Non-negative Weighted #CSPs: An Effective Complexity Dichotomy

Jin-Yi Cai
University of Wisconsin, Madison

Xi Chen
Columbia University

Pinyan Lu
Microsoft Research Asia

Abstract

We prove a complexity dichotomy theorem for all non-negative weighted counting Constraint Satisfaction Problems (CSP). This caps a long series of important results on counting problems including unweighted and weighted graph homomorphisms [19, 8, 18, 12] and the celebrated dichotomy theorem for unweighted #CSP [6, 4, 21, 22]. Our dichotomy theorem gives a succinct criterion for tractability. If a set $\mathcal{F}$ of constraint functions satisfies the criterion, then the counting CSP problem defined by $\mathcal{F}$ is solvable in polynomial time; if it does not satisfy the criterion, then the problem is #P-hard. We furthermore show that the question of whether $\mathcal{F}$ satisfies the criterion is decidable in NP.

Surprisingly, our tractability criterion is simpler than the previous criteria for the more restricted classes of problems, although when specialized to those cases, they are logically equivalent. Our proof mainly uses Linear Algebra, and represents a departure from Universal Algebra, the dominant methodology in recent years.

1 Introduction

The study of Constraint Satisfaction Problems (CSP) has been one of the most active research areas, where enormous progress has been made in recent years. The investigation of CSP includes at least the following major branches: Decision Problems — whether a solution exists [36, 27, 3, 32]; Optimization Problems — finding a solution that satisfies the most constraints (or in the weighted case achieving the highest total weight) [26, 31, 1, 17, 34, 38, 35]; and Counting Problems — to count the number of solutions, including its weighted version [6, 4, 10, 7, 21]. The decision CSP dichotomy conjecture of Feder and Vardi [23], that every decision CSP problem defined by a constraint language Γ is either in P or NP-complete, remains open. A great deal of work has been devoted to the optimization version of CSP, constituting a significant fraction of on-going activities in approximation algorithms.

The subject of this paper is on counting CSP; more precisely on *weighted* counting Constraint Satisfaction Problems, denoted as weighted #CSP. For *unweighted* #CSP, the problem is usually stated as follows: D is a fixed finite set called the domain set. A fixed finite set of constraint predicates $\Gamma = \{\Theta_1, \ldots, \Theta_h\}$ is given, where each Θ_i is a relation on D^{r_i} of some finite arity r_i. Then an instance of #CSP(Γ) consists of a finite set of variables $x_1, \ldots, x_n$, ranging over D, and a finite set of constraints from Γ, each applied to a subset of these variables. It defines a new n-ary relation R where $(x_1, \ldots, x_n) \in R$ if and only if all the constraints are satisfied. The #CSP problem then asks for the size of R. In a (non-negatively) weighted #CSP, the set Γ is replaced by a fixed finite set of constraint functions, $\mathcal{F} = \{f_1, \ldots, f_h\}$, where each f_i maps D^{r_i} to non-negative reals $\mathbb{R}_+$. An instance of #CSP($\mathcal{F}$) consists of variables $x_1, \ldots, x_n$, ranging over D, and a finite set of constraint functions from $\mathcal{F}$, each applied to a subset of these variables. It defines a new n-ary function F: for any assignment $(x_1, \ldots, x_n)$, $F(x_1, \ldots, x_n)$ is the product of the constraint function evaluations. The output is then the so-called *partition function*, that is, the sum of F over all assignments $\{x_1, \ldots, x_n\} \to D$. The unweighted #CSP is the special case where each constraint function is 0-1 valued. (A formal definition will be given in Section 2.)

Regarding unweighted #CSP, Bulatov [4] proved a sweeping dichotomy theorem. He gave a criterion, *congruence singularity*, and showed that for any finite set of constraint predicates Γ over any finite domain D, if Γ satisfies this condition, then #CSP(Γ) is solvable in P; otherwise it is #P-complete. His proof uses deep structural theorems from universal algebra [11, 28, 24]. Indeed this approach using universal algebra has been one of the most exciting developments in the study of the complexity of CSP in recent years, first used in decision CSP [29, 30, 3, 2], and has been called the *Algebraic Approach*.

However, this is not the *only* approach. In [21], Dyer and Richerby gave an alternative proof of the dichotomy theorem for unweighted #CSP. Their proof is considerably more direct, and uses no universal algebra other than the notion of a Mal'tsev polymorphism. They also showed that the dichotomy is decidable [20, 22]. Furthermore, by treating rational weights as integral multiples of a common denominator, the dichotomy theorem can be extended to include positive rational weights [7].

In this paper, we give a complexity dichotomy theorem for all non-negative weighted #CSP($\mathcal{F}$). To describe our approach, let us first briefly recap the proofs by Bulatov and by Dyer and Richerby. Bulatov's proof is deeply embedded in a structural theory of universal algebra called *tame congruence theory* [28]. (A congruence is an equivalence relation expressible in a given universal algebra.) The starting point of this *Algebraic Approach* is the realization of a close connection between unweighted #CSP(Γ) and the *relational clone* $\langle\Gamma\rangle$ generated by Γ. $\langle\Gamma\rangle$ is the closure set of all relations expressible from Γ by boolean conjunction $\wedge$ and the existential quantifier $\exists$. A basic property, called *congruence permutability*, is then shown to be a necessary condition for the tractability of #CSP(Γ) [9, 6, 10]. It is known from universal algebra that congruence permutability is equivalent to the existence of *Mal'tsev polymorphisms*. It is also equivalent to the more combinatorial condition of *strong rectangularity* of

Dyer and Richerby [21]: For any n-ary relation R defined by an instance of #CSP(Γ), if we partition its n variables into three parts: $\mathbf{u} = (u_1, \ldots, u_k), \mathbf{v} = (v_1, \ldots, v_\ell)$ and $\mathbf{w} = (w_1, \ldots, w_{n-k-\ell})$, then the following $|D|^k \times |D|^\ell$ matrix $\mathbf{M}$ must be *block-diagonal* after separately permuting its rows and columns: $M(\mathbf{u}, \mathbf{v}) = 1$ if there exists a $\mathbf{w}$ such that $(\mathbf{u}, \mathbf{v}, \mathbf{w}) \in R$; and $M(\mathbf{u}, \mathbf{v}) = 0$ otherwise. (See the formal definition in Section 2.)

Assuming Γ satisfies this necessary condition (otherwise #CSP(Γ) is already #P-hard), Bulatov's proof delves much more deeply than Mal'tsev polymorphisms and uses a lot more results and techniques from universal algebra. The Dyer-Richerby proof manages to avoid much of universal algebra. They went on to give a more combinatorial criterion, called *strong balance*: For any n-ary relation R defined by an instance of #CSP(Γ), if we partition its n variables into four parts: $\mathbf{u} = (u_1, \ldots, u_k), \mathbf{v} = (v_1, \ldots, v_\ell), \mathbf{w} = (w_1, \ldots, w_t), \mathbf{z} = (z_1, \ldots, z_{n-k-\ell-t})$, then the following $|D|^k \times |D|^\ell$ integer matrix $\mathbf{M}$ must be block-diagonal and *all of its blocks are of rank* 1 (which we will refer to as a *block-rank-1* matrix):

$$M(\mathbf{u}, \mathbf{v}) = \Big|\big\{\mathbf{w} : \exists\, \mathbf{z} \text{ such that } (\mathbf{u}, \mathbf{v}, \mathbf{w}, \mathbf{z}) \in R\big\}\Big|, \qquad \text{for all } \mathbf{u} \in D^k \text{ and } \mathbf{v} \in D^\ell. \tag{1}$$

(See the formal definition in Section 9.) Dyer and Richerby [21] show that strong balance (which implies strong rectangularity) is the criterion for the tractability of #CSP(Γ). They further prove that it is equivalent to Bulatov's criterion of *congruence singularity* which is stated in the language of universal algebra.

The first difficulty we encountered when trying to extend the unweighted dichotomy to weighted #CSP($\mathcal{F}$) is that there is no direct extension of the notion of strong balance above in the weighted world. While the number of $\mathbf{w}$ satisfying R on the right side of (1) can be naturally replaced by the sum of F (any function defined by an #CSP($\mathcal{F}$) instance) over $\mathbf{w}$, we do not see any easy way to introduce existential quantifiers to this more general weighted setting. Moreover, the use of existential quantifiers in the notion of strong balance is crucial to the proof of Dyer and Richerby: their polynomial-time counting algorithm for tractable #CSP(Γ) heavily relies on them.

While there seems to be no natural notion of an existential quantifier in the weighted setting, we came to a key observation that the notion of strong balance is equivalent to the one *without using any existential quantifiers* (that is, we only consider partitions of the variables into 3 parts with no $\mathbf{z}$). We include the proof of this equivalence in Section 9. This inspires us to use the following seemingly weaker notion of *balance* for weighted #CSP($\mathcal{F}$), with no existential quantifiers at all: For any n-ary function F defined by a #CSP($\mathcal{F}$) instance, if we partition its n variables into three parts: $\mathbf{u} = (u_1, \ldots, u_k), \mathbf{v} = (v_1, \ldots, v_\ell)$ and $\mathbf{w} = (w_1, \ldots, w_{n-k-\ell})$, then the following $|D|^k \times |D|^\ell$ matrix $\mathbf{M}$ must be *block-rank-1*:

$$M(\mathbf{u}, \mathbf{v}) = \sum_{\mathbf{w} \in D^{n-k-\ell}} F(\mathbf{u}, \mathbf{v}, \mathbf{w}), \qquad \text{for all } \mathbf{u} \in D^k \text{ and } \mathbf{v} \in D^\ell.$$

It is easy to show that balance is a necessary condition for the tractability of #CSP($\mathcal{F}$). But is it also sufficient? If $\mathcal{F}$ is balanced, can we solve it in polynomial time? We show that this is indeed the case by giving a polynomial time counting scheme for all #CSP($\mathcal{F}$)s with $\mathcal{F}$ being balanced. Our algorithm works differently from the one of Dyer and Richerby. It avoids the use of existential quantifiers and is designed specially for weighted and balanced #CSP($\mathcal{F}$)s. As a result, we get the following dichotomy for non-negatively weighted #CSP with a logically simpler criterion:

Theorem 1 (Main)**.** *#CSP($\mathcal{F}$) is in polynomial-time if $\mathcal{F}$ is balanced; and is #P-hard otherwise.*

A new ingredient of our proof is the concept of *a vector representation* for a non-negative function. Let F be a function over $x_1, \ldots, x_n$. Then $s_1, \ldots, s_n : D \to \mathbb{R}_+$ is a vector representation of F if for any $(x_1, \ldots, x_n) \in D^n$

such that $F(x_1, \ldots, x_n) > 0$, we have $F(x_1, \ldots, x_n) = s_1(x_1) \cdots s_n(x_n)$. The first step of our algorithm is to show that given any instance of #CSP($\mathcal{F}$), where $\mathcal{F}$ is balanced, the function it defines has a vector representation which can be computed in polynomial time. However, F may have a lot of "holes" where $s_1(x_1) \cdots s_n(x_n) > 0$ but $F(x_1, \ldots, x_n) = 0$ so it is still not clear how to do the sum of F over $x_1, \ldots, x_n$.

