Journal of Computer and System Sciences 80 (2014) 217–236

Contents lists available at SciVerse ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

The complexity of complex weighted Boolean #CSP

Jin-Yi Cai [a], Pinyan Lu [b,*], Mingji Xia [c]

[a] Computer Sciences Department, University of Wisconsin, Madison, WI 53706, USA
[b] Microsoft Research Asia, Beijing, 100080, PR China
[c] Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany

ARTICLE INFO

Article history:
Received 16 July 2012
Received in revised form 5 June 2013
Accepted 11 July 2013
Available online 24 July 2013

Keywords:
CSP
Counting problems
Dichotomy theorem

ABSTRACT

We prove a complexity dichotomy theorem for the most general form of Boolean #CSP where every constraint function takes values in the field of complex numbers $\mathbb{C}$. We first give a non-trivial tractable class of Boolean #CSP which was inspired by holographic reductions. The tractability crucially depends on algebraic cancelations which are absent for non-negative numbers. We then completely characterize all the tractable Boolean #CSP with complex-valued constraints and show that we have found all the tractable ones, and every remaining problem is #P-hard. We also improve our result by proving the same dichotomy theorem holds for Boolean #CSP with maximum degree 3 (every variable appears at most three times). The concept of Congruity and Semi-congruity provides a key insight and plays a decisive role in both the tractability and hardness proofs. We also introduce *local* holographic reductions as a technique in hardness proofs.

1. Introduction

The complexity of counting problems is a fascinating subject. Valiant defined the class #P to capture most of these counting problems [2]. Beyond the complexity of individual problems, there has been a great deal of interest in proving complexity dichotomy theorems which state that for a wide class of counting problems, every problem in the class is either computable in polynomial time (tractable) or #P-hard [3–9].

In this paper we address the following type of counting problems, called Boolean #CSP [10,11]. Let $\mathscr{F}$ be a set of functions, where each $F \in \mathscr{F}$ is a function $F : \{0,1\}^k \to \mathbb{C}$, mapping Boolean variables to the complex numbers. The #CSP problem #CSP($\mathscr{F}$) is defined as follows: The input is a finite set of constraints on Boolean variables $x_1, x_2, \ldots, x_n$ of the form $F(x_{i_1}, x_{i_2}, \ldots, x_{i_k})$, where $F \in \mathscr{F}$. The output is

$$\sum_{x_1,x_2,\ldots,x_n \in \{0,1\}} \prod F(x_{i_1}, x_{i_2}, \ldots, x_{i_k}).$$

If each F takes values 0, 1, then this counts the number of assignments "satisfying" all the Boolean constraints. In general, functions $F \in \mathscr{F}$ can take arbitrary values. Complexity dichotomy theorems have been obtained for many cases [3,6,12,13,8]. Dyer, Goldberg and Jerrum [9] showed that if all functions in $\mathscr{F}$ take non-negative values, then the counting problem is solvable in polynomial time in precisely the following two cases, and is #P-hard in all other cases: (1) every function in $\mathscr{F}$ is of a product type (a product of unary functions, binary equality functions and binary disequality functions); and (2) every function in $\mathscr{F}$ is a *pure affine* function (a constant on an affine subspace and zero on other inputs).

* Corresponding author.
E-mail addresses: jyc@cs.wisc.edu (J.-Y. Cai), pinyanl@microsoft.com (P. Lu), xmjljx@gmail.com (M. Xia).

0022-0000/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcss.2013.07.003

In this paper we consider problems #CSP($\mathscr{F}$) where functions $F \in \mathscr{F}$ take arbitrary complex values. The presence of both positive and negative values, and more generally, complex numbers, offers the opportunity for interesting and more general cancelations, which could lead to efficient algorithms. It turns out that this is indeed the case. We discover a non-trivial class of tractable #CSP($\mathscr{F}$) problems, where algebraic cancelation is crucial.

We came to this class of tractable #CSP($\mathscr{F}$) from a novel direction, that of Holant problems and holographic reductions, first proposed by Valiant [14–17]. As this subject is still relatively new, we give a brief description of it. A *signature grid* $\Omega = (G, \mathscr{F})$ is a tuple, where $G = (V, E)$ is a graph, and each $v \in V(G)$ is assigned a function $F_v \in \mathscr{F}$. A Boolean assignment σ for every $e \in E$ gives an evaluation $\prod_{v\in V} F_v(\sigma|_{E(v)})$, where $E(v)$ denotes the incident edges of v. The counting problem on an input instance Ω is to compute

$$\text{Holant}(\Omega) = \sum_{\sigma} \prod_{v\in V} F_v(\sigma|_{E(v)}).$$

For example, consider the Perfect Matching problem on G. This problem corresponds to attaching the Exact-One function at every vertex of G, and the sum in Holant(Ω) over all 0–1 edge assignments counts the number of perfect matchings. If we used the At-Most-One function at every vertex, then we are counting all (not necessarily perfect) matchings.

There is a simple relation between #CSP and Holant problems. We can represent an instance of a #CSP problem by a bipartite graph G where the left-hand side (LHS) is labeled by variables and the right-hand side (RHS) is labeled by constraints. We define a signature grid Ω on G by assigning an Equality function to every variable node on LHS (and every constraint node on RHS has the given constraint function). Then Holant(Ω) is exactly the same as the #CSP counting problem. In effect, the Equality function on each variable node forces the incident edges take the same value; this effectively reduces edge assignments in Holant(Ω) to vertex assignments on LHS in the #CSP problem. Thus #CSP problems are precisely the special case of Holant problems on bipartite graphs where every vertex on LHS is assigned an Equality function.

