(x, y) = \alpha(x)\beta(y)$ where $M \neq 0$ but we will use a third characterisation, given by Corollary 9.)

Let $H \subseteq A_1 \times A_2 \times A_3$ be a ternary relation. We say that H is *balanced* if the *balance matrix*

$$M(x, y) = |\{z \in A_3 : (x, y, z) \in H\}| \qquad (x \in A_1,\ y \in A_2)$$

is a rank-one block matrix. For $r > 3$, a relation $H \subseteq D^r$ can be expressed as a ternary relation in $D^k \times D^\ell \times D^{r-k-\ell}$ for any $k, \ell \geq 1$ with $k + \ell < r$. We say that a relation H of arity $r > 3$ is balanced if every such expression of H as a ternary relation is balanced.

▶ **Definition 6.** A constraint language Γ is *strongly balanced* if every relation of arity ≥ 3 in $\langle\Gamma\rangle$ is balanced.

Note that infinitely many relations are pp-definable in any constraint language. Our goal in the remainder of this paper is to show that, this notwithstanding, the property of being strongly balanced is decidable.

Towards this goal, we derive a different characterisation of rank-one block matrices. This may seem more complicated than the original definition, but it is more suited to our purpose. By the *underlying relation* of a matrix $M(x, y)$, we mean the relation $\{(x, y) : M(x, y) \neq 0\}$. We say that a matrix is rectangular if its underlying relation is.

▶ **Lemma 7.** *A is a rank-one block matrix if, and only if, every 2×2 submatrix of A is a rank-one block matrix.*

Proof. Let A be a $k \times \ell$ rank-one block matrix and let

$$B = \begin{bmatrix} a_{ir} & a_{is} \\ a_{jr} & a_{js} \end{bmatrix} \qquad (i, j \in [k],\ r, s \in [\ell])$$

be any 2×2 submatrix of A. If any of a_{ir}, a_{is}, a_{jr}, a_{js} is zero, at least two must be zero, since A is rectangular. In this case, B is clearly a rank-one block matrix. If a_{ir}, a_{is}, a_{jr}, a_{js} are all nonzero, B must be a submatrix of some block of A. Since this block has rank one, B also has rank one.

Conversely, suppose A is not a rank-one block matrix. If its underlying relation is not rectangular, there exist $a_{ir}, a_{is}, a_{jr} > 0$ with $a_{js} = 0$. The corresponding matrix B clearly has rank two, and has only one block so is not a rank-one block matrix. If the underlying relation of A is rectangular, then A must have a block of rank at least two. This block must have some 2×2 submatrix B with rank two and all its elements non-zero. ◀

▶ **Lemma 8.** *A rectangular 2×2 matrix A is a rank-one block matrix if, and only if, $a_{11}^2 a_{22}^2 a_{12} a_{21} = a_{12}^2 a_{21}^2 a_{11} a_{22}$.*

Proof. The equation holds if any of a_{11}, a_{12}, a_{21} or a_{22} is zero. But, then, rectangularity implies that at least two of them must be zero and A is a rank-one block matrix in all possible cases. Otherwise, the equation is equivalent to $a_{11}a_{22} = a_{12}a_{21}$, which is the condition that A is singular. So A is one block, with rank one. The argument is clearly reversible. ◀

▶ **Corollary 9.** *A rectangular $k \times \ell$ matrix A is a rank-one block matrix if, and only if, $a_{ir}^2 a_{js}^2 a_{is} a_{jr} = a_{is}^2 a_{jr}^2 a_{ir} a_{js}$, for all $i, j \in [k]$ and $r, s \in [\ell]$.*

Proof. When $i = j$ or $r = s$, the two sides of the equation are identical. Otherwise, the equality follows directly from Lemmas 7 and 8. ◀

It is possible to modify the above so that Corollary 9 involves products of only five elements, rather than six, but we do not pursue that refinement here.

The following lemma gives a basic precondition for strong balance.