The next step is quite a surprise. Assuming $\mathcal{F}$ is balanced, we show how to construct one-variable functions $t_2, \ldots, t_n : D \to \mathbb{R}_+$ in polynomial time such that for any $(u_1, \ldots, u_n) \in D^n$ with $F(u_1, \ldots, u_n) > 0$, we have

$$\sum_{x_2,\ldots,x_n \in D} F(u_1, x_2, \ldots, x_n) = s_1(u_1) \cdot \prod_{j=2}^{n} \frac{s_j(u_j)}{t_j(u_j)}. \tag{2}$$

The intriguing part of (2) is that its left side only depends on u_1 but it holds for any $(u_1, \ldots, u_n) \in D^n$ as long as $F(u_1, \ldots, u_n) > 0$. A crucial ingredient we use in constructing $t_2, \ldots, t_n$ and proving (2) here is the succinct data structure called *frame* introduced by Dyer and Richerby for unweighted #CSP [21] (which is similar to the "compact representation" of Bulatov and Dalmau [5]). Once we have $t_2, \ldots, t_n$ and (2), computing the partition function becomes trivial.

After obtaining the dichotomy, we also show in Section 6 that the tractability criterion (that is, whether $\mathcal{F}$ is balanced or not) is decidable in NP. The proof follows the approach of Dyer and Richerby [20] for unweighted #CSP, with new ideas and constructions developed for the weighted setting.

This advance, from unweighted to weighted #CSP, is akin to the leap from the Dyer-Greenhill result on counting 0-1 graph homomorphisms [19] to the Bulatov-Grohe result for the non-negative case [8]. The Bulatov-Grohe result paved the way for all future developments. This is because not only the Bulatov-Grohe result is intrinsically important and sweeping but also they gave an elegant dichotomy criterion, which allows its easy application. Almost all future results in this area use the Bulatov-Grohe criterion. Here our result covers all non-negative counting CSP. It achieves a similar leap from the 0-1 case of Bulatov and Dyer-Richerby, and in the meanwhile, simplifies the dichotomy criterion. Therefore it is hoped that it will also be useful for future research.

In hindsight, perhaps one may re-evaluate the *Algebraic Approach*. We now know that there is another *Algebraic Approach*, based primarily on matrix algebra rather than (relational) universal algebra, which gives us a more direct and complete dichotomy theorem for #CSPs. It is perhaps also a case where the proper generalization, namely weighted #CSP, leads to a simpler resolution of the problem than the original unweighted #CSP.

Weighted #CSP has many special cases that have been studied intensively. Graph homomorphisms can be considered as a special case of weighted #CSP where there is only one binary constraint function. There has been great advances made on graph homomorphisms [19, 8, 18, 12]. Our dichotomy theorem generalizes all previous dichotomy theorems where the constraint functions are non-negative. Looking beyond non-negatively weighted counting type problems, in graph homomorphisms [25, 13, 37] great progress has already been made. To extend that to #CSPs with real or even complex weights will require significantly more effort (even for directed graph homomorphisms [12]). For Boolean #CSP with complex weights, a dichotomy was obtained [15]. Going beyond CSP type problems, holographic algorithms and reductions are aimed precisely at these counting problems where cancelation is the main feature. The work on Holant problems and their dichotomy theorems are the beginning steps in that direction [15, 16, 14].

2 Preliminaries

We start with some definitions about non-negative matrices.

Let $\mathbf{M}$ be a non-negative $m \times n$ matrix. We say $\mathbf{M}$ is *rectangular* if one can permute its rows and columns separately, so that $\mathbf{M}$ becomes a block-diagonal matrix. More exactly, $\mathbf{M}$ is rectangular if there exist s pairwise disjoint and nonempty subsets of $[m]$, denoted by $A_1, \ldots, A_s$, and s pairwise disjoint and nonempty subsets of $[n]$, denoted by $B_1, \ldots, B_s$, for some $s \geq 0$, such that for all $i \in [m]$ and $j \in [n]$,

$$M(i,j) > 0 \iff i \in A_k \text{ and } j \in B_k \text{ for some } k \in [s].$$

Now let $\mathbf{M}$ be a non-negative and rectangular $m \times n$ matrix with s blocks $A_1 \times B_1, \ldots, A_s \times B_s$. We say it is *block-rank-*1 if the $A_k \times B_k$ sub-matrix of $\mathbf{M}$, for every $k \in [s]$, is of rank 1.

The two lemmas below then follow directly from the definition of block-rank-1 matrices:

Lemma 1. *Let* $\mathbf{M}$ *be a block-rank-*1 *matrix with* $s \geq 1$ *blocks:* $A_1 \times B_1, \ldots, A_s \times B_s$. *If* $i^* \in A_k$ *and* $j^* \in B_k$ *for some* $k \in [s]$, *then for any* $i \in A_k$ *we have*

$$\frac{\sum_{j \in B_k} M(i,j)}{\sum_{j \in B_k} M(i^*,j)} = \frac{M(i,j^*)}{M(i^*,j^*)}.$$

Lemma 2. *If* $\mathbf{M}$ *is a non-negative matrix but is not block-rank-*1, *then there exist two rows of* $\mathbf{M}$ *that are neither linearly dependent nor orthogonal.*

2.1 Basic #P-Hardness About Counting Graph Homomorphisms

Every symmetric and non-negative $n \times n$ matrix $\mathbf{A}$ defines a graph homomorphism (or partition) function $Z_{\mathbf{A}}(\cdot)$ as follows: Given any undirected graph $G = (V, E)$, we have

$$Z_{\mathbf{A}}(G) \overset{\text{def}}{=} \sum_{\xi: V \to [n]} \prod_{uv \in E} A\big(\xi(u), \xi(v)\big).$$

We need the following important result of Bulatov and Grohe [8] to derive the hardness part of our dichotomy:

Theorem 2. *Let* $\mathbf{A}$ *be a symmetric and non-negative matrix with algebraic entries, then the problem of computing* $Z_{\mathbf{A}}(\cdot)$ *is in polynomial time if* $\mathbf{A}$ *is* block-rank-1*; and is #P-hard otherwise.*

2.2 Weighted #CSPs

Let $D = \{1, 2, \ldots, d\}$ be the domain set, where the size d will be considered as a constant. A *weighted* constraint language $\mathcal{F}$ over the domain D is a finite set of functions $\{f_1, \ldots, f_h\}$ in which $f_i : D^{r_i} \to \mathbb{R}$ is an r_i-ary function over D for some $r_i \geq 1$. The arity r_i of f_i, $i \in [h]$, the number of functions h in $\mathcal{F}$, as well as the values of f_i, will all be considered as constants (except in Section 6 where the decidability of the dichotomy is discussed). In this paper, we only consider *non-negative* weighted constraint languages in which every f_i maps D^{r_i} to *non-negative* and *algebraic* numbers.

The pair $(D, \mathcal{F})$ defines the following problem which we simply denote by $(D, \mathcal{F})$:

1. Let $\mathbf{x} = (x_1, \ldots, x_n) \in D^n$ be a set of n variables over D. The input is then a collection I of m tuples $(f, i_1, \ldots, i_r)$ in which f is an r-ary function in $\mathcal{F}$ and $i_1, \ldots, i_r \in [n]$. We call $n + m$ the size of I.

2. The input I defines the following function F_I over $\mathbf{x} = (x_1, \ldots, x_n) \in D^n$:

$$F_I(\mathbf{x}) \stackrel{\text{def}}{=} \prod_{(f, i_1, \ldots, i_r) \in I} f(x_{i_1}, \ldots, x_{i_r}), \qquad \text{for every } \mathbf{x} \in D^n.$$

And the output of the problem is the following sum:

$$Z(I) \stackrel{\text{def}}{=} \sum_{\mathbf{x} \in D^n} F_I(\mathbf{x}).$$

2.3 Reduction from Unweighted to Weighted #CSPs

A special case is when every function in the language is boolean. In this case, we can view each of the functions as a relation. We use the following notation for this special case.

An *unweighted* constraint language Γ over the domain set D is a finite set of relations $\{\Theta_1, \ldots, \Theta_h\}$ in which every Θ_i is an r_i-ary relation over D^{r_i} for some $r_i \geq 1$. The language Γ defines the following problem which we denote by (D, Γ):

1. Let $\mathbf{x} = (x_1, \ldots, x_n) \in D^n$ be a set of n variables over D. The input is then a collection I of m tuples $(\Theta, i_1, \ldots, i_r)$ in which Θ is an r-ary relation in Γ and $i_1, \ldots, i_r \in [n]$. We call $n + m$ the size of I.

2. The input I defines the following relation R_I over $\mathbf{x} = (x_1, \ldots, x_n) \in D^n$:

$$\mathbf{x} \in R_I \iff \text{for every tuple } (\Theta, i_1, \ldots, i_r) \in I, \text{ we have } (x_{i_1}, \ldots, x_{i_r}) \in \Theta.$$

And the output of the problem is the number of $\mathbf{x} \in D^n$ in the relation R_I.

For any non-negative weighted constraint language $\mathcal{F} = \{f_1, \ldots, f_h\}$, it is natural to define its corresponding unweighted constraint language $\Gamma = \{\Theta_1, \ldots, \Theta_h\}$, where $\mathbf{x} \in \Theta_i$ if and only if $f_i(\mathbf{x}) > 0$, for all $i \in [h]$ and $\mathbf{x} \in D^{r_i}$. In Section 7, we give a polynomial-time reduction from (D, Γ) to $(D, \mathcal{F})$.

Lemma 3. *Problem (D, Γ) is polynomial-time reducible to $(D, \mathcal{F})$.*

Corollary 1. *If $(D, \mathcal{F})$ is not #P-hard, then neither is (D, Γ).*

2.4 Strong Rectangularity

In the proof of the complexity dichotomy theorem for unweighted #CSPs [4, 21], an important necessary condition for (D, Γ) being not #P-hard is *strong rectangularity*:

Definition 1 (Strong Rectangularity)**.** *We say* Γ *is* strongly rectangular *if for any input* I *of* (D,Γ) (*which defines an* n*-ary relation* R_I *over* $(x_1,\ldots,x_n)\in D^n$) *and for any integers* $a,b: 1\le a<b\le n$*, the following* $d^a\times d^{b-a}$ *matrix* $\mathbf{M}$ *is* rectangular*: the rows of* $\mathbf{M}$ *are indexed by* $\mathbf{u}\in D^a$ *and the columns are indexed by* $\mathbf{v}\in D^{b-a}$*, and*

$$M(\mathbf{u},\mathbf{v})=\Big|\big\{\mathbf{w}\in D^{n-b}:(\mathbf{u},\mathbf{v},\mathbf{w})\in R_I\big\}\Big|,\quad \textit{for all } \mathbf{u}\in D^a \textit{ and } \mathbf{v}\in D^{b-a}.$$

For the special case when $b=n$*, we have* $M(\mathbf{u},\mathbf{v})=1$ *if* $(\mathbf{u},\mathbf{v})\in R_I$ *and* $M(\mathbf{u},\mathbf{v})=0$ *otherwise.*

The following theorem can be found in [4] and [21]:

Theorem 3. *If* Γ *is not strongly rectangular, then* (D,Γ) *is #P-hard.*

As a result, if $(D,\mathcal{F})$ is not #P-hard, then Γ must be strongly rectangular by Corollary 1 and Theorem 3, where Γ is the unweighted language that corresponds to $\mathcal{F}$. The strong rectangularity of Γ then gives us the following algorithmic results from [20], using the succinct and efficiently computable data structure called frame. They turn out to be very useful later in the study of the original weighted problem $(D,\mathcal{F})$. We start with some notation.