On the other hand, Holant problems can be considered as #CSP problems where every variable appears twice. Note that being syntactically more restrictive in Holant problems makes it more challenging to prove dichotomy theorems, since some techniques, such as many “gadget constructions”, take us out of the class. By the same token, to prove #P-hardness for #CSP problems where each variable appears at most 3 times is more difficult.

In the study of Holant problems, we discovered that the following three families of functions are tractable. (We list each function as a vector of function values ordered according to the lexicographic order of the corresponding inputs. We call such a vector its truth table. In the following, we denote $i = \sqrt{-1}$.)

$$\mathscr{F}_1 = \{\lambda([1,0]^{\otimes k} + i^r[0,\ \ 1]^{\otimes k})\};$$
$$\mathscr{F}_2 = \{\lambda([1,1]^{\otimes k} + i^r[1,-1]^{\otimes k})\};$$
$$\mathscr{F}_3 = \{\lambda([1,i]^{\otimes k} + i^r[1,\ -i]^{\otimes k})\},$$

where $\lambda \in \mathbb{C}$, $k = 1, 2, \ldots,$ and $r = 0, 1, 2, 3$.

We can show that Holant(Ω) for any $\Omega = (G, \mathscr{F}_1 \cup \mathscr{F}_2 \cup \mathscr{F}_3)$ is computable in polynomial time. They are all related to each other by holographic reductions.

We note that complex-valued functions appear naturally. The special case where $r = 1$, $k = 2$ and $\lambda = (1+i)^{-1}$ in $\mathscr{F}_3$ is noteworthy. In this case we get a real-valued function $H(0,0) = H(0,1) = H(1,0) = 1$ and $H(1,1) = -1$. The matrix form of this function is the Hadamard matrix $H = \left(\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right)$. If we take $r = 0$, any k and $\lambda = 1$ in $\mathscr{F}_1$ we get the Equality function on k bits. If $\Omega = (G, \mathscr{F})$ with $\mathscr{F}$ consists of exactly the function H and all Equality functions, then Holant(Ω) is computing the partition function for graph homomorphism $Z_H(G)$. This problem essentially counts the number of induced subgraphs with an odd number of edges. The complexity of $Z_H(G)$ had been open for some time [6] and was independently proved to be tractable in a magnificent paper by Goldberg et al. [7] and by the conference version of this paper [1] (see also [18]) as a consequence that Holant(Ω) for any $\Omega = (G, \mathscr{F}_1 \cup \mathscr{F}_2 \cup \mathscr{F}_3)$ is computable in polynomial time. In [7] they proved a dichotomy theorem for all real-valued partition functions of graph homomorphism. We note that even though some members of $\mathscr{F}_1 \cup \mathscr{F}_2 \cup \mathscr{F}_3$ are real-valued functions, holographic reductions connect them all together and inextricably lead to complex-valued functions. We prefer to investigate the complexity of #CSP type problems in the complex domain $\mathbb{C}$ whenever possible, not only for esthetic reasons, but also because it is in the complex domain $\mathbb{C}$ the intrinsic relationship is revealed via eigenvalues and eigenvectors. This is the reason underlying why the real-valued function H appears in the family $\mathscr{F}_3$ whose definition uses complex numbers. Our investigation of complex-valued constraint functions is also motivated by partition functions in quantum physics. In classical statistical physics, the partition function is always real-valued. However, in a generic quantum system for which complex numbers are the right language, the partition function is in general complex-valued. In particular, if the physics model is over a discrete graph and is non-orientable, then the edge weights defining the partition function of a spin system is a symmetric complex matrix.

After the discovery of this tractable family $\mathscr{F}_1 \cup \mathscr{F}_2 \cup \mathscr{F}_3$, the question naturally arises as to whether there are other kinds of non-trivial cancelations which lead to efficient algorithms. Our initial guess was surely there are other tractable Boolean #CSP($\mathscr{F}$) problems, given our surreptitious discovery of $\mathscr{F}_1 \cup \mathscr{F}_2 \cup \mathscr{F}_3$ as a by-product of holographic reductions. The surprising result is that there are none.

This is our main result. We prove a complexity dichotomy theorem for complex-valued Boolean #CSP. The tractability proof for the symmetric function family $\mathscr{F}_1 \cup \mathscr{F}_2 \cup \mathscr{F}_3$ also proves the tractability for its natural generalization to asymmetric functions. The dichotomy theorem says that a Boolean #CSP is tractable iff *either* all its constraint functions $\mathscr{F}$ are of a certain product type, *or* all are from this generalized family. (See Theorem 3.1.) Bulatov et al. independently proved a dichotomy theorem for all Boolean #CSP with real-valued constraint functions [19]. Our dichotomy theorem generalizes both dichotomy theorems in [9] and [19].

Because we have to rule out all other manners of fortuitous cancelations similar to Theorem 4.1 this part of the proof is more difficult. We isolate a property we call Congruity and Semi-congruity, which provides a key insight and plays a decisive role in both the tractability and hardness proofs.

Our second main theorem, Theorem 3.2, gives a refinement of the first, by restricting the maximum occurrence of each variable to 3 times. This part of the proof is more demanding. The starting point is the dichotomy theorem just proved for Boolean #CSP, Theorem 3.1. We introduce a new technique called *local* holographic reductions. We use this technique together with the method called polynomial interpolation [2,20,4] to prove our second main theorem. The use of holographic reductions implicitly or explicitly seems crucial to this part of the proof.

To avoid unnecessary complications with models of computation for $\mathbb{C}$, we will restrict all numbers used to algebraic numbers, although it is possible to add some fixed finite set of transcendental numbers. For simplicity we will still denote it by $\mathbb{C}$.

1.1. Related works and further development

Some results in this paper have been reported at a conference [1]. Independently from our work, a dichotomy for real weighted #CSP was obtained in [19], where each constraint function can take any real values. A dichotomy for complex weighted graph homomorphism was proved in [21]. In both cases, interesting cancelation occurs and brings new tractable problems.