▶ **Lemma 10.** *Every strongly balanced constraint language is strongly rectangular.*

Proof. Suppose $\Psi \in \langle\Gamma\rangle$ is not rectangular. There are $\mathbf{ac}, \mathbf{bc}, \mathbf{ad} \in R$ such that $\mathbf{bd} \notin R$. Let Ψ' be the relation $\{\mathbf{uvv} : \mathbf{uv} \in \Psi\}$ and let M' be its balance matrix. The 2×2 submatrix corresponding to rows $\mathbf{a}$ and $\mathbf{b}$ and columns $\mathbf{c}$ and $\mathbf{d}$ is

$$B = \begin{bmatrix} \alpha & \beta \\ \gamma & 0 \end{bmatrix}$$

for some $\alpha, \beta, \gamma \geq 1$. This submatrix is not a rank-one block matrix so, by Lemma 7, nor is M'. Therefore, Γ is not strongly balanced. ◀

5 Decidability

We now give a relaxation of the strong balance criterion, by noting the conditions sufficient for the success of the algorithm in [10]. For an instance with n variables, the algorithm only requires that ternary relations on $D \times D \times D^i$, for $i \in [n-2]$, be balanced. Therefore, let $\Psi(\mathbf{x})$, with $\mathbf{x} = (x_1, \ldots, x_n)$, be an arbitrary formula pp-definable in Γ. For the algorithm to succeed, it suffices that the $q \times q$ matrix

$$M(a,b) = \big|\{\mathbf{x} \in [V \to D] : \mathbf{x} \in \Psi,\, x_1 = a,\, x_2 = b\}\big| \qquad (\forall a, b \in D)$$

is a rank-one block matrix for any Ψ. We may therefore take this as the criterion for strong balance.

We will construct an algorithm to solve the following decision problem.

STRONG BALANCE
- Instance : A relational structure $\mathfrak{G} = (D, \Gamma)$.
- Question : Is Γ strongly balanced?

Recall from Section 2 that we may assume that $\|\Gamma\| \geq q$. Thus, we may take $\|\Gamma\|$ as the measure of input size for STRONG BALANCE and we bound the complexity of STRONG BALANCE as a function of this value. Complexity is a secondary issue, since $\|\Gamma\|$ is a constant in the nonuniform model for $\#\mathsf{CSP}(\Gamma)$. In this model, we are only required to show that some algorithm exists to solve STRONG BALANCE. However, we believe that the computational complexity of deciding the dichotomy is intrinsically interesting. Our approach will be to show that the strong balance condition is equivalent to a structural property of Γ that can be checked in NP.

We may assume that Γ is strongly rectangular since, if it is not, we know by Lemma 10 that it is not strongly balanced. For the remainder of this section, we fix an n-ary pp-definable relation $\Psi \in \langle\Gamma\rangle$ with balance matrix M.

By Corollary 9, the condition for M to be a rank-one block matrix is that

$$M(a,c)^2 M(a,d) M(b,d)^2 M(b,c) = M(a,d)^2 M(a,c) M(b,c)^2 M(b,d) \quad \text{for all } a,b,c,d \in D.$$

We can reformulate this condition using the construction of powers of $\mathfrak{G}$. If $\mathbf{a} = (a_1, \ldots, a_k)$ and $\mathbf{b} = (b_1, \ldots, b_k)$, the balance matrix M_k for Ψ^k is the $q^k \times q^k$ matrix given by

$$\begin{aligned} M_k(\mathbf{a}, \mathbf{b}) &= \big|\{\mathbf{x} \in [V \to D^k] \cap \Psi^k : x_1 = \mathbf{a},\, x_2 = \mathbf{b}\}\big| \\ &= M(a_1, b_1) M(a_2, b_2) \cdots M(a_k, b_k)\,. \end{aligned}$$

The condition for M to be a rank-one block matrix can be rewritten as

$$M_6(\bar{a}, \bar{c}) = M_6(\bar{a}, \bar{d})\,, \tag{1}$$

where

$$\bar{a} = (a,a,a,b,b,b),\ \bar{c} = (c,c,d,d,d,c),\ \bar{d} = (d,d,c,c,c,d)\,. \tag{2}$$

Let us fix $\bar{a}$, $\bar{c}$, $\bar{d}$. For notational simplicity, let us write $\bar{\mathfrak{S}}$ for $\mathfrak{S}^6$, $\bar{\Gamma}$ for Γ^6, $\bar{\Psi}$ for Ψ^6, $\bar{M}$ for M_6 and $\bar{D}$ for D^6. Then, from (1), we must verify that $\bar{M}(\bar{a},\bar{c}) = \bar{M}(\bar{a},\bar{d})$ for all relations $R_{\bar{\Psi}}$ that are pp-definable in $\bar{\Gamma}$ and given $\bar{a},\bar{c},\bar{d} \in \bar{D}$. We use a method of Lovász [16]; see also [8]. For $\bar{s} \in \bar{D}$, and a pp-definition $\bar{\Psi}$ in variables V, let

$$\begin{aligned}\mathrm{Hom}_{\bar{s}}(\bar{\Psi}) &= \{\mathbf{x} \in [V \to \bar{D}] \cap \bar{\Psi} : x_1 = \bar{a},\ x_2 = \bar{s}\}\\ \mathrm{hom}_{\bar{s}}(\bar{\Psi}) &= |\mathrm{Hom}_{\bar{s}}(\bar{\Psi})|\,.\end{aligned}$$