Let I be an input instance of (D,Γ) which defines a relation R over n variables $\mathbf{x}=(x_1,\ldots,x_n)$.

Definition 2. *For any* $i\in[n]$*, we use* $\mathsf{pr}_i R\subseteq D$ *to denote the projection of* R *on the* i*th coordinate:* $a\in\mathsf{pr}_i R$ *if and only if there exist tuples* $\mathbf{u}\in D^{i-1}$ *and* $\mathbf{v}\in D^{n-i}$ *such that* $(\mathbf{u},a,\mathbf{v})\in R$*.*

We define the following relation $\sim_i$ *on* $\mathsf{pr}_i R$*:* $a\sim_i b$ *if there exist tuples* $\mathbf{u}\in D^{i-1}$ *and* $\mathbf{v}_a,\mathbf{v}_b\in D^{n-i}$ *such that* $(\mathbf{u},a,\mathbf{v}_a)\in R$ *and* $(\mathbf{u},b,\mathbf{v}_b)\in R$*.*

Lemma 4 ([20])**.** *If* Γ *is strongly rectangular then given any input* I *of* (D,Γ) *which defines a relation* R*, we have*

(A). *For any* $i\in[n]$*, we can compute the set* $\mathsf{pr}_i R$ *in polynomial time in the size of* I*. Moreover, for every* $a\in\mathsf{pr}_i R$*, we can find a tuple* $\mathbf{u}\in R$ *such that* $u_i=a$ *in polynomial time.*

(B). *For any* $i\in[n]$*, the relation* $\sim_i$ *must be an* equivalence *relation and can be computed in polynomial time. We will use* $\mathcal{E}_{i,k}\subseteq D$*,* $k=1,2,\ldots,$ *to denote the equivalent classes of* $\sim_i$*.*

(C). *For any equivalence class* $\mathcal{E}_{i,k}$*, we can find, in polynomial time, a tuple* $\mathbf{u}^{[i,k]}\in D^{i-1}$ *as well as a tuple* $\mathbf{v}^{[i,k,a]}\in D^{n-i}$ *for each element* $a\in\mathcal{E}_{i,k}$ *such that* $(\mathbf{u}^{[i,k]},a,\mathbf{v}^{[i,k,a]})\in R$ *for all* $a\in\mathcal{E}_{i,k}$*.*

As a corollary, if $(D,\mathcal{F})$ is not #P-hard, then we are able to use all the algorithmic results above for (D,Γ) as subroutines, in the quest of finding a polynomial-time algorithm for $(D,\mathcal{F})$.

3 A Dichotomy for Non-negative Weighted #CSPs and its Decidability

In this section, we prove a dichotomy theorem for all non-negative weighted #CSPs and show that the characterization can be checked in NP. The lemmas used in the proofs will be proved in the rest of the paper.

In the proof of our dichotomy theorem as well as its decidability, the following two notions of *weak balance* and *balance* play a crucial role. It is similar to and, in some sense, weaker than the concept of *strong balance* used in [20]. (Notably we do not use any existential quantifier in the definitions.)

Definition 3 (Weak Balance)**.** *We say $\mathcal{F}$ is* weakly balanced *if for any input instance I of $(D, \mathcal{F})$ (which defines a non-negative function $F(x_1, \ldots, x_n)$ over D) and for any integer $a : 1 \le a < n$, the following $d^a \times d$ matrix $\mathbf{M}$ is* block-rank-1*: the rows of $\mathbf{M}$ are indexed by $\mathbf{u} \in D^a$ and the columns are indexed by $v \in D$, and*

$$M(\mathbf{u}, v) = \sum_{\mathbf{w} \in D^{n-a-1}} F(\mathbf{u}, v, \mathbf{w}), \quad \textit{for all } \mathbf{u} \in D^a \textit{ and } v \in D.$$

For the special case when $a + 1 = n$, we have $M(\mathbf{u}, v) = F(\mathbf{u}, v)$ is block-rank-1.

Definition 4 (Balance)**.** *We call $\mathcal{F}$* balanced *if for any input instance I of $(D, \mathcal{F})$ (which defines a non-negative function $F(x_1, \ldots, x_n)$ over D) and for any integers $a, b : 1 \le a < b \le n$, the following $d^a \times d^{b-a}$ matrix $\mathbf{M}$ is* block-rank-1*: the rows of $\mathbf{M}$ are indexed by $\mathbf{u} \in D^a$ and the columns are indexed by $\mathbf{v} \in D^{b-a}$, and*

$$M(\mathbf{u}, \mathbf{v}) = \sum_{\mathbf{w} \in D^{n-b}} F(\mathbf{u}, \mathbf{v}, \mathbf{w}), \quad \textit{for all } \mathbf{u} \in D^a \textit{ and } \mathbf{v} \in D^{b-a}.$$

For the special case when $b = n$, we have $M(\mathbf{u}, \mathbf{v}) = F(\mathbf{u}, \mathbf{v})$ is block-rank-1.

It is clear that *balance* implies *weak balance*. We prove the following complexity dichotomy theorem.

Theorem 4. *$(D, \mathcal{F})$ is in P if Γ is strongly rectangular and $\mathcal{F}$ is weakly balanced; and is #P-hard otherwise.*

Proof. Assume $(D, \mathcal{F})$ is not #P-hard. By Corollary 1 and Theorem 3, Γ must be strongly rectangular. We prove the following lemma in Section 8, showing that $\mathcal{F}$ must be balanced and thus, weakly balanced:

Lemma 5. *If $\mathcal{F}$ is not balanced, then $(D, \mathcal{F})$ is #P-hard.*

In the next two sections (Sections 4 and 5) we focus on the proof of the following algorithmic lemma:

Lemma 6. *If Γ is strongly rectangular and $\mathcal{F}$ is weakly balanced, then $(D, \mathcal{F})$ is in polynomial time.*

The dichotomy theorem then follows directly. □

While the characterization of the dichotomy in Theorem 4 above is very useful in the proof of its decidability, we can easily simplify it without using strong rectangularity. We prove the following equivalent characterization using the notion of balance:

Lemma 7. *$(D, \mathcal{F})$ is in polynomial time if $\mathcal{F}$ is balanced; and is #P-hard otherwise.*

Proof. Assume $(D, \mathcal{F})$ is not #P-hard; otherwise we are already done. By Lemma 5, we know $\mathcal{F}$ must be balanced. By Theorem 4, it suffices to show that if $\mathcal{F}$ is balanced, then Γ is strongly rectangular, where we use Γ to denote the unweighted constraint language that corresponds to $\mathcal{F}$. This follows directly from the definitions of strong rectangularity and balance, since a matrix that is block-rank-1 must first be rectangular. □

Next, we show that the complexity dichotomy is efficiently decidable. Given D and $\mathcal{F}$, the decision problem of whether $(D, \mathcal{F})$ is in P or #P-hard is actually in NP. (Note that here D and $\mathcal{F} = \{f_1, \ldots, f_h\}$ are considered no longer as constants, but as the input of the decision problem. The input size is d plus the number of bits needed to describe $f_1, \ldots, f_h$.) We prove the following theorem in Section 6. The proof follows the approach of Dyer and Richerby [20], with new ideas and constructions developed for the more general weighted case. It uses a method of Lovász [33], which was also used in [18].

Theorem 5. *Given D and $\mathcal{F}$, the problem of deciding whether $(D, \mathcal{F})$ is in P or #P-hard is in* NP.

4 Vector Representation

Assume $\mathcal{F}$ is weakly balanced, and let f be an r-ary function in $\mathcal{F}$. We use Θ to denote the corresponding r-ary relation of f in Γ. In this section, we show that there must exist r non-negative one-variable functions $s_1, \ldots, s_r : D \to \mathbb{R}_+$, such that for all $\mathbf{x} \in D^r$, either $\mathbf{x} \notin \Theta$ and $f(\mathbf{x}) = 0$; or we have $f(\mathbf{x}) = s_1(x_1) \cdots s_r(x_r)$. We call any $\mathbf{s} = (s_1, \ldots, s_r)$ that satisfies the property above a *vector representation* of f. We prove the following lemma:

Lemma 8. *If $\mathcal{F}$ is weakly balanced, then every function $f \in \mathcal{F}$ has a vector representation.*

To this end we need the following notation. Let f be any r-ary function over D. Then for any $\ell \in [r]$, we use $f^{[\ell]}$ to denote the following ℓ-ary function over D:

$$f^{[\ell]}(x_1, \ldots, x_\ell) \overset{\text{def}}{=} \sum_{x_{\ell+1}, \ldots, x_r \in D} f(x_1, \ldots, x_\ell, x_{\ell+1}, \ldots, x_r), \qquad \text{for all } x_1, \ldots, x_\ell \in D.$$

In particular, we have $f^{[r]} \equiv f$.

Let f be an r-ary non-negative function with $r \geq 1$. We say f is *block-rank-*1 if either $r = 1$; or the following $d^{r-1} \times d$ matrix $\mathbf{M}$ is block-rank-1: the rows of $\mathbf{M}$ are indexed by $\mathbf{u} \in D^{r-1}$ and the columns are indexed by $v \in D$, and $M(\mathbf{u}, v) = f(\mathbf{u}, v)$ for all $\mathbf{u} \in D^{r-1}$ and $v \in D$.

By the definition of weak balance, Lemma 8 is a direct corollary of the following lemma:

Lemma 9. *Let $f(x_1, \ldots, x_r)$ be an r-ary non-negative function. If $f^{[\ell]}$ is block-rank-*1 *for all $\ell \in [r]$, then f has a vector representation* $\mathbf{s}$.

Proof. We prove the lemma by induction on r, the arity of f.

The base case when $r = 1$ is trivial. Now assume for induction that the claim is true for all $(r-1)$-ary non-negative functions, for some $r \geq 2$. Let f be an r-ary non-negative function such that $f^{[\ell]}$ is block-rank-1 for all $\ell \in [r]$. By definition, it is easy to see that

$$\left(f^{[r-1]}\right)^{[\ell]} = f^{[\ell]}, \qquad \text{for all } \ell \in [r-1].$$

As a result, if we denote $f^{[r-1]}$, an $(r-1)$-ary non-negative function, by g, then $g^{[\ell]}$ is block-rank-1 for every $\ell \in [r-1]$. Therefore, by the inductive hypothesis, $g = f^{[r-1]}$ has a vector representation $(s_1, \ldots, s_{r-1})$.