Great progress was made recently for the complexity of counting CSP over general domain. Bulatov proved a sweeping dichotomy theorem for unweighted #CSP [8]. An alternative proof was given by Dyer and Richerby [22]. The dichotomy was extended to non-negative weight [23] and finally to complex weight [24]. In terms of the logical scope of the dichotomy, the final result covers the dichotomy in this paper as a special case.[1] However, the dichotomy in [24] is not effective while the dichotomy in this paper is very explicit. The dichotomy for maximum degree 3 complex weighted #CSP is not subsumed by these further developments.

After the conference version of the current paper, many new results were obtained based on the dichotomy result in this paper. In [25], a dichotomy for Weighted Boolean #CSP Mod k, for any integer k, was proved. When k is an odd prime, the final result is algebraically exactly the same as the dichotomy in this paper when the imaginary unit $i = \sqrt{-1}$ is interpreted as a fourth primitive root of unity. Crucially based on the dichotomy in this paper, a number of dichotomy theorems were proved in Boolean Holant framework [26–28].

2. Definitions

Let $\mathscr{F}$ be a set of functions, where each $F \in \mathscr{F}$ is a function $F : \{0,1\}^k \to \mathbb{C}$, mapping Boolean variables to $\mathbb{C}$. The #CSP problem #CSP($\mathscr{F}$) is defined as follows: The input is a finite set of constraints on Boolean variables $x_1, x_2, \ldots, x_n$ of the form $F(x_{i_1}, x_{i_2}, \ldots, x_{i_k})$, where $F \in \mathscr{F}$. The output is

$$\sum_{x_1, x_2, \ldots, x_n \in \{0,1\}} \prod F(x_{i_1}, x_{i_2}, \ldots, x_{i_k}).$$

Definition 2.1. For any positive integer k, we use #R_k-CSP($\mathscr{F}$) to denote all #CSP($\mathscr{F}$) problems where every variable appears in at most k constraints.

A symmetric function F on Boolean variables can be expressed by $[f_0, f_1, \ldots, f_k]$, where f_j is the value of F on inputs of weight j. Let $=_k$ denote the equality function of arity k, that is, $(=_k) = [1, 0, \ldots, 0, 1]$, where there are $k-1$ zeros in the bracket. We also use Δ_0, Δ_1 to denote $[1,0]$ and $[0,1]$ respectively. A binary function F is also expressed by the matrix $\left[\begin{smallmatrix} F(0,0) & F(0,1) \\ F(1,0) & F(1,1) \end{smallmatrix}\right]$.

Suppose F is a function on input variables $x_1, x_2, \ldots, x_k$. $F^{x_s=c}$ denotes the function $F^{x_s=c}(x_1, \ldots, x_{s-1}, x_{s+1}, \ldots, x_k) = F(x_1, \ldots, x_{s-1}, c, x_{s+1}, \ldots, x_k)$, and $F^{x_s=*}$ denotes the function $F^{x_s=*}(x_1, \ldots, x_{s-1}, x_{s+1}, \ldots, x_k) = \sum_{x_s} F(x_1, \ldots, x_k)$. A function of arity k can be expressed by its truth table of length 2^k.

[1] While the new #CSP dichotomy on an arbitrary domain D [24] does imply logically a complexity dichotomy on the Boolean domain, it is not clear how to derive our main theorem in this paper, especially the specific dichotomy criterion, from this general theorem.

The underlying relation of F is given by $R_F = \{X \in \{0,1\}^k \mid F(X) \neq 0\}$. A relation $R \subseteq \{0,1\}^k$ is affine means it is the affine subspace composed of solutions of a system of linear equations; equivalently, if $a, b, c \in R$, then the bit-wise XOR $a \oplus b \oplus c \in R$ [11]. If R_F is affine, we say F has affine support. We also view relations as functions from $\{0,1\}^k$ to $\{0,1\}$.

Because a global constant factor does not affect the complexity of a counting problem, we may regard a function F and $c \cdot F$ as the same function, where c is a non-zero constant in $\mathbb{C}$, and we sometimes omit such a factor.

Now we also define another related counting framework called Holant. Let $\mathscr{F}$ be a set of functions. A *signature grid* is a tuple $\Omega = (G, \mathscr{F}, \pi)$, where $G = (V, E)$ is an undirected graph, and $\pi : V \to \mathscr{F}$ labels each $v \in V$ with a function $f_v \in \mathscr{F}$ where the arity of f_v equals the degree of v. f_v takes the incident edges of v as input variables. And if the function is not symmetric then we also assume a 1–1 association is given between the incident edges and the input variables. The Holant problem on instance Ω is to compute

$$\mathrm{Holant}_\Omega = \sum_{\sigma : E \to \{0,1\}} \prod_{v \in V} f_v(\sigma|_{E(v)}),$$

a sum over all 0–1 edge assignments, of the products of the function evaluations at each vertex. Given a set of functions $\mathscr{F}$, we define the problem Holant($\mathscr{F}$):

- Input: A *signature grid* $\Omega = (G, \mathscr{F}, \pi)$;
- Output: Holant_Ω.

We can also define a bipartite Holant problem. Given two sets of functions $\mathscr{F}$ and $\mathscr{G}$, we define a counting problem Holant($\mathscr{G} \mid \mathscr{F}$):

- Input: A *signature grid* $\Omega = (G, \mathscr{G}, \mathscr{F}, \pi)$, where $G = (V_1, V_2, E)$ is a bipartite graph, and π maps V_1 to $\mathscr{G}$ and maps V_2 to $\mathscr{F}$;
- Output: Holant_Ω.

We also use $\#\mathscr{G} \mid \mathscr{F}$ to denote Holant($\mathscr{G} \mid \mathscr{F}$).