However, a homomorphism $V \to \bar{D}$ that is consistent with $\bar{\Psi}$ is just a satisfying assignment to $\bar{\Psi}$. $\bar{M}(\bar{a},\bar{s})$ is the number of such assignments with $x_1 = \bar{a}$ and $x_2 = \bar{s}$, i.e., the number of homomorphisms that map $x_1 \mapsto \bar{a}$ and $x_2 \mapsto \bar{s}$. This proves the following.

▶ **Lemma 11.** Γ *is strongly balanced if, and only if,* $\mathrm{hom}_{\bar{c}}(\bar{\Psi}) = \mathrm{hom}_{\bar{d}}(\bar{\Psi})$ *for all formulae* $\bar{\Psi}$ *and all* $\bar{a},\bar{c},\bar{d}$ *of the form above.*

We will also need to consider the injective functions in $\mathrm{Hom}_{\bar{s}}(\bar{\Psi})$. For $\bar{s} \in \bar{D}$, let

$$\begin{aligned}\mathrm{Mon}_{\bar{s}}(\bar{\Psi}) &= \{\mathbf{x} \in [V \hookrightarrow \bar{D}] \cap \bar{\Psi} : x_1 = \bar{a},\ x_2 = \bar{s}\}\\ \mathrm{mon}_{\bar{s}}(\bar{\Psi}) &= |\mathrm{Mon}_{\bar{s}}(\bar{\Psi})|\,.\end{aligned}$$

▶ **Lemma 12.** $\mathrm{hom}_{\bar{c}}(\bar{\Psi}) = \mathrm{hom}_{\bar{d}}(\bar{\Psi})$ *for all* $\bar{\Psi}$ *if, and only if,* $\mathrm{mon}_{\bar{c}}(\bar{\Psi}) = \mathrm{mon}_{\bar{d}}(\bar{\Psi})$ *for all* $\bar{\Psi}$.

Proof. Consider the set $\mathcal{I}$ of all partitions I of V into disjoint classes $\bar{I}_1,\dots,\bar{I}_{k_I}$, such that $1 \in \bar{I}_1$, $2 \in \bar{I}_2$. Writing $I \preceq I'$ whenever I is a refinement of I', $\mathbb{P} = (\mathcal{I},\preceq)$ is a poset. We will write $\perp$ for the partition into singletons, so $\perp \preceq I$ for all $I \in \mathcal{I}$.

Let V/I denote the set of classes $\bar{I}_1,\dots,\bar{I}_{k_I}$ of the partition I, so $|V/I| = k_I$, and let $\bar{I}_1$, $\bar{I}_2$ be denoted by $1/I$, $2/I$. Let $\bar{\Psi}/I$ denote the relation obtained from $\bar{\Psi}$ by imposing equality on all pairs of variables that occur in the same partition of I. Thus, the constraints $x_1 = \bar{a}$, $x_2 = \bar{s}$ become $x_{1/I} = \bar{a}$, $x_{2/I} = \bar{s}$. Then we have

$$\mathrm{hom}_{\bar{s}}(\bar{\Psi}) \;=\; \mathrm{hom}_{\bar{s}}(\bar{\Psi}/\perp) \;=\; \sum_{I\in\mathcal{I}} \mathrm{mon}_{\bar{s}}(\bar{\Psi}/I) \;=\; \sum_{I\in\mathcal{I}} \mathrm{mon}_{\bar{s}}(\bar{\Psi}/I)\zeta(\perp,I)\,, \tag{3}$$

where $\zeta(I,I') = 1$, if $I \preceq I'$, and $\zeta(I,I') = 0$, otherwise, is the ζ-function of $\mathbb{P}$. Thus, if $\mathrm{mon}_{\bar{c}}(\bar{\Psi}) = \mathrm{mon}_{\bar{d}}(\bar{\Psi})$ for all $\bar{\Psi}$, then

$$\mathrm{hom}_{\bar{c}}(\bar{\Psi}) \;=\; \sum_{I\in\mathcal{I}} \mathrm{mon}_{\bar{c}}(\bar{\Psi}/I)\zeta(\perp,I) \;=\; \sum_{I\in\mathcal{I}} \mathrm{mon}_{\bar{d}}(\bar{\Psi}/I)\zeta(\perp,I) \;=\; \mathrm{hom}_{\bar{d}}(\bar{\Psi})\,. \tag{4}$$