Finally, we show how to construct s_r so that $(s_1, \ldots, s_{r-1}, s_r)$ is a vector representation of f. To this end, we let $\mathbf{M}$ denote the following $d^{r-1} \times d$ matrix: The rows are indexed by $\mathbf{u} \in D^{r-1}$ and the columns are indexed by $v \in D$, and $M(\mathbf{u}, v) = f(\mathbf{u}, v)$ for all $\mathbf{u} \in D^{r-1}$ and $v \in D$. By the assumption we know that $\mathbf{M}$ is block-rank-1. Therefore, by definition, there exist pairwise disjoint and nonempty subsets of D^{r-1}, denoted by $A_1, \ldots, A_s$, and pairwise disjoint and nonempty subsets of D, denoted by $B_1, \ldots, B_s$, for some $s \geq 0$, such that $M(\mathbf{u}, v) > 0$ if, and only if $\mathbf{u} \in A_i$ and $v \in B_i$ for some $i \in [s]$; and for every $i \in [s]$, the $A_i \times B_i$ sub-matrix of $\mathbf{M}$ is of rank 1.

We now construct $s_r : D \to \mathbb{R}_+$ as follows. For every $i \in [s]$, we arbitrarily pick a vector from A_i and denote it $\mathbf{u}_i$. Then for $v \in D$, we set $s_r(v)$ as follows:

1. If $v \notin B_i$ for any $i \in [s]$, then $s_r(v) = 0$; and

2. Otherwise, assume $v \in B_i$. Then

$$s_r(v) = \frac{M(\mathbf{u}_i, v)}{\sum_{v' \in B_i} M(\mathbf{u}_i, v')}. \tag{3}$$

To prove that $(s_1, \ldots, s_r)$ is actually a vector representation of f, we only need to show that for every tuple $(\mathbf{u}, v)$ such that $\mathbf{u} \in A_i$ and $v \in B_i$ for some $i \in [s]$ (since otherwise we have $f(\mathbf{u}, v) = 0$), we have

$$f(\mathbf{u}, v) = M(\mathbf{u}, v) = s_r(v) \prod_{j \in [r-1]} s_j(u_j).$$

By using Lemma 1 and (3), we have

$$M(\mathbf{u}, v) = M(\mathbf{u}_i, v) \cdot \frac{\sum_{v' \in B_i} M(\mathbf{u}, v')}{\sum_{v' \in B_i} M(\mathbf{u}_i, v')} = s_r(v) \cdot f^{[r-1]}(\mathbf{u}) = s_r(v) \prod_{j \in [r-1]} s_j(u_j),$$

where the last equation above follows from the inductive hypothesis that $(s_1, \ldots, s_{r-1})$ is a vector representation of $g = f^{[r-1]}$. This finishes the induction, and the lemma is proved. □

5 Tractability: The Counting Algorithm

In this section, we prove Lemma 6 by giving a polynomial-time algorithm for the problem $(D, \mathcal{F})$, assuming Γ is strongly rectangular and $\mathcal{F}$ is weakly balanced. As mentioned earlier, because Γ is strongly rectangular we can use the three polynomial-time algorithms described in Lemma 4 as subroutines. Also because $\mathcal{F}$ is weakly balanced, we may assume, by Lemma 8, that every r-ary function f in $\mathcal{F}$ has a vector representation $\mathbf{s}_f = (s_{f,1}, \ldots, s_{f,r})$, where $s_{f,i} : D \to \mathbb{R}_+$ for all $i \in [r]$.

Now let I be an input instance of $(D, \mathcal{F})$ and let F denote the function it defines over $\mathbf{x} = (x_1, \ldots, x_n) \in D^n$. For each tuple in I, one can replace the first component, that is, a function f in $\mathcal{F}$, by its corresponding relation Θ in Γ. We use I' to denote the new set, which is clearly an input instance of (D, Γ) and defines a relation R over $\mathbf{x} \in D^n$. We have $F(\mathbf{x}) > 0$ if and only if $\mathbf{x} \in R$, for all $\mathbf{x} \in D^n$.

The first step of our algorithm is to construct a vector representation $\mathbf{s} = (s_1, \ldots, s_n)$ of F, using the vector representations $\mathbf{s}_f$ of f, $f \in \mathcal{F}$:

Lemma 10. *Given I, one can compute $s_1(\cdot), \ldots, s_n(\cdot)$ in polynomial time such that for all $\mathbf{x} \in D^n$, either $\mathbf{x} \notin R$ and $F(\mathbf{x}) = 0$; or $F(\mathbf{x}) = s_1(x_1) \cdots s_n(x_n)$.*

Proof. We start with $s_1, \ldots, s_n$ where $s_i(a) = 1$ for all $i \in [n]$ and $a \in D$. We then enumerate the tuples in I one by one. For each $(f, i_1, \ldots, i_r) \in I$ and each $j \in [r]$, we update the function $s_{i_j}(\cdot)$ using $s_{f,j}(\cdot)$ as follows:

$$s_{i_j}(a) \overset{\text{set}}{=} s_{i_j}(a) \cdot s_{f,j}(a), \qquad \text{for every } a \in D.$$

It is easy to check that the tuple $(s_1, \ldots, s_n)$ we get is a vector representation of F. □

The second step of the algorithm is to construct a sequence of one-variable functions $t_n(\cdot), t_{n-1}(\cdot), \dots, t_2(\cdot)$ that have the following nice property: for any $i \in \{1, \dots, n-1\}$ and for any $\mathbf{u} \in R$, we have

$$\sum_{x_{i+1},\dots,x_n \in D} F(u_1, \dots, u_i, x_{i+1}, \dots, x_n) = s_1(u_1) \cdots s_i(u_i) \cdot \frac{s_{i+1}(u_{i+1})}{t_{i+1}(u_{i+1})} \cdots \frac{s_n(u_n)}{t_n(u_n)}. \tag{4}$$

Before giving the construction and proving (4), we show that $Z(I)$ is easy to compute once we have $t_n, \dots, t_2$.

For this purpose, we first compute $\mathsf{pr}_1 R$ in polynomial time using the algorithm in Lemma 4 (A). In addition, we find a vector $\mathbf{u}_a = (u_{a,1}, u_{a,2}, \dots, u_{a,n}) \in R$ for each $a \in \mathsf{pr}_1 R$ such that $u_{a,1} = a$ in polynomial time. Then

$$Z(I) = \sum_{\mathbf{x} \in D^n} F(\mathbf{x}) = \sum_{a \in \mathsf{pr}_1 R} \sum_{x_2,\dots,x_n \in D} F(a, x_2, \dots, x_n) = \sum_{a \in \mathsf{pr}_1 R} s_1(a) \prod_{j \in [2:n]} \left(\frac{s_j(u_{a,j})}{t_j(u_{a,j})} \right),$$

which clearly can be evaluated in polynomial time using $s_1, \dots, s_n$ and $t_2, \dots, t_n$.

Now we construct $t_n, t_{n-1}, \dots, t_2$ and prove (4) by induction. We start with $t_n(\cdot)$.

Because $\mathcal{F}$ is weakly balanced, the following $d^{n-1} \times d$ matrix $\mathbf{M}$ must be block-rank-1: the rows are indexed by $\mathbf{u} \in D^{n-1}$ and the columns are indexed by $v \in D$, and $M(\mathbf{u}, v) = F(\mathbf{u}, v)$ for all $\mathbf{u} \in D^{n-1}$ and $v \in D$. By the definition of $\sim_n$, we have $v_1 \sim_n v_2$ if and only if columns v_1 and v_2 are in the same block of $\mathbf{M}$ and thus, the equivalent classes $\{\mathcal{E}_{n,k}\}$ are exactly the column index sets of those blocks of $\mathbf{M}$.

We define $t_n(\cdot)$ as follows. For every $a \in D$, if $a \notin \mathsf{pr}_n R$ then $t_n(a) = 0$; Otherwise, a belongs to one of the equivalence classes $\mathcal{E}_{n,k}$ of $\sim_n$ and

$$t_n(a) = \frac{s_n(a)}{\sum_{b \in \mathcal{E}_{n,k}} s_n(b)}. \tag{5}$$

By using the algorithm in Lemma 4 (B) $t_n(\cdot)$ can be constructed efficiently. We now prove (4) for $i = n-1$. Given any $\mathbf{u} \in R$, we have $u_n \in \mathsf{pr}_n R$ by definition and let $\mathcal{E}_{n,k}$ denote the equivalence class that u_n belongs to. Then

$$\sum_{b \in D} F(u_1, \dots, u_{n-1}, b) = \sum_{b \in \mathcal{E}_{n,k}} F(u_1, \dots, u_{n-1}, b) = \prod_{j \in [n-1]} s_j(u_j) \sum_{b \in \mathcal{E}_{n,k}} s_n(b) = \prod_{j \in [n-1]} s_j(u_j) \cdot \frac{s_n(u_n)}{t_n(u_n)}.$$

The last equation follows from the construction (5) of $t_n(\cdot)$ and the assumption that $u_n \in \mathcal{E}_{n,k}$.

Now assume for induction that we already constructed $t_{i+1}, \dots, t_n$, for some $i \in [2 : n-1]$, and they satisfy (4). To construct $t_i(\cdot)$, we first observe that the following $d^{i-1} \times d$ matrix $\mathbf{M}$ must be block-rank-1, because $\mathcal{F}$ is weakly balanced: the rows are indexed by $\mathbf{u} = (u_1, \dots, u_{i-1}) \in D^{i-1}$ and the columns are indexed by $v \in D$,

$$M(\mathbf{u}, v) = \sum_{\mathbf{w} \in D^{n-i}} F(\mathbf{u}, v, \mathbf{w}) = \sum_{(\mathbf{u}, v, \mathbf{w}) \in R} F(\mathbf{u}, v, \mathbf{w}).$$

Similarly, by the definition of $\sim_i$, its equivalent classes $\{\mathcal{E}_{i,k}\}$ are precisely the column index sets of those blocks of $\mathbf{M}$. By (4) and the inductive hypothesis we immediately have the following concise form for $M(\mathbf{u}, v)$: for any $\mathbf{w} = (w_{i+1}, \dots, w_n) \in D^{n-i}$ such that $(\mathbf{u}, v, \mathbf{w}) \in R$, we have

$$M(\mathbf{u}, v) = \left(\prod_{j \in [i-1]} s_j(u_j) \right) s_i(v) \left(\prod_{j \in [i+1:n]} \frac{s_j(w_j)}{t_j(w_j)} \right). \tag{6}$$

Note that by (4), the choice of $\mathbf{w}$ can be arbitrary as long as $(\mathbf{u}, v, \mathbf{w}) \in R$.

We now construct $t_i(\cdot)$. For every $a \in D$,

1. If $a \notin \mathsf{pr}_i R$, then $t_i(a) = 0$; and
2. Otherwise, let $\mathcal{E}_{i,k}$ denote the equivalence class of $\sim_i$ that a belongs to. Then by using the algorithm in Lemma 4 (C), we find a tuple $\mathbf{u}^{[i,k]} \in D^{i-1}$ and a tuple $\mathbf{v}^{[i,k,b]} \in D^{n-i}$ for each $b \in \mathcal{E}_{i,k}$ such that

   $$\left(\mathbf{u}^{[i,k]}, b, \mathbf{v}^{[i,k,b]}\right) \in R, \qquad \text{for all } b \in \mathcal{E}_{i,k}.$$

   Then we set

   $$t_i(a) = \frac{M(\mathbf{u}^{[i,k]}, a)}{\sum_{b \in \mathcal{E}_{i,k}} M(\mathbf{u}^{[i,k]}, b)}. \tag{7}$$

   By (6), $t_i(a)$ can be computed efficiently using tuples $\mathbf{u}^{[i,k]}$ and $\mathbf{v}^{[i,k,b]}$, for $b \in \mathcal{E}_{i,k}$.