By the definitions, it can be seen that #CSP($\mathscr{F}$) is exactly Holant($\{=_1, =_2, =_3, \ldots\} \mid \mathscr{F}$), $\#R_3$-CSP($\mathscr{F}$) is just Holant($\{=_1, =_2, =_3\} \mid \mathscr{F}$), and the Holant problem Holant($\mathscr{F}$) corresponds to the computation of the value of Holant($\{=_2\} \mid \mathscr{F}$).

The following is Valiant's Holant Theorem [14].

Theorem 2.1 *(Holant Theorem). $\#\mathscr{G} \mid \mathscr{F}$ is equivalent to $\#\widetilde{\mathscr{G}} \mid \widetilde{\mathscr{F}}$, where $\widetilde{\mathscr{F}} = \{M^{\otimes \mathrm{arity}(f)} f \mid f \in \mathscr{F}\}$ and $\widetilde{\mathscr{G}} = \{g(M^{-1})^{\otimes \mathrm{arity}(g)} \mid g \in \mathscr{G}\}$, for any 2×2 non-singular matrix M. Here we write $f \in \mathscr{F}$ as a column vector of dimension $2^{\mathrm{arity}(f)}$ and write $g \in \mathscr{G}$ as a row vector of dimension $2^{\mathrm{arity}(g)}$.*

3. Results and proof outline

We define two classes of functions, for which the #CSP problems are tractable.

Let X denote the $(k+1)$-dimensional column vector $(x_1, x_2, \ldots, x_k, 1)^{\mathrm{T}}$ over the Boolean field $\mathbb{Z}_2$. Suppose A is a Boolean matrix. χ_{AX} denotes the affine relation on inputs $x_1, x_2, \ldots, x_k$, whose value is 1 if AX is the zero vector, and 0 if AX is not the zero vector.

Definition 3.1. We denote by $\mathscr{A}$ the set of all functions which have the form $\chi_{AX} \cdot i^{L_1(X)+L_2(X)+\cdots+L_n(X)}$, where $i = \sqrt{-1}$, each L_j is a 0–1 indicator function of the form $\langle \alpha_j, X \rangle$, where α_j is a $(k+1)$-dimensional vector over $\mathbb{Z}_2$, and the dot product $\langle \cdot, \cdot \rangle$ is computed over $\mathbb{Z}_2$.

Recall that we regard a function and any non-zero constant multiple of it as the same function. Thus each function in $\mathscr{A}$ can have an arbitrary constant multiple $\lambda \in \mathbb{C}$. The additions among $L_j(X)$ are the usual addition in $\mathbb{Z}$. It can be computed mod 4, but not mod 2.

Definition 3.2. Define

$$\mathscr{D} = \left\{[a_1, b_1] \otimes [a_2, b_2] \otimes \cdots \otimes [a_k, b_k] \,\middle|\, a_i, b_i \in \mathbb{C}\right\}$$

to be the set of functions that can be expressed as a tensor product of unary functions.

A function in $\mathscr{D}$ on k variables is the product of k unary functions applied to its k variables respectively. (Here $\mathscr{D}$ stands for *degenerate*. A binary function is in $\mathscr{D}$ iff its corresponding matrix is singular.)

Definition 3.3. We denote by $\mathscr{P}$ the set of all functions which can be expressed as a product of unary functions, binary equality functions ($[1,0,1]$) and binary disequality functions ($[0,1,0]$) (on not necessarily disjoint subsets of variables).

We note that $\mathscr{P}$ is a superset of $\mathscr{D}$.

Theorem 3.1. *Suppose $\mathscr{F}$ is a class of functions mapping Boolean inputs to complex numbers. If $\mathscr{F} \subseteq \mathscr{A}$ or $\mathscr{F} \subseteq \mathscr{P}$, then #CSP($\mathscr{F}$) is computable in polynomial time. Otherwise, #CSP($\mathscr{F}$) is #P-hard.*

Our theorem applies to both finite and infinite $\mathscr{F}$. When $\mathscr{F}$ is infinite, for the tractable case, the polynomial time algorithm has input size including the description of the constraint functions used in the instance; for the #P-hard case, a finite subset $\mathscr{F}' \subset \mathscr{F}$ exists such that #CSP($\mathscr{F}'$) is #P-hard.

Proof outline: The polynomial time algorithm for #CSP($\mathscr{P}$) is easy. Section 4 gives a polynomial time algorithm for #CSP($\mathscr{A}$). In dichotomy theorems for unweighted and non-negative weighted Boolean #CSP problems, the tractable part is relatively obvious. In our dichotomy theorem, we have a more interesting tractable part because of cancelations. In Lemma 5.6, we prove that #CSP($\{F\}$) is #P-hard unless F has affine support. This structure is essential in the proof of Lemma 5.7 and Lemma 5.8, the two key lemmas of the hardness reduction. The common strategy of Lemma 5.7 and Lemma 5.8 is to reduce the arity of a given function. In Lemma 5.7, we prove that given a function F not in $\mathscr{A}$, we can simulate (in polynomial time) a unary function $F' \notin \mathscr{A}$; in Lemma 5.8, we prove that given a function G not in $\mathscr{P}$, we can simulate (in polynomial time) a binary or a ternary function $G' \notin \mathscr{P}$. Then we prove that #CSP($\{F', G'\}$) is #P-hard. The starting point of the hardness result is Lemma 5.2, which says that if $\mathscr{F}$ contains only one binary symmetric function and is not in $\mathscr{A} \cup \mathscr{P}$, then the #CSP problem is #P-hard. To complete the proof, we show that we can always combine functions F' and G' to realize a binary symmetric function which is not in $\mathscr{A} \cup \mathscr{P}$.

We also prove a stronger dichotomy theorem that the hardness result holds even for #R_3-CSP($\mathscr{F}$).