Conversely, suppose that $\mathrm{hom}_{\bar{c}}(\bar{\Psi}) = \mathrm{hom}_{\bar{d}}(\bar{\Psi})$ for all $\bar{\Psi}$. The reasoning used to give (3) implies, more generally, that

$$\mathrm{hom}_{\bar{s}}(\bar{\Psi}/I) \;=\; \sum_{I\preceq I'} \mathrm{mon}_{\bar{s}}(\bar{\Psi}/I') \;=\; \sum_{I'\in\mathcal{I}} \mathrm{mon}_{\bar{s}}(\bar{\Psi}/I')\zeta(I,I')\,.$$

Now, Möbius inversion for posets [20, Ch. 25] implies that the matrix $\zeta\colon \mathcal{I}\times\mathcal{I} \to \{0,1\}$ has an inverse $\mu\colon \mathcal{I}\times\mathcal{I} \to \mathbb{Z}$. It follows directly that

$$\mathrm{mon}_{\bar{s}}(\bar{\Psi}) \;=\; \sum_{I\in\mathcal{I}} \mathrm{hom}_{\bar{s}}(\bar{\Psi}/I)\mu(\perp,I)\,.$$

Thus, with $\hom_{\bar{c}}(\bar{\Psi}) = \hom_{\bar{d}}(\bar{\Psi})$ for all $\bar{\Psi}$, we have

$$\operatorname{mon}_{\bar{c}}(\bar{\Psi}) \;=\; \sum_{I \in \mathcal{I}} \hom_{\bar{c}}(\bar{\Psi}/I)\mu(\bot, I) \;=\; \sum_{I \in \mathcal{I}} \hom_{\bar{d}}(\bar{\Psi}/I)\mu(\bot, I) \;=\; \operatorname{mon}_{\bar{d}}(\bar{\Psi})\,. \tag{5}$$

Now, (4) and (5) give the conclusion. ◀

▶ **Lemma 13.** *$\operatorname{mon}_{\bar{c}}(\bar{\Psi}) = \operatorname{mon}_{\bar{d}}(\bar{\Psi})$ for all $\bar{\Psi}$, if, and only if, there is an automorphism $\eta \colon \bar{D} \leftrightarrow \bar{D}$ of $\bar{\mathfrak{S}} = (\bar{D}, \bar{\Gamma})$ such that $\eta(\bar{a}) = \bar{a}$ and $\eta(\bar{c}) = \bar{d}$.*

Proof. The condition holds if $\bar{\mathfrak{S}}$ has such an automorphism since, if $\bar{\Psi}(\mathbf{x}) = \exists \mathbf{y}\, \bar{\Phi}(\mathbf{x}, \mathbf{y})$ for some $\bar{\Phi}$, then

$$\begin{aligned}
\operatorname{mon}_{\bar{c}}(\bar{\Psi}) \;&=\; |\{\mathbf{x} \in [V \hookrightarrow \bar{D}] : x_1 = \bar{a},\, x_2 = \bar{c},\, \exists \mathbf{y}\, (\mathbf{x}, \mathbf{y}) \in \bar{\Phi}\}| \\
&=\; |\{\eta(\mathbf{x}) \in [V \hookrightarrow \bar{D}] : x_1 = \eta(\bar{a}),\, x_2 = \eta(\bar{c}),\, \exists \mathbf{y}\, (\eta(\mathbf{x}), \eta(\mathbf{y})) \in \bar{\Phi}\}| \\
&=\; |\{\mathbf{x} \in [V \hookrightarrow \bar{D}] : x_1 = \bar{a},\, x_2 = \bar{d},\, \exists \mathbf{y}\, (\mathbf{x}, \mathbf{y}) \in \bar{\Phi}\}| \\
&=\; \operatorname{mon}_{\bar{d}}(\bar{\Psi})\,.
\end{aligned}$$