This finishes the construction of $t_i(\cdot)$.

Finally we prove (4). Let $\mathbf{u}$ be any tuple in R and $\mathcal{E}_{i,k}$ be the equivalence class of $\sim_i$ that u_i belongs to. Then

$$\sum_{x_i,\ldots,x_n \in D} F(u_1, \ldots, u_{i-1}, x_i, \ldots, x_n) = \sum_{b \in \mathcal{E}_{i,k}} \sum_{x_{i+1},\ldots,x_n \in D} F(u_1, \ldots, u_{i-1}, b, x_{i+1}, \ldots, x_n).$$

Let $\mathbf{u}^*$ denote the $(i-1)$-tuple $(u_1, \ldots, u_{i-1})$. Then by the definition of $\mathbf{M}$, we can rewrite the sum as

$$\sum_{x_i,\ldots,x_n \in D} F(u_1, \ldots, u_{i-1}, x_i, \ldots, x_n) = \sum_{b \in \mathcal{E}_{i,k}} M(\mathbf{u}^*, b).$$

Recall the tuples $\mathbf{u}^{[i,k]}$ and $\mathbf{v}^{[i,k,b]}$, $b \in \mathcal{E}_{i,k}$, which we used in the construction of $t_i(\cdot)$. Because $\mathbf{M}$ is block-rank-1 and because $\mathbf{u}^*$ and $\mathbf{u}^{[i,k]}$ are known to belong to the same block of $\mathbf{M}$, we have

$$\sum_{b \in \mathcal{E}_{i,k}} M(\mathbf{u}^*, b) = \sum_{b \in \mathcal{E}_{i,k}} \frac{M(\mathbf{u}^*, u_i)}{M(\mathbf{u}^{[i,k]}, u_i)} \cdot M(\mathbf{u}^{[i,k]}, b) = \frac{M(\mathbf{u}^*, u_i)}{M(\mathbf{u}^{[i,k]}, u_i)} \cdot \sum_{b \in \mathcal{E}_{i,k}} M(\mathbf{u}^{[i,k]}, b).$$

However, by the definition (7) of $t_i(\cdot)$, we have

$$\sum_{b \in \mathcal{E}_{i,k}} M(\mathbf{u}^{[i,k]}, b) = \frac{M(\mathbf{u}^{[i,k]}, u_i)}{t_i(u_i)},$$

since we assumed that $u_i \in \mathcal{E}_{i,k}$. As a result, we have

$$\sum_{x_i,\ldots,x_n \in D} F(u_1, \ldots, u_{i-1}, x_i, \ldots, x_n) = \sum_{b \in \mathcal{E}_{i,k}} M(\mathbf{u}^*, b) = \frac{M(\mathbf{u}^*, u_i)}{t_i(u_i)} = \left(\prod_{j \in [i-1]} s_j(u_j)\right) \left(\prod_{j \in [i:n]} \frac{s_j(u_j)}{t_j(u_j)}\right).$$

The last equation follows from (6). This finishes the construction of $t_n, \ldots, t_2$ and the proof of Lemma 6.

6 Decidability of the Dichotomy

In this section, we prove Theorem 5 by showing that the decision problem is in NP.

By Theorem 4 we need to decide, given D and $\mathcal{F}$, whether Γ is strongly rectangular and $\mathcal{F}$ is weakly balanced or not. The first part can be done in NP [4, 20] by exhaustively searching for a Mal'tsev polymorphism.

Lemma 11 ([4, 20])**.** *Given* Γ*, deciding whether it is strongly rectangular is in* NP.

6.1 Primitive Balance

Next we show the notion of weak balance is equivalent to the following even *weaker* notion of *primitive balance*:

Definition 5 (Primitive Balance)**.** *We say* $\mathcal{F}$ *is* primitively balanced *if for any instance* I *of* $(D, \mathcal{F})$ *and the* n*-ary function* $F_I(x_1, \ldots, x_n)$ *it defines, the following* $d \times d$ *matrix* $\mathbf{M}_I$ *is block-rank-*1*: The rows of* $\mathbf{M}_I$ *are indexed by* $x_1 \in D$ *and the columns are indexed by* $x_2 \in D$*, and*

$$M_I(x_1, x_2) = \sum_{x_3,\ldots,x_n \in D} F_I(x_1, x_2, x_3, \ldots, x_n), \quad \textit{for all } x_1, x_2 \in D. \tag{8}$$

It is clear that weak balance implies primitive balance. The following lemma proves the inverse direction:

Lemma 12. *If* Γ *is primitively balanced , then it is also weakly balanced.*

Proof. Assume for a contradiction that Γ is not weakly balanced. By definition, this means there exist an I over n-variables and an integer $a : 1 \leq a < n$ such that the following $d^a \times d$ matrix $\mathbf{M}$ is not block-rank-1: the rows of $\mathbf{M}$ are indexed by $\mathbf{u} \in D^a$ and the columns are indexed by $v \in D$, and

$$M(\mathbf{u}, v) = \sum_{\mathbf{w} \in D^{n-a-1}} F_I(\mathbf{u}, v, \mathbf{w}), \quad \text{for all } \mathbf{u} \in D^a \text{ and } v \in D.$$

As a result, we know by Lemma 2 that $\mathbf{A} = \mathbf{M}^{\mathrm{T}}\mathbf{M}$ is not block-rank-1.

To reach a contradiction, we construct I' from I as follows: I' has $2n - a$ variables in the following order:

$$x_1, x_2, y_1, \ldots, y_a, z_1, \ldots, z_{n-a-1}, w_1, \ldots, w_{n-a-1}.$$

The instance I' consists of two parts: a copy of I over $(y_1, \ldots, y_a, x_1, z_1, \ldots, z_{n-a-1})$ and a copy of I over $(y_1, \ldots, y_a, x_2, w_1, \ldots, w_{n-a-1})$. Let $F_{I'}$ denote the function that I' defines. It gives us the following $d \times d$ matrix $\mathbf{M}_{I'}$:

$$M_{I'}(x_1, x_2) = \sum_{\mathbf{y} \in D^a, \mathbf{z}, \mathbf{w} \in D^{n-a-1}} F_I(\mathbf{y}, x_1, \mathbf{z}) \cdot F_I(\mathbf{y}, x_2, \mathbf{w}) = \sum_{\mathbf{y} \in D^a} M(\mathbf{y}, x_1) \cdot M(\mathbf{y}, x_2) = A(x_1, x_2),$$

which we know is not block-rank-1. This contradicts with the assumption that $\mathcal{F}$ is primitively balanced . $\square$

Now the decision problem reduces to the following, and we call it

> PRIMITIVE BALANCE: Given D and $\mathcal{F}$ such that Γ is strongly rectangular (which by Lemma 11 can be verified in NP), decide whether $\mathcal{F}$ is primitively balanced.

Since Γ is strongly rectangular, we know that for any input I of $(D,\mathcal{F})$, the $d\times d$ matrix $\mathbf{M}_I$ defined in (8) must be rectangular. We need the following useful lemma from [20], which gives us a simple way to check whether a rectangular matrix is block-rank-1 or not.

Lemma 13 ([20])**.** *A rectangular $d\times d$ matrix $\mathbf{M}$ is block-rank-1* if and only if

$$M(\alpha,\kappa)^2M(\beta,\lambda)^2M(\alpha,\lambda)M(\beta,\kappa)=M(\alpha,\lambda)^2M(\beta,\kappa)^2M(\alpha,\kappa)M(\beta,\lambda) \tag{9}$$

for all $\alpha\neq\beta\in D$ and $\kappa\neq\lambda\in D$.

As a result, for PRIMITIVE BALANCE it suffices to check whether (9) holds for $\mathbf{M}_I$, for all instances I and for all $\alpha\neq\beta,\kappa\neq\lambda\in D$. In the rest of this section, we fix $\alpha\neq\beta\in D$ and $\kappa\neq\lambda\in D$, and show that the decision problem (that is, whether (9) holds for all I) is in NP. Theorem 5 then follows immediately since there are only polynomially many possible tuples $(\alpha,\beta,\kappa,\lambda)$ to check.

6.2 Reformulation of the Decision Problem

Fixing $\alpha\neq\beta\in D$ and $\kappa\neq\lambda\in D$, we follow [20] and reformulate the decision problem using a new pair $(\mathfrak{D},\mathfrak{F})$, that is, the *6-th power* of $(D,\mathcal{F})$:

1. First, the new domain $\mathfrak{D}=D^6$, and we use $\mathfrak{s}=(s_1,\ldots,s_6)$ to denote an element in $\mathfrak{D}$, where $s_i\in D$.
2. Second, $\mathfrak{F}=\{g_1,\ldots,g_h\}$ has the same number of functions as $\mathcal{F}$ and every g_i, $i\in[h]$, has the same arity r_i as f_i. Function $g_i:\mathfrak{D}^{r_i}\rightarrow\mathbb{R}_+$ is constructed explicitly from f_i as follows:

$$g_i(\mathfrak{s}_1,\ldots,\mathfrak{s}_{r_i})=\prod_{j\in[6]}f_i(s_{1,j},\ldots,s_{r_i,j}),\qquad\text{for all }\mathfrak{s}_1,\ldots,\mathfrak{s}_{r_i}\in\mathfrak{D}=D^6.$$

In the rest of the section, we will always use x_i to denote variables over D and y_i,z_i to denote variables over $\mathfrak{D}$.

Given any input instance I of $(D,\mathcal{F})$ over n variables $(x_1,\ldots,x_n)$, it naturally defines an input instance $\mathfrak{I}$ of $(\mathfrak{D},\mathfrak{F})$ over n variables $(y_1,\ldots,y_n)$ as follows: for each tuple $(f,i_1,\ldots,i_r)\in I$, add a tuple $(g,i_1,\ldots,i_r)$ to $\mathfrak{I}$, where $g\in\mathfrak{F}$ corresponds to $f\in\mathcal{F}$. Moreover, this is clearly a bijection between the set of all I and the set of all $\mathfrak{I}$. Similarly, we let $G:\mathfrak{D}^n\rightarrow\mathbb{R}_+$ denote the n-ary function that $\mathfrak{I}$ defines:

$$G(y_1,\ldots,y_n)=\prod_{(g,i_1,\ldots,i_r)\in\mathfrak{I}}g(y_{i_1},\ldots,y_{i_r}),\qquad\text{for all }y_1,\ldots,y_n\in\mathfrak{D}.$$

The reason why we introduce the new tuple $(\mathfrak{D},\mathfrak{F})$ is because it gives us a new and much simpler formulation of the decision problem we are interested.