Theorem 3.2. *If $\mathscr{F} \nsubseteq \mathscr{A}$ and $\mathscr{F} \nsubseteq \mathscr{P}$, then #$R_3$-CSP($\mathscr{F}$) is #P-hard.*

Note that if we further restrict each variable to occur at most twice, then this dichotomy statement is not true; there are more tractable cases. For example, if every variable occurs exactly twice, then this is the Holant problem Holant($\mathscr{F}$), for which any set of binary functions is tractable under this restriction [18,27,28].

We also remark that the expressibility as $\lambda i^{L_1(X)+L_2(X)+\cdots+L_n(X)}$ is equivalent to an expression of the form $\lambda' i^{Q(X)}$ where Q is a homogeneous quadratic polynomial over $\mathbb{Z}$ with the additional requirement that every cross term $x_s x_t$ has an even coefficient, where $s \neq t$. To see this we observe that each $L_j(X)$ as an integer sum mod 2 can be replaced by $(L_j(X))^2$ as an integer sum mod 4, since $L(X) = 0, 1 \pmod 2$ iff $(L(X))^2 = 0, 1 \pmod 4$. After this, all cross terms have an even coefficient, and i^{x_s} can be replaced by $i^{x_s^2}$. Conversely, we can express Q mod 4 as a sum of squares of affine forms of X, using the extra condition that all cross terms have an even coefficient.

4. Tractable cases

We first show that #CSP($\mathscr{P}$) is tractable. Each constraint function in an instance of #CSP($\mathscr{P}$) is a product of unary functions, binary equality functions and binary disequality functions. We may replace each function by its factors as separate constraints. For the new instance of the #CSP, group variables into connected components depending on whether they are connected by binary functions. Within each connected component, start at any variable with a truth assignment 0 or 1, following any edge labeled by a binary equality or disequality function, there is at most one consistent extension of this assignment to all variables within this connected component. If the extension is inconsistent (along some edge), then the value is 0; otherwise we can easily calculate the value by multiplying the functional evaluations. Thus in each connected component there are at most two assignments with non-zero product values, and these can be easily computed. The value of the problem is the product of its values on each connected component. Hence, #CSP($\mathscr{P}$) is computable in polynomial time.

Now we analyze #CSP($\mathscr{A}$). Firstly, we show how to get rid of the factor χ_{AX}.

Lemma 4.1. *Let $F(x_1, x_2, \ldots, x_k) = \chi_{AX} i^{L_1(X)+L_2(X)+\cdots+L_n(X)} \in \mathscr{A}$. If $AX = 0$ is infeasible over $\mathbb{Z}_2$, then $\sum_{x_1,x_2,\ldots,x_k} F = 0$. Suppose $AX = 0$ is not infeasible. Then in polynomial time, we can construct another function $H(y_1, y_2, \ldots, y_s) = i^{L'_1(Y)+L'_2(Y)+\cdots+L'_n(Y)} \in \mathscr{A}$, such that $0 \leqslant s \leqslant k$, and $\sum_{x_1,x_2,\ldots,x_k} F = \sum_{y_1,y_2,\ldots,y_s} H$.*

Proof. In polynomial time we can solve the linear system $AX = 0$ over $\mathbb{Z}_2$, and decide if it is feasible. Suppose $AX = 0$ is feasible. W.l.o.g., we can assume that $y_1, y_2, \ldots, y_s$ is a set of independent variables over $\mathbb{Z}_2$ and the others are dependent variables, where $0 \leqslant s \leqslant k$. Each dependent variable can be expressed by an affine form of $y_1, y_2, \dots, y_s$. For any $L_j(X)$, we can substitute all the dependent variables and get an affine form of $y_1, y_2, \dots, y_s$, which we denote by $L'_j(Y)$. So we have

$$\sum_{x_1,x_2,\dots,x_k} \chi_{AX} i^{L_1(X)+L_2(X)+\dots+L_n(X)} = \sum_{y_1,y_2,\dots,y_s} i^{L'_1(Y)+L'_2(Y)+\dots+L'_n(Y)}. \qquad \square$$

The following lemma gives a key property of the function $i^{L_1(X)+L_2(X)+\dots+L_n(X)}$. This property plays an important role in both the tractability proof and the hardness proof.

Lemma 4.2. *Let $F(x_1, x_2, \dots, x_k) = i^{L_1(X)+L_2(X)+\dots+L_n(X)}$. Exactly one of the following two statements holds:*

1. (Congruity) *There exists a constant $c \in \{1, -1, i, -i\}$ such that for all $x_2, x_3, \dots, x_k \in \{0, 1\}$ we have $F^{x_1=1}/F^{x_1=0}(x_2, x_3, \dots, x_k) = c$;*
2. (Semi-congruity) *There exists a constant $c \in \{1, i\}$ and an affine subspace S of $T = \{(x_2, x_3, \dots, x_k) \mid x_i \in \mathbb{Z}_2\}$ with $\dim S = k-2$, such that $F^{x_1=1}/F^{x_1=0}(x_2, x_3, \dots, x_k) = c$ on S, and $F^{x_1=1}/F^{x_1=0}(x_2, x_3, \dots, x_k) = -c$ on $T - S$.*

Proof. If for every $1 \leqslant j \leqslant n$, the coefficient of x_1 is zero in $L_j(X)$, then $F^{x_1=1}/F^{x_1=0}$ is a constant 1. Otherwise, w.l.o.g. suppose the coefficients for x_1 are non-zero in exactly the first m affine linear forms $L_j(X)$. Obviously, the other $L_j(X)$'s cancel in the ratio $F^{x_1=1}/F^{x_1=0}$.