Conversely, suppose we have $\operatorname{mon}_{\bar{c}}(\bar{\Psi}) = \operatorname{mon}_{\bar{d}}(\bar{\Psi})$ for all $\bar{\Psi}$. Consider the following $\bar{\Gamma}$-formula $\bar{\Phi}$ with domain $\bar{D}$ and variables $x_{\bar{t}}$ ($\bar{t} \in \bar{D}$):

$$\bar{\Phi}(\mathbf{x}) = \bigwedge_{\bar{H} \in \bar{\Gamma}} \; \bigwedge_{(\bar{u}_1, \ldots, \bar{u}_r) \in \bar{H}} \bar{H}(x_{\bar{u}_1}, \ldots, x_{\bar{u}_r})\,.$$

Then

$$\operatorname{Mon}_{\bar{s}}(\bar{\Phi}) = \{\mathbf{x} \in [\bar{D} \hookrightarrow \bar{D}] : x_{\bar{a}} = \bar{a},\, x_{\bar{c}} = \bar{s},\, \mathbf{x} \in \bar{\Phi}\}\,.$$

We have $\operatorname{Mon}_{\bar{c}}(\bar{\Phi}) \neq \emptyset$, since the identity assignment $x_{\bar{t}} = \bar{t}$ for all $\bar{t} \in \bar{D}$ is clearly satisfying. Thus, by the assumption, $\operatorname{Mon}_{\bar{d}}(\bar{\Phi}) \neq \emptyset$. Let $\eta \in \operatorname{Mon}_{\bar{d}}(\bar{\Phi})$, so η is an endomorphism of $\bar{\mathfrak{S}}$ with $\eta(\bar{a}) = \bar{a}$ and $\eta(\bar{c}) = \bar{d}$. Since $[D \hookrightarrow D] = [D \leftrightarrow D]$, $\eta \colon D \leftrightarrow D$ is the required automorphism. ◀

▶ **Corollary 14.** *$\mathfrak{S} = (D, \Gamma)$ is strongly balanced if, and only if, for all $\bar{a}, \bar{c}, \bar{d}$ as defined in* (2)*, $\bar{\mathfrak{S}} = (\bar{D}, \bar{\Gamma})$ has an automorphism ψ such that $\psi(\bar{a}) = \bar{a}$ and $\psi(\bar{c}) = \bar{d}$.*

Proof. This follows from Lemmas 11, 12 and 13. ◀

This characterisation of strong balance leads directly to a nondeterministic algorithm.

▶ **Theorem 15.** STRONG BALANCE *is in* NP.

Proof. We first determine whether Γ is strongly rectangular, using the method of Lemma 5. If it is not, then Γ is not strongly balanced, by Lemma 10.

Otherwise, we can construct $\bar{\mathfrak{S}} = (\bar{D}, \bar{\Gamma})$ in time $O(\|\Gamma\|^6)$. Let $\bar{q} = q^6 = |\bar{D}|$ and let Π denote the set of $\bar{q}\,!$ permutations of $\bar{D}$. Each $\pi \in \Pi$ is a function $\pi \colon \bar{D} \hookrightarrow \bar{D}$ and so a potential automorphism of $\bar{\mathfrak{S}}$. For each of the q^4 possible choices $a, b, c, d \in D$, we determine $\bar{a}, \bar{c}, \bar{d} \in \bar{D}$ in polynomial time. We select $\pi \in \Pi$ nondeterministically and check that $\pi(\bar{a}) = \bar{a}$, $\pi(\bar{c}) = \bar{d}$ and that π preserves all $\bar{H} \in \bar{\Gamma}$. The computation requires $O(q^4\|\bar{\Gamma}\|^2) = O(\|\Gamma\|^{16})$ time in total, so everything other than the $O(q^{10}) = O(\|\Gamma\|^{10})$ nondeterministic choices can be done deterministically in a polynomial number of steps. ◀

We have paid little attention to the efficiency of the computations in Theorem 15. If the elements of D are encoded as binary numbers in $[q]$, comparisons and nondeterministic choices require $O(\log q)$ bit operations, rather than the $O(1)$ operations in our accounting. On the other hand, membership in H^6 can be tested in $O(\|H\|)$ comparisons, rather than the $O(\|H\|^6)$ that we have allowed. This might be reduced further by storing H in a suitable data structure, instead of a simple matrix. As we have noted, Corollary 9 can be refined to products of five terms, which can be used to improve the algorithm of Theorem 15.