To see this, we let $\mathfrak{a},\mathfrak{b},\mathfrak{c}$ denote the following three specific elements from $\mathfrak{D}$:

$$\mathfrak{a}=(\alpha,\alpha,\alpha,\beta,\beta,\beta),\quad\mathfrak{b}=(\kappa,\kappa,\lambda,\lambda,\lambda,\kappa),\quad\mathfrak{c}=(\lambda,\lambda,\kappa,\kappa,\kappa,\lambda).$$

Since $\alpha\neq\beta$ and $\kappa\neq\lambda$, $\mathfrak{a},\mathfrak{b},\mathfrak{c}$ are three distinct elements in $\mathfrak{D}$. We adopt the notation of [20]. For each $\mathfrak{s}\in\mathfrak{D}$, let

$$\text{hom}_{\mathfrak{s}}(\mathfrak{I})\overset{\text{def}}{=}\sum_{y_3,\ldots,y_n\in\mathfrak{D}}G(\mathfrak{a},\mathfrak{s},y_3,\ldots,y_n),\qquad\text{for every instance }\mathfrak{I}\text{ of }(\mathfrak{D},\mathfrak{F}).$$

It is easy to prove the following two equations. Let $\mathfrak{I}$ be the instance of $(\mathfrak{D}, \mathfrak{F})$ that corresponds to I, and $\mathbf{M}_I$ be the $d \times d$ matrix as defined in (8). Then

$$\begin{aligned} \hom_{\mathfrak{b}}(\mathfrak{I}) &= M_I(\alpha,\kappa)^2 M_I(\beta,\lambda)^2 M_I(\alpha,\lambda) M_I(\beta,\kappa) \quad \text{and} \\ \hom_{\mathfrak{c}}(\mathfrak{I}) &= M_I(\alpha,\lambda)^2 M_I(\beta,\kappa)^2 M_I(\alpha,\kappa) M_I(\beta,\lambda) \end{aligned}$$

As a result, we have the following reformulation of the decision problem:

$$\mathbf{M}_I \text{ satisfies (9) for all } I \iff \hom_{\mathfrak{b}}(\mathfrak{I}) = \hom_{\mathfrak{c}}(\mathfrak{I}) \text{ for all } \mathfrak{I}$$

The next reformulation considers sums over *injective* tuples only. We say $(y_1, \ldots, y_n) \in \mathfrak{D}^n$ is an injective tuple if $y_i \neq y_j$ for all $i \neq j \in [n]$ (or equivalently, if we view $(y_1, \ldots, y_n)$ as a map from $[n]$ to $\mathfrak{D}$, it is injective). We use Y_n to denote the set of injective n-tuples. (Clearly this definition is only useful when $n \leq |\mathfrak{D}|$, otherwise Y_n is empty.) We now define functions $\text{mon}_{\mathfrak{s}}(\mathfrak{I})$, which are sums over injective tuples: For each $\mathfrak{s} \in \mathfrak{D}$, let

$$\text{mon}_{\mathfrak{s}}(\mathfrak{I}) \overset{\text{def}}{=} \sum_{(\mathfrak{a},\mathfrak{s},y_3,\ldots,y_n)\in Y_n} G(\mathfrak{a},\mathfrak{s},y_3,\ldots,y_n), \qquad \text{for every instance } \mathfrak{I} \text{ of } (\mathfrak{D},\mathfrak{F}).$$

The following lemma shows that $\hom_{\mathfrak{b}}(\mathfrak{I}) = \hom_{\mathfrak{c}}(\mathfrak{I})$ for all $\mathfrak{I}$ if and only if the same equation holds for the sums over injective tuples. The proof is exactly the same as Lemma 41 in [20], using the Mobius inversion. So we skip it here.

Lemma 14 ([20], Lemma 41)**.** $\hom_{\mathfrak{b}}(\mathfrak{I}) = \hom_{\mathfrak{c}}(\mathfrak{I})$ *for all* $\mathfrak{I}$ *if and only if* $\text{mon}_{\mathfrak{b}}(\mathfrak{I}) = \text{mon}_{\mathfrak{c}}(\mathfrak{I})$ *for all* $\mathfrak{I}$.

Finally, the following reformulation gives us a condition that can be checked in NP:

Lemma 15. $\text{mon}_{\mathfrak{b}}(\mathfrak{I}) = \text{mon}_{\mathfrak{c}}(\mathfrak{I})$ *for all* $\mathfrak{I}$ *if, and only if, there exists a bijection* π *from the domain* $\mathfrak{D}$ *to itself* (*which we will refer to as an* automorphism *from* $(\mathfrak{D},\mathfrak{F})$ *to itself*) *such that* $\pi(\mathfrak{a}) = \pi(\mathfrak{a})$, $\pi(\mathfrak{b}) = \pi(\mathfrak{c})$, *and for every* r*-ary function* $g \in \mathfrak{F}$, *we have*

$$g(y_1,\ldots,y_r) = g\Big(\pi(y_1),\ldots,\pi(y_r)\Big), \qquad \textit{for all } y_1,\ldots,y_r \in \mathfrak{D}. \tag{10}$$

Proof. We start with the easier direction: If π exists, then $\text{mon}_{\mathfrak{b}}(\mathfrak{I}) = \text{mon}_{\mathfrak{c}}(\mathfrak{I})$ for all $\mathfrak{I}$. This is because for any injective n-tuple $(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n) \in Y_n$, we can apply π and get a new injective n-tuple $(\mathfrak{a},\mathfrak{c},\pi(y_3),\ldots,\pi(y_n)) \in Y_n$ and this is a bijection from $(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n) \in Y_n$ and $(\mathfrak{a},\mathfrak{c},z_3,\ldots,z_n) \in Y_n$. Moreover, by (10) we have

$$G(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n) = G\big(\mathfrak{a},\mathfrak{c},\pi(y_3),\ldots,\pi(y_n)\big).$$

As a result, the two sums $\text{mon}_{\mathfrak{b}}(\mathfrak{I})$ and $\text{mon}_{\mathfrak{c}}(\mathfrak{I})$ over injective tuples must be equal.

The other direction is more difficult. First, we prove that if $\text{mon}_{\mathfrak{b}}(\mathfrak{I}) = \text{mon}_{\mathfrak{c}}(\mathfrak{I})$ for all $\mathfrak{I}$, then for any $\mathfrak{I}$ and any tuple $(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n) \in Y_n$ with $G(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n) > 0$, there exists a $(\mathfrak{a},\mathfrak{c},z_3,\ldots,z_n) \in Y_n$ such that

$$G(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n) = G(\mathfrak{a},\mathfrak{c},z_3,\ldots,z_n). \tag{11}$$

To prove this we look at the following sequence of instances $\mathfrak{I}_1 = \mathfrak{I}, \mathfrak{I}_2, \ldots$ defined from $\mathfrak{I}$, where $\mathfrak{I}_j$ consists of exactly j copies of $\mathfrak{I}$ over the same set of variables. We use G_j to denote the n-ary function that $\mathfrak{I}_j$ defines, then

$$G_j(y_1,\ldots,y_n)=\big(G(y_1,\ldots,y_n)\big)^j,\quad \text{for all } y_1,\ldots,y_n\in\mathfrak{D}.$$

Let $Q=\{q_1,\ldots,q_{|Q|}\}$ denote the set of all possible positive values of G over Y_n; let $k_i\ge 0$ denote the number of tuples $(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n)\in Y_n$ such that $G(\mathfrak{a},\mathfrak{b},y_3,\ldots,y_n)=q_i$, $i\in[|Q|]$; and let $\ell_i\ge 0$ denote the number of tuples $(\mathfrak{a},\mathfrak{c},y_3,\ldots,y_n)\in Y_n$ such that $G(\mathfrak{a},\mathfrak{c},y_3,\ldots,y_n)=q_i$, $i\in[|Q|]$. Then by $\text{mon}_\mathfrak{b}(\mathfrak{I}_j)=\text{mon}_\mathfrak{c}(\mathfrak{I}_j)$,

$$\sum_{i\in[|Q|]}k_i\cdot(q_i)^j=\sum_{i\in[|Q|]}\ell_i\cdot(q_i)^j,\quad \text{for all } j\ge 1.$$

Viewing $k_i-\ell_i$ as variables, the above equation gives us a linear system with a Vandermonde matrix if we let j go from 1 to $|Q|$. As a result, we must have $k_i=\ell_i$ for all $i\in[|Q|]$, and (11) follows.

To finish the proof, we need the following technical lemma:

Lemma 16. *Let Q be a finite and nonempty set of positive numbers. Then for any $k\ge 1$, there exists a sequence of positive integers $N_1,\ldots,N_k$ such that*

$$q_1^{N_1}q_2^{N_2}\cdots q_k^{N_k}=(q_1')^{N_1}(q_2')^{N_2}\cdots(q_k')^{N_k},\quad \textit{where } q_1,\ldots,q_k,q_1'\ldots,q_k'\in Q \tag{12}$$

if and only if $q_i=q_i'$ for every $i\in[k]$.

Proof. The lemma is trivial if $|Q|=1$, so we assume $|Q|\ge 2$. We use induction on k. The basis is trivial: we just set $N_1=1$. Now assume the lemma holds for some $k\ge 1$, and $N_1,\ldots,N_k$ is the sequence for k. We show how to find N_{k+1} so that $N_1,\ldots,N_{k+1}$ satisfies the lemma for $k+1$. To this end, we let

$$c_{\text{min}}=\min_{q>q'\in Q}q/q'>1\qquad\text{and}\qquad c_{\text{max}}=\max_{q>q'\in Q}q/q'.$$

Then we let N_{k+1} be a large enough integer such that

$$\big(c_{\text{min}}\big)^{N_{k+1}}>\big(c_{\text{max}}\big)^{\sum_{i\in[k]}N_i}.$$

To prove the correctness, we assume (12) holds. First, we must have $q_{k+1}=q_{k+1}'$. Otherwise, assume without generality that $q_{k+1}>q_{k+1}'$, then by (12)

$$\big(c_{\text{min}}\big)^{N_{k+1}}\le\big(q_{k+1}/q_{k+1}'\big)^{N_{k+1}}=\big(q_1'/q_1\big)^{N_1}\cdots\big(q_k'/q_k\big)^{N_k}\le\big(c_{\text{max}}\big)^{\sum_{i\in[k]}N_k},$$

which contradicts with the definition of N_{k+1}. Once we have $q_{k+1}=q_{k+1}'$, they can be removed from (12) and by the inductive hypothesis, we have $q_i=q_i'$ for all $i\in[k]$. This finishes the induction, and the lemma is proved. ☐

To find π, we define the following $\mathfrak{I}$. It has $|\mathfrak{D}|$ variables and we denote them by $y_\mathfrak{s}$, $\mathfrak{s}\in\mathfrak{D}$. (In particular, $y_\mathfrak{a}$ and $y_\mathfrak{b}$ are the first and second variables of $\mathfrak{I}$ so that later $\text{mon}_\mathfrak{s}(\mathfrak{I})$ is well-defined.) Let L be the set of all tuples $(g,\mathfrak{s}_1,\ldots,\mathfrak{s}_r)$, where g is an r-ary function in $\mathfrak{F}$ and $g(\mathfrak{s}_1,\ldots,\mathfrak{s}_r)>0$. We let $N_1,\ldots,N_{|L|}$ be the sequence of positive integers that satisfies Lemma 16 with $k=|L|$ and

$$Q=\Big\{g(\mathfrak{s}_1,\ldots,\mathfrak{s}_r):(g,\mathfrak{s}_1,\ldots,\mathfrak{s}_r)\in L\Big\}.$$

Then we enumerate all tuples in L in any order. For the ith tuple $(g, \mathfrak{s}_1, \ldots, \mathfrak{s}_r) \in L$, $i \in [|L|]$, we add N_i copies of the same tuple $(g, \mathfrak{s}_1, \ldots, \mathfrak{s}_r)$ to $\mathfrak{I}$. This finishes the definition of $\mathfrak{I}$.