For any assignment to $x_2, x_3, \dots, x_k$, consider the two assignments $(0, x_2, x_3, \dots, x_k)$ and $(1, x_2, x_3, \dots, x_k)$. For each $1 \leqslant j \leqslant m$, $L_j(1, x_2, x_3, \dots, x_k) = 1 - L_j(0, x_2, x_3, \dots, x_k)$. Therefore the ratio $F^{x_1=1}/F^{x_1=0} = \prod_{j=1}^m i^{1-2L_j(0,x_2,x_3,\dots,x_k)} = i^m(-1)^{\sum_{j=1}^m L_j(0,x_2,x_3,\dots,x_k)}$. Here m is independent of the assignment on $x_2, x_3, \dots, x_k$. Since the base is -1 now, the sum can be evaluated as a sum mod 2. Therefore there is an affine linear form $\alpha(X) = \sum_{\ell=2}^k \alpha_\ell x_\ell + \alpha_{k+1} \pmod 2$, such that $F^{x_1=1}/F^{x_1=0} = i^m(-1)^{\alpha(X)}$.

If all $\alpha_\ell = 0$, for $2 \leqslant \ell \leqslant k$, then this ratio is a constant and we are in the case of Congruity. If $\alpha_\ell = 1$, for some $2 \leqslant \ell \leqslant k$, then we have Semi-congruity. $\square$

Theorem 4.1. #CSP($\mathscr{A}$) *is polynomial time computable.*

Proof. We first observe that $\mathscr{A}$ is closed under multiplication. Therefore given an instance of #CSP($\mathscr{A}$), the value of the output can be expressed as the summation on a single function $F = \chi_{AX} i^{L_1(X)+L_2(X)+\dots+L_n(X)} \in \mathscr{A}$. We also note that if $F \in \mathscr{A}$, so is $F^{x_s=c}$.

In each step of our algorithm, we reduce the number of variables by at least one and still get a summation of this form.

If the linear system $AX = 0$ over $\mathbb{Z}_2$ is infeasible, the function is a totally zero function and we just output 0. If $AX = 0$ is feasible (including possibly vacuous) then by Lemma 4.1 we can remove the factor χ_{AX} and possibly decrease the number of variables at the same time.

Now we assume it has the form $F = i^{L_1(X)+L_2(X)+\dots+L_n(X)}$, we apply Lemma 4.2 to remove x_1. There are three cases.

Case 1: We have Congruity in Lemma 4.2. Then $F^{x_1=1}/F^{x_1=0}$ is a constant c, and

$$\sum_{x_1,x_2,\dots,x_k} F = (1+c) \cdot \sum_{x_2,x_3,\dots,x_k} F^{x_1=0}.$$

So we get a new summation $\sum_{x_2,x_3,\dots,x_k} F^{x_1=0}$ and have removed a variable x_1.

Case 2: We have Semi-congruity in Lemma 4.2, and $c = 1$. Then on the affine subspace S, the ratio $F^{x_1=1}/F^{x_1=0} = 1$, and on the complementary subspace $T - S$ the ratio $F^{x_1=1}/F^{x_1=0} = -1$. For all $(x_2, x_3, \dots, x_k) \in T - S$, the terms cancel, $F^{x_1=1}(x_2, x_3, \dots, x_k) + F^{x_1=0}(x_2, x_3, \dots, x_k) = 0$. On S, the terms are equal. It follows that

$$\sum_{x_1,x_2,\dots,x_k} F = 2 \sum_{x_2,x_3,\dots,x_k} \chi_S F^{x_1=0}.$$

Note that $\chi_S F^{x_1=0}$ is also a function in $\mathscr{A}$, so we get a new summation of this form and have removed a variable x_1.

Case 3: We have Semi-congruity in Lemma 4.2, and $c = i$. Then for all $(x_2, x_3, \dots, x_k)$ in the affine subspace S, we have $F^{x_1=1}/F^{x_1=0} = i$, and in $T - S$, we have $F^{x_1=1}/F^{x_1=0} = -i$. It follows that

$$\sum_{x_1,x_2,\dots,x_k} F = \sum_S (1+i) F^{x_1=0} + \sum_{T-S} (1-i) F^{x_1=0}.$$

Now we make a crucial observation. The ratio of $1+i$ and $1-i$ is exactly i. As a result we can rewrite the two sums as follows:

$$\sum_{x_1,x_2,\dots,x_k} F = \sum_S (1-i) \cdot F^{x_1=0} \cdot i^{L(X')} + \sum_{T-S} (1-i) \cdot F^{x_1=0} \cdot i^{L(X')},$$

where $L(X')$, on $X' = (x_2, x_3, \dots, x_k, 1)$, is a 0–1 indicator function which takes the value 1 on S and 0 on $T - S$. Thus we can combine the two sums and get

$$\sum_{x_1,x_2,\dots,x_k} F = (1-i) \cdot \sum_{x_2,x_3,\dots,x_k} \left(F^{x_1=0} \cdot i^{L(X')}\right).$$

Note that $F^{x_1=0} \cdot i^{L(X')}$ is also a function in $\mathscr{A}$. So we get a new summation of this form and have removed a variable x_1.

After at most k steps we can eliminate all the variables and obtain the value of the initial summation. Both k and n are bounded by input size. In each iteration, we either resolve a linear system $AX = 0$ or compute a linear equation by Lemma 4.2 representing the affine subspace S, both of which can be done in polynomial time. After each iteration, the formula inside the summation gets at most one more factor $i^{L(X')}$ or χ_S, so the whole algorithm is in polynomial time. □

5. Hardness

In this section, we prove the hardness part of Theorem 3.1, following the outline in Section 3.