If we consider the domain D as fixed, the problem of deciding whether constraint languages over that domain D are strongly balanced is in deterministic polynomial time. With D fixed, there are a constant number of potential Mal'tsev polymorphisms that must be checked to determine strong rectangularity, and the numbers of tuples $\bar{a}$, $\bar{c}$, $\bar{d}$ and possible automorphisms on D are also fixed constants.

6 Conclusions

We have shown that there is an algorithm that determines whether a constraint language is strongly balanced. This means that the complexity dichotomy for $\#\mathsf{CSP}(\Gamma)$ is effective, thus answering the major open problem that arose from the proofs that the dichotomy exists [3, 10].

Although we have shown strong balance to be decidable in NP, we have only established an upper bound. We believe the complexity of the problem to be interesting in its own right. It is not hard to see that the problem of determining whether the automorphisms of Corollary 14 exist is reducible to the graph isomorphism problem. It therefore seems unlikely that strong balance is NP-complete as this would imply NP-completeness of graph isomorphism which would, in turn, imply the collapse of the polynomial hierarchy [19]. However, we leave open the question of whether strong balance is equivalent to graph isomorphism or whether more efficient algorithms exist.

Bulatov's proof of the $\#\mathsf{CSP}(\Gamma)$ dichotomy is expressed not in terms of strong balance but in terms of the "congruence singularity" of Γ (or, more precisely, of an algebra defined from Γ). We have shown the two conditions to be equivalent but it remains open if there is a direct proof that the property of congruence singularity is decidable.

References

1. A. A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element domain. *Journal of the ACM*, 53(1):66–120, 2006.
2. A. A. Bulatov. The complexity of the counting constraint satisfaction problem. *Electronic Colloquium on Computational Complexity*, 14(093), 2007. (Revised Feb. 2009).
3. A. A. Bulatov. The complexity of the counting constraint satisfaction problem. In *Proc. 35th International Colloquium on Automata, Languages and Programming (Part 1)*, LNCS 5125, pp. 646–661. Springer, 2008.
4. J.-Y. Cai, P. Lu, and M. Xia. Holant problems and counting CSP. In *Proc. 41st Annual ACM Symposium on Theory of Computing (STOC 2009)*, pp. 715–724. ACM, 2009.
5. N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125(1):1–12, 1996.
6. K. Denecke and S. L. Wismath. *Universal Algebra and Applications in Theoretical Computer Science*. Chapman and Hall/CRC, 2002.
7. M. E. Dyer, L. A. Goldberg, and M. R. Jerrum. A complexity dichotomy for hypergraph partition functions. *Computational Complexity*, 19(4):605–633, 2010.

8 M. E. Dyer, L. A. Goldberg, and M. S. Paterson. On counting homomorphisms to directed acyclic graphs. *Journal of the ACM*, 54(6), 2007.

9 M. E. Dyer and C. S. Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17(3–4):260–289, 2000. (Corrigendum in *Random Structures and Algorithms*, 25(3):346–352, 2004.)

10 M. E. Dyer and D. M. Richerby. On the complexity of #CSP. In *Proc. 42nd ACM Symposium on Theory of Computing (STOC 2010)*, pp. 725–734. ACM, 2010.

11 R. Fagin. Monadic generalized spectra. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 21:89–96, 1975.

12 R. Fagin, L. J. Stockmeyer, and M. Y. Vardi. On monadic NP vs monadic co-NP. *Information and Computation*, 120(1):78–92, 1995.

13 T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.

14 P. Hell and J. Nešetřil. On the complexity of H-coloring. *Journal of Combinatorial Theory (Series B)*, 48(1):92–110, 1990.

15 R. E. Ladner. On the structure of polynomial time reducibility. *Journal of the ACM*, 22(1):155–171, 1975.

16 L. Lovász. Operations with structures. *Acta. Math. Acad. Sci. Hung.*, 18:321–328, 1967.

17 J. Nešetřil, M. H. Siggers and L. Zádori. A combinatorial constraint satisfaction problem dichotomy classification conjecture. *European Journal of Combinatorics*, 31(1):280–296, 2010.

18 T. Schaefer. The complexity of satisfiability problems. In *Proc. 10th Annual ACM Symposium on Theory of Computing*, pp. 216–226. ACM Press, 1978.

19 U. Schöning. Graph isomorphism is in the low hierarchy. *Journal of Computer and System Sciences*, 37(3):312–323, 1988.

20 J. van Lint and R. Wilson. *A Course in Combinatorics.* (2nd ed.). CUP, 2001.