From the definition of $\mathfrak{I}$, it is easy to see that $G(y_\mathfrak{s} : y_\mathfrak{s} = \mathfrak{s} \text{ for all } \mathfrak{s} \in \mathfrak{D}) > 0$. Therefore, by (11) we know there exists a tuple $(z_\mathfrak{s} : \mathfrak{s} \in \mathfrak{D}) \in Y_n$ such that $z_\mathfrak{a} = \mathfrak{a}$, $z_\mathfrak{b} = \mathfrak{c}$, and

$$G\big(y_\mathfrak{s} : y_\mathfrak{s} = \mathfrak{s} \text{ for all } \mathfrak{s} \in \mathfrak{D}\big) = G\big(z_\mathfrak{s} : \mathfrak{s} \in \mathfrak{D}\big) > 0.$$

We show that $\pi(\mathfrak{s}) \overset{\text{def}}{=} z_\mathfrak{s}$, for every $\mathfrak{s} \in \mathfrak{D}$, is the bijection that we are looking for.

First, using Lemma 16, it follows from the definition of $\mathfrak{I}$ that for every tuple $(g, \mathfrak{s}_1, \ldots, \mathfrak{s}_r) \in L$, we have

$$g(\mathfrak{s}_1, \ldots, \mathfrak{s}_r) = g\big(\pi(\mathfrak{s}_1), \ldots, \pi(\mathfrak{s}_r)\big).$$

So we only need to show that $g(\pi(\mathfrak{s}_1), \ldots, \pi(\mathfrak{s}_r)) = 0$ whenever $g(\mathfrak{s}_1, \ldots, \mathfrak{s}_r) = 0$. This follows directly from the fact that π is a bijection and thus, $(\mathfrak{s}_1, \ldots, \mathfrak{s}_r) \to (\pi(\mathfrak{s}_1), \ldots, \pi(\mathfrak{s}_r))$ is also a bijection. $\square$

With Lemma 14 and Lemma 15, we only need to check whether there exists an automorphism π from $(\mathfrak{D}, \mathfrak{F})$ to itself such that $\pi(\mathfrak{a}) = \mathfrak{a}$ and $\pi(\mathfrak{b}) = \mathfrak{c}$. We can just exhaustively check all possible bijections from $\mathfrak{D}$ to itself, and this gives us an algorithm in NP.

7 Proof of Lemma 3

Let I be an input of (D, Γ) with n variables $\mathbf{x} = (x_1, \ldots, x_n)$ and m tuples, and R be the relation it defines.

For each $k \geq 1$, we let I_k denote the following input of $(D, \mathcal{F})$: I_k has n variables $(x_1, \ldots, x_n)$; and for each $(\Theta, i_1, \ldots, i_r) \in I$, we add k copies of $(f, i_1, \ldots, i_r)$ to I_k, where $f \in \mathcal{F}$ is the r-ary function that corresponds to $\Theta \in \Gamma$. We use $F_k(\mathbf{x})$ to denote the n-ary non-negative function that I_k defines. Then it is clear that

$$F_k(\mathbf{x}) = \Big(F_1(\mathbf{x})\Big)^k, \qquad \text{for all } \mathbf{x} \in D^n. \tag{13}$$

We will show that to compute $|R|$, one only needs to evaluate $Z(I_k)$ for k from 1 to some polynomial of m. This gives us a polynomial-time reduction from (D, Γ) to $(D, \mathcal{F})$.

Now we let Q_m denote the set of all integer tuples

$$\mathbf{q} = \Big(q_{i,\mathbf{t}} \geq 0 : i \in [h] \text{ and } \mathbf{t} \in D^{r_i} \text{ such that } f_i(\mathbf{t}) > 0\Big)$$

that sum to m. And let VALUE_m denote the following set of positive numbers:

$$\text{VALUE}_m = \left\{ \prod_{i \in [h],\, \mathbf{t} \in D^{r_i}} \Big(f_i(\mathbf{t})\Big)^{q_{i,\mathbf{t}}} : \mathbf{q} \in Q_m \right\}.$$

It is easy to show that both $|Q_m|$ and $|\text{VALUE}_m|$ are polynomial in m (as d, h and r_i, $i \in [h]$ are all constants) and can be computed in polynomial time in m. Moreover, by the definition of VALUE_m we have for every $\mathbf{x} \in D^n$:

$$F_1(\mathbf{x}) > 0 \implies F_1(\mathbf{x}) \in \text{VALUE}_m.$$

For every $c \in \text{VALUE}_m$, we let N_c denote the number of $\mathbf{x} \in D^n$ such that $F_1(\mathbf{x}) = c$. Then we have

$$Z(I_1) = \sum_{c \in \text{VALUE}_m} N_c \cdot c \tag{14}$$

We also have

$$\big|R\big| = \sum_{c \in \text{VALUE}_m} N_c \tag{15}$$

and by (13)

$$Z(I_k) = \sum_{c \in \text{VALUE}_m} N_c \cdot c^k, \quad \text{for every } k \geq 1. \tag{16}$$

If we view $\{N_c : c \in \text{VALUE}_m\}$ as variables, then by taking $k = 1, \ldots, |\text{VALUE}_m|$, (16) gives us a Vandermonde system from which we can compute N_c, $c \in \text{VALUE}_m$, in polynomial time. We can then use (15) to compute $|R|$.

This finishes the proof of Lemma 3.

8 Proof of Lemma 5

Assume that $\mathcal{F}$ is not balanced. Then by definition, there exists an input instance I for $(D, \mathcal{F})$ such that

1. It defines an n-ary function $F(x_1, \ldots, x_n)$; and
2. There exist integers $a, b : 1 \leq a < b \leq n$ such that the following $d^a \times d^{b-a}$ matrix $\mathbf{M}$ is not block-rank-1: the rows are indexed by $\mathbf{u} \in D^a$ and the columns are indexed by $\mathbf{v} \in D^{b-a}$, and

$$M(\mathbf{u}, \mathbf{v}) = \sum_{\mathbf{w} \in D^{n-b}} F(\mathbf{u}, \mathbf{v}, \mathbf{w}), \quad \text{for all } \mathbf{u} \in D^a \text{ and } \mathbf{v} \in D^{b-a}.$$

Because $\mathbf{M}$ is not block-rank-1, by Lemma 2, it has two rows that are neither linearly dependent nor orthogonal. We let $\mathbf{M}(\mathbf{u}_1, *)$ and $\mathbf{M}(\mathbf{u}_2, *)$ be such two rows, where $\mathbf{u}_1, \mathbf{u}_2 \in D^a$. Then

$$0 < \big\langle \mathbf{M}(\mathbf{u}_1, *), \mathbf{M}(\mathbf{u}_2, *) \big\rangle^2 < \big\langle \mathbf{M}(\mathbf{u}_1, *), \mathbf{M}(\mathbf{u}_1, *) \big\rangle \cdot \big\langle \mathbf{M}(\mathbf{u}_2, *), \mathbf{M}(\mathbf{u}_2, *) \big\rangle. \tag{17}$$

We let $\mathbf{A} = \mathbf{M}\mathbf{M}^{\mathrm{T}}$, which is clearly a symmetric and non-negative $d^a \times d^a$ matrix, with both of its rows and columns indexed by $\mathbf{u} \in D^a$. It then immediately follows from (17) that $\mathbf{A}$ is not block-rank-1, since all the four entries in the $\{\mathbf{u}_1, \mathbf{u}_2\} \times \{\mathbf{u}_1, \mathbf{u}_2\}$ sub-matrix of $\mathbf{A}$ are positive but this 2×2 sub-matrix is of rank 2 by (17).

To finish the proof, we give a polynomial-time reduction from $Z_{\mathbf{A}}(\cdot)$ to $(D, \mathcal{F})$. Because the former is #P-hard by Theorem 2 (since $\mathbf{A}$ is not block-rank-1), we know that $(D, \mathcal{F})$ is also #P-hard.

Let $G = (V, E)$ be an input undirected graph of $Z_{\mathbf{A}}(\cdot)$. We construct an input instance I_G of $(D, \mathcal{F})$ from G, using I (which is considered as a constant here since it does not depend on G), as follows.

1. For every vertex $v \in V$, we create a variables over D, denoted by $x_{v,1}, \ldots, x_{v,a}$; and
2. For every edge $e = vv' \in E$, we add $(b - a) + 2(n - b)$ variables over D, denoted by

$$y_{e,a+1}, \ldots, y_{e,b}, z_{e,b+1}, \ldots, z_{e,n}, z'_{e,b+1}, \ldots, z'_{e,n}.$$

Then we make a copy of I over the following n variables:

$$\big(x_{v,1},\ldots,x_{v,a},y_{e,a+1},\ldots,y_{e,b},z_{e,b+1},\ldots,z_{e,n}\big)$$

as well as the following n variables:

$$\big(x_{v',1},\ldots,x_{v',a},y_{e,a+1},\ldots,y_{e,b},z'_{e,b+1},\ldots,z'_{e,n}\big).$$

This finishes the construction of I_G.

It is easy to show by the definitions of $\mathbf{M}$ and $\mathbf{A}$ above that $Z_{\mathbf{A}}(G)=Z(I_G)$. This gives us a polynomial-time reduction from problems $Z_{\mathbf{A}}(\cdot)$ to $(D,\mathcal{F})$ since I_G can be constructed from G in polynomial time.

9 Equivalence of Balance and Strong Balance

In [21] Dyer and Richerby used the following notion of strong balance for unweighted constraint languages Γ and showed that (D,Γ) is in polynomial time if Γ is strongly balanced; and is #P-hard otherwise.

Definition 6. *Let Γ be an unweighted constraint language over D. We call Γ* strongly balanced *if for every input instance I of (D,Γ) (which defines an n-ary relation R) and for any $a,b,c: 1\le a<b\le c\le n$, the following $d^a\times d^{b-a}$ matrix $\mathbf{M}$ is block-rank-1: the rows are indexed by $\mathbf{u}\in D^a$ and the columns are indexed by $\mathbf{v}\in D^{b-a}$,*

$$M(\mathbf{u},\mathbf{v})=\Big|\big\{\mathbf{w}\in D^{c-b}:\exists\,\mathbf{z}\in D^{n-c}\text{ such that }(\mathbf{u},\mathbf{v},\mathbf{w},\mathbf{z})\in R\big\}\Big|,\quad\text{for all }\mathbf{u}\in D^a\text{ and }\mathbf{v}\in D^{b-a}.\tag{18}$$

There are two special cases. When $c=b$, $M(\mathbf{u},\mathbf{v})$ is 1 if there exists a $\mathbf{z}\in D^{n-c}$ such that $(\mathbf{u},\mathbf{v},\mathbf{z})\in R$; and is 0 otherwise. When $n=c$, $M(\mathbf{u},\mathbf{v})$ is the number of $\mathbf{w}\in D^{c-b}$ such that $(\mathbf{u},\mathbf{v},\mathbf{w})\in R$.