Hardness of problems is proved by reductions. In a reduction, we simulate the functions in the original problem by constructing gadgets, polynomial interpolation, or holographic reductions. Let $F \in \mathscr{F}$ be a function of arity d, $1 \leqslant j \leqslant d$ and $c \in \{0, 1\}$. By fixing the jth input as c, we get a function of arity $d - 1$ which is denoted by $F^{x_j=c}$. We also define $F^{x_j=*} = F^{x_j=0} + F^{x_j=1}$ as a function of arity $d - 1$. In #CSP problems, if we have $F \in \mathscr{F}$, we can simply get $F^{x_j=*}$ if x_j does not occur in any other constraint. We can simulate Δ_0 and Δ_1 by the pinning lemma in [9], so we can get $F^{x_j=c}$ using F and Δ_c. We summarize these as the following lemma.

Lemma 5.1. *If $F \in \mathscr{F}$, then* #CSP$(\mathscr{F} \cup \{\Delta_0, \Delta_1, F^{x_j=0}, F^{x_j=1}, F^{x_j=*}\}) \leqslant_T$ #CSP$(\mathscr{F})$.

5.1. One binary function

The starting point of our hardness proof is the following lemma.

Lemma 5.2. *If $[a, b, c] \notin \mathscr{A} \cup \mathscr{P}$, then #CSP$(\{[a, b, c]\})$ is #P-hard. Explicitly, all binary tractable functions $[a, b, c]$ are from $\mathscr{A} \cup \mathscr{P}$, and have one of the following forms: $[x, 0, y]$, $[0, x, 0]$, $[x^2, xy, y^2]$, $x[1, \pm i, 1]$ or $x[1, \pm 1, -1]$, where $x, y \in \mathbb{C}$.*

This lemma says that, if restricted to one single symmetric binary function, our Theorem 3.1 holds. This lemma can also be derived from the general complex weighted Graph Homomorphism problem, for which Cai, Chen and Lu [21] have proved a complete dichotomy theorem, a subsequent result to this.

We give a proof of Lemma 5.2. We first note that every one of the five listed exceptional cases is in $\mathscr{A} \cup \mathscr{P}$, and it can be checked directly that all binary symmetric functions in $\mathscr{A} \cup \mathscr{P}$ take one of these five forms.

In several places of this proof, a reduction method called polynomial interpolation [2,20,4] is used. We first show a simple special case using polynomial interpolation method as applied here. The general method is similar, which involves setting up and then solving a system of linear equations to get the answer of the original problem. The solvability of these linear systems here in this proof is always by the fact that it is a Vandermonde system. (See Section 6, in particular the proof of Lemma 6.1 for more variations on this theme.)

Consider #CSP$(\mathscr{F})$, where $\mathscr{F}$ contains some function $F = [1, a, 1]$. Suppose we want to simulate a function $H = [1, b, 1]$, that is, to reduce #CSP$(\mathscr{F} \cup \{H\})$ to #CSP$(\mathscr{F})$. Given an instance I of #CSP$(\mathscr{F} \cup \{H\})$, where there are n constraints given by H, we construct instances I_j of #CSP$(\mathscr{F})$, by replacing each constraint $H(x_{i_1}, x_{i_2})$ in I by j many constraints $F(x_{i_1}, x_{i_2})$. We use #(I) to denote the value of the #CSP problem instance I. Each occurrence of H takes an input of the form $(0, 0), (0, 1), (1, 0), (1, 1)$. We can write the sum defining #(I) as a sum over all assignments stratified according to the number of $(1, 0)$ or $(0, 1)$ assigned at the n occurrences of H. Let w_i denote the sum over all assignments with exactly i of n occurrences of H assigned $(1, 0)$ or $(0, 1)$ (the other $n - i$ are assigned $(0, 0)$ or $(1, 1)$.) Then the value #(I) can be written as the summation $\#(I) = \sum_{i=0}^{n} w_i b^i$. Meanwhile, we have $\#(I_j) = \sum_{i=0}^{n} w_i a^{ij}$. We let $j = 1, \dots, n + 1$ to get a system of linear equations about w_i, whose coefficient matrix is a Vandermonde matrix (a^{ij}), $i, j = 1, \dots, n + 1$. If a is non-zero and not a root of unity, this is a non-singular matrix, and we can solve for all w_i, which gives us #(I). This is essentially how every reduction by polynomial interpolation in this section will be done.

Our starting point here is the following fact. This lemma is a special case of the dichotomy theorem in [6].

Lemma 5.3. *Let $[a, b, c]$ be a symmetric binary function, where a, b, c are non-negative real numbers. Then #CSP$(\{[a, b, c]\})$ is #P-hard unless $[a, b, c]$ is of one of the following three forms: $[a, 0, c]$; $[0, b, 0]$; or $[x^2, xy, y^2]$.*

Now we consider complex-valued functions. First we prove two simple lemmas.

Lemma 5.4. *For any symmetric binary function $[0, b, c]$, where $bc \neq 0$, #CSP$(\{[0, b, c]\})$ is #P-hard.*

Proof. Since $b \neq 0$, we can normalize it and assume $b = 1$. So we have $[0, 1, c]$. First suppose c is a root of unity. Let $c^k = 1$. We can realize $[0, 1^k, c^k] = [0, 1, 1]$ by k copies of $[0, 1, c]$. This problem is the counting problem for vertex covers, hence it is #P-hard. Now suppose c is not a root of unity. We can realize all $[0, 1, x]$ by polynomial interpolation. In particular, we can realize $[0, 1, 1]$, which is #P-hard. □

Lemma 5.5. *For any symmetric binary function* $[1, b, c]$*, where* $bc \neq 0$ *and* $c \neq b^2$*, there exist two unary functions* $[1, x]$ *and* $[1, y]$ *such that* $\#\mathrm{CSP}(\{[1, b, c], [1, x], [1, y]\})$ *is* #P*-hard.*

Proof. We use F to denote the binary function $[1, b, c]$, and U to denote a unary function $[1, x]$. Then we can realize a binary function G by

$$G(x_1, x_2) = \sum_{x_3} F(x_1, x_3) F(x_3, x_2) U(x_3).$$

It can be computed that $G = [1 + b^2 x, b(1 + cx), b^2 + c^2 x]$. If $c \neq -b^2$, we can choose $x = -\frac{1}{b^2}$, and get $G = [0, \frac{b^2-c}{b}, \frac{b^4-c^2}{b^2}]$. Since $c \neq \pm b^2$, by Lemma 5.4, we know the problem is #P-hard. So we proved that if $c \neq -b^2$, there exists a unary function $[1, x]$ such that $\#\mathrm{CSP}(\{[1, b, c], [1, x]\})$ is #P-hard.