Theorem 6. *(D,Γ) is in polynomial time if Γ is strongly balanced; and is #P-hard otherwise.*

Notably the difference between the notion of balance we used for weighted languages $\mathcal{F}$ (Definition 4) and the one above for unweighted languages Γ [21] is that we do not allow the use of existential quantifiers in the former. One can similarly define the following notion of balance for unweighted Γ:

Definition 7. *Let Γ be an unweighted constraint language over D. We call Γ* balanced *if for every instance I of (D,Γ) (which defines an n-ary relation R) and for any $a,b: 1\le a<b\le n$, the following $d^a\times d^{b-a}$ matrix $\mathbf{M}$ is block-rank-1: the rows are indexed by $\mathbf{u}\in D^a$ and the columns are indexed by $\mathbf{v}\in D^{b-a}$,*

$$M(\mathbf{u},\mathbf{v})=\Big|\big\{\mathbf{w}\in D^{n-b}:(\mathbf{u},\mathbf{v},\mathbf{w})\in R\big\}\Big|,\quad\text{for all }\mathbf{u}\in D^a\text{ and }\mathbf{v}\in D^{b-a}.\tag{19}$$

We show below that these two notions, strong balance and balance, are equivalent.

Lemma 17 (Equivalence of Balance and Strong Balance)**.** *If Γ is balanced, then it is also strongly balanced.*

Proof. We assume that Γ is balanced. Let I be any instance of (D,Γ) which defines an n-ary relation R. Let a,b and c be integers such that $1\le a<b\le c\le n$. It suffices to show that the matrix $\mathbf{M}$ in (18) is block-rank-1.

For this purpose, we define a new input instance I_k of (D, Γ) for each $k \geq 1$:

1. First, I_k has $c + k(n-c)$ variables in the following order:

$$x_1, \ldots, x_c, y_{1,c+1}, \ldots, y_{1,n}, \ldots, y_{k,c+1}, \ldots, y_{k,n}.$$

   Below we let $\mathbf{y}_i$, $i \in [k]$, denote $(y_{i,c+1}, \ldots, y_{i,n})$ for convenience.

2. For each $i \in [k]$, we add a copy of I on the following n variables of I_k: $x_1, \ldots, x_c, y_{i,c+1}, \ldots, y_{i,n}$.

It is clear that I_1 is exactly I. We also use R_k to denote the relation that I_k defines, $k \geq 1$.

Because Γ is balanced, the following $d^a \times d^{b-a}$ matrix $\mathbf{M}^{[k]}$ is block-rank-1: For $\mathbf{u} \in D^a$ and $\mathbf{v} \in D^{b-a}$,

$$M^{[k]}(\mathbf{u}, \mathbf{v}) = \left|\Big\{(\mathbf{w}, \mathbf{y}_1, \ldots, \mathbf{y}_k) : \mathbf{w} \in D^{c-b}, \mathbf{y}_1, \ldots, \mathbf{y}_k \in D^{n-c} \text{ and } (\mathbf{u}, \mathbf{v}, \mathbf{w}, \mathbf{y}_1, \ldots, \mathbf{y}_k) \in R_k\Big\}\right|.$$

From the definition of I_k, we have $M(\mathbf{u}, \mathbf{v}) > 0$ if and only if $M^{[k]}(\mathbf{u}, \mathbf{v}) > 0$, for all $\mathbf{u} \in D^a$ and $\mathbf{v} \in D^{b-a}$.

Therefore, there exist pairwise disjoint and nonempty subsets of D^a, denoted $A_1, \ldots, A_s$, and pairwise disjoint and nonempty subsets of D^{b-a}, denoted $B_1, \ldots, B_s$, for some $s \geq 0$, such that

$$M(\mathbf{u}, \mathbf{v}) > 0 \iff M^{[k]}(\mathbf{u}, \mathbf{v}) > 0 \iff \mathbf{u} \in A_\ell \text{ and } \mathbf{v} \in B_\ell \text{ for some } \ell \in [s].$$

Now to prove that $\mathbf{M}$ is block-rank-1, we only need to show that for every $\ell \in [s]$,

$$M(\mathbf{u}_1, \mathbf{v}_1) \cdot M(\mathbf{u}_2, \mathbf{v}_2) = M(\mathbf{u}_1, \mathbf{v}_2) \cdot M(\mathbf{u}_2, \mathbf{v}_1), \quad \text{for all } \mathbf{u}_1, \mathbf{u}_2 \in A_\ell \text{ and } \mathbf{v}_1, \mathbf{v}_2 \in B_\ell. \tag{20}$$

To prove (20), we let

$$W_{i,j} = \Big\{\mathbf{w} \in D^{c-b} : \exists\, \mathbf{y} \in D^{n-c} \text{ such that } (\mathbf{u}_i, \mathbf{v}_j, \mathbf{w}, \mathbf{y}) \in R\Big\}, \quad \text{for } i, j \in \{1, 2\}.$$

Furthermore, for every $\mathbf{w} \in W_{i,j}$, we let $Y_{i,j,\mathbf{w}}$ denote the (nonempty) set of $\mathbf{y} \in D^{n-c}$ such that $(\mathbf{u}_i, \mathbf{v}_j, \mathbf{w}, \mathbf{y}) \in R$. Now using $W_{i,j}$ and $Y_{i,j,\mathbf{w}}$, it follows from the definition of I_k that

$$M^{[k]}(\mathbf{u}_i, \mathbf{v}_j) = \sum_{\mathbf{w} \in W_{i,j}} \Big|Y_{i,j,\mathbf{w}}\Big|^k.$$

Because $\mathbf{M}^{[k]}$ is block-rank-1, we have the following equation for every $k \geq 1$:

$$\sum_{\mathbf{w} \in W_{1,1}, \mathbf{w}' \in W_{2,2}} \Big(\big|Y_{1,1,\mathbf{w}}\big| \cdot \big|Y_{2,2,\mathbf{w}'}\big|\Big)^k = \sum_{\mathbf{w} \in W_{1,2}, \mathbf{w}' \in W_{2,1}} \Big(\big|Y_{1,2,\mathbf{w}}\big| \cdot \big|Y_{2,1,\mathbf{w}'}\big|\Big)^k.$$

Since the equation above holds for every $k \geq 1$, the two sides must have the same number of positive terms. By definition, we have $Y_{i,j,\mathbf{w}}$ is nonempty for all $\mathbf{w} \in W_{i,j}$. As a result, we have

$$|W_{1,1}| \cdot |W_{2,2}| = |W_{1,2}| \cdot |W_{2,1}|$$

and (20) follows. This finishes the proof of Lemma 17. □

References

[1] P. Austrin and E. Mossel. Approximation resistant predicates from pairwise independence. In *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity*, pages 249–258, 2008.

[2] A.A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science*, pages 321–330, 2003.

[3] A.A. Bulatov. A dichotomy theorem for constraints on a three-element set. *Journal of the ACM*, 53(1):66–120, 2006.

[4] A.A. Bulatov. The complexity of the counting constraint satisfaction problem. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, pages 646–661, 2008.

[5] A.A. Bulatov and V. Dalmau. A simple algorithm for Mal'tsev constraints. *SIAM Journal on Computing*, 36(1):16–27, 2006.

[6] A.A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. *Information and Computation*, 205(5):651–678, 2007.

[7] A.A. Bulatov, M.E. Dyer, L.A. Goldberg, M. Jalsenius, M.R Jerrum, and D. Richerby. The complexity of weighted and unweighted #CSP. *arXiv:1005.2678*, 2010.

[8] A.A. Bulatov and M. Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348(2):148–186, 2005.

[9] A.A. Bulatov and P. Jeavons. An algebraic approach to multi-sorted constraints. In *Proceedings of 9th International Conference on Principles and Practice of Constraint Programming*, 2003.

[10] A.A Bulatov and M.A. Valeriote. Recent results on the algebraic approach to the CSP. In N. Creignou, P.G. Kolaitis, and H. Vollmer, editors, *Complexity of Constraints*, pages 68–92. Springer-Verlag, 2008.

[11] S. Burris and H.P. Sankappanavar. *A course in universal algebra*, volume 78 of Graduate Texts in Mathematics. Springer-Verlag, New York-Berlin, 1981.

[12] J.-Y. Cai and X. Chen. A decidable dichotomy theorem on directed graph homomorphisms with non-negative weights. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, 2010.

[13] J.-Y. Cai, X. Chen, and P. Lu. Graph homomorphisms with complex values: A dichotomy theorem. In *Proceedings of the 37th International Colloquium on Automata, Languages and Programming*, 2010.

[14] J.-Y. Cai, S. Huang, and P. Lu. From holant to #CSP and back: Dichotomy for holant^c problems. In *Proceedings of the 21st International Symposium on Algorithms and Computation,* also available at *arXiv:1004.0803*, 2010.

[15] J.-Y. Cai, P. Lu, and M. Xia. Holant problems and counting CSP. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 715–724, 2009.

[16] J.-Y. Cai, P. Lu, and M. Xia. Holographic algorithms with matchgates capture precisely tractable planar #CSP. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 427–436, 2010.

[17] I. Dinur, E. Mossel, and O. Regev. Conditional hardness for approximate coloring. *SIAM Journal on Computing*, 39(3):843–873, 2009.

[18] M.E. Dyer, L.A. Goldberg, and M. Paterson. On counting homomorphisms to directed acyclic graphs. *Journal of the ACM*, 54, 2007.

[19] M.E. Dyer and C. Greenhill. The complexity of counting graph homomorphisms. In *Proceedings of the 9th International Conference on Random Structures and Algorithms*, pages 260–289, 2000.

[20] M.E. Dyer and D.M. Richerby. An effective dichotomy for the counting constraint satisfaction problem. *arXiv:1003.3879*, 2010.

[21] M.E. Dyer and D.M. Richerby. On the complexity of #CSP. In *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 725–734, 2010.

[22] M.E. Dyer and D.M. Richerby. The #CSP dichotomy is decidable. In *Proceedings of the 28th Symposium on Theoretical Aspects of Computer Science*, 2011.

[23] 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.

[24] R. Freese and R. McKenzie. *Commutator Theory for Congruence Modular Varieties*. Cambridge University Press, 1987.

[25] L.A. Goldberg, M. Grohe, M. Jerrum, and M. Thurley. A complexity dichotomy for partition functions with mixed signs. *SIAM Journal on Computing*, 39(7):3336–3402, 2010.

[26] J. Hastad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.

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

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

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

[30] P.G. Jeavons, D.A. Cohen, and M.C. Cooper. Constraints, consistency and closure. *Artificial Intelligence*, 101:251–265, 1998.

[31] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for max-cut and other 2-variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.

[32] G. Kun and M. Szegedy. A new line of attack on the dichotomy conjecture. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 725–734, 2009.

[33] L. Lovász. Operations with structures. *Acta Mathematica Hungarica*, 18:321–328, 1967.

[34] P. Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th annual ACM symposium on Theory of computing*, pages 245–254, 2008.

[35] P. Raghavendra and D. Steurer. How to round any CSP. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 586–594, 2009.

[36] T.J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th annual ACM symposium on Theory of computing*, pages 216–226, 1978.

[37] M. Thurley. The complexity of partition functions on Hermitian matrices. *arXiv:1004.0992*, 2010.

[38] M. Tulsiani. CSP gaps and reductions in the Lasserre hierarchy. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 303–312, 2009.