Now suppose $c = -b^2$. We choose $x = -\frac{2}{b^2}$, and get $G = [-1, 3b, -b^2]$. Now for this new symmetric binary function, we can again perform the construction above using a unary function $[1, y]$. Since $b \neq 0$ and $b^2 \neq (3b)^2$, we can prove that the problem is #P-hard. □

Now we prove our main lemma in this section, namely Lemma 5.2: *If* $[a, b, c] \notin \mathscr{A} \cup \mathscr{P}$*, then* $\#\mathrm{CSP}(\{[a, b, c]\})$ *is* #P*-hard.*

Proof of Lemma 5.2. There are several cases. If $a = 0$, we know $bc \neq 0$, otherwise it is in one of the five exceptional cases. So by Lemma 5.4, $\#\mathrm{CSP}(\{[a, b, c]\})$ is #P-hard. The case $c = 0$ is symmetric. Since $[a, b, c] \notin \mathscr{A} \cup \mathscr{P}$, we know $b \neq 0$. Therefore we will assume in the following that $abc \neq 0$, and by normalizing, we can assume $a = 1$.

There are three cases for proving the complexity of $\#\mathrm{CSP}(\{[1, b, c]\})$, with $bc \neq 0$.

1. c is not a root of unity.
   Connect two inputs of $=_3$ by $[1, b, c]$, we can get the function $[1, c]$, and realize any function of the form $[1, x]$ by polynomial interpolation. So by Lemma 5.5, we know that $\#\mathrm{CSP}(\{[a, b, c]\})$ is #P-hard.
2. c is a root of unity, b is not a root of unity.
   Suppose $c^k = 1$. We can realize $[1, b^k, c^k] = [1, b^k, 1]$ by k copies of $[1, b, c]$. Because b is not a root of unity, we can use it to realize $[1, 2, 1]$ (actually any $[1, x, 1]$) by interpolation. This is already #P-hard, by Lemma 5.3.
3. Both b and c are roots of unity.
   We can realize $G = \begin{bmatrix} 1 & b \\ b & c \end{bmatrix}^2 = [1 + b^2, b + bc, b^2 + c^2]$.
   (a) $b = -1$. $G = [2, -1 - c, 1 + c^2]$.
   Since $[1, b, c] = [1, -1, c] \notin \mathscr{A} \cup \mathscr{P}$, we know $c \neq \pm 1$. If $c = \pm i$, we get $G = [2, -1 \mp i, 0]$, which is #P-hard by Lemma 5.4 (or rather a symmetric version of Lemma 5.4, flipping 0 and 1). If $c \notin \{\pm 1, \pm i\}$, then there are no zero entries in G. Since c is a root of unity, and $c \neq \pm 1$, we have $|1 + c^2| \neq 2$. In particular $\frac{1+c^2}{2}$ is not a root of unity. Normalizing we have $[1, \frac{-1-c}{2}, \frac{1+c^2}{2}]$. So $\#\mathrm{CSP}(\{G\})$ is #P-hard by case 1.
   (b) $b = -c$. $G = [1 + c^2, -c - c^2, 2c^2]$.
   Since $[1, b, c] = [1, -c, c] \notin \mathscr{A} \cup \mathscr{P}$, we know $c \neq \pm 1$. If $c = \pm i$, we get $G = [0, 1 \mp i, -2]$, which is #P-hard by Lemma 5.4. If $c \notin \{\pm 1 \pm i\}$, then there are no zero entries in G. Normalizing we get $[1, \frac{-c-c^2}{1+c^2}, \frac{2c^2}{1+c^2}]$. For c a root of unity, the equation $|1 + \frac{1}{c^2}| = 2$ would imply that $c^2 = 1$. As $c \neq \pm 1$, we have $|1 + c^2| \neq |2c^2|$. In particular $\frac{2c^2}{1+c^2}$ is not a root of unity. It follows from case 1 that $\#\mathrm{CSP}(\{G\})$ is #P-hard.
   (c) $c = 1$. $G = [1 + b^2, 2b, 1 + b^2]$.
   Since $[1, b, c] = [1, b, 1] \notin \mathscr{A} \cup \mathscr{P}$, we know $b \notin \{\pm 1, \pm i\}$. So $1 + b^2 \neq 0$, and $|\frac{2b}{1+b^2}| = \frac{2}{|1+b^2|} \neq 1$, so $\frac{2b}{1+b^2}$ is not a root of unity. Therefore the problem is #P-hard by case 2.
   (d) $b \neq -1$, $b + c \neq 0$, $c \neq 1$. Moreover we are given $c \neq b^2$ since $[1, b, c] \notin \mathscr{A} \cup \mathscr{P}$.
   By identifying one input of $[1, b, c]$ and the input of $[1, 1]$, we get $[1 + b, b + c]$. Neither of the two entries is 0. We claim, because $|b| = |c| = 1$, $|1 + b| = |b + c|$ if and only if $c = 1$ or $c = b^2$. Since we assumed $c \neq 1$ and $c \neq b^2$, $\frac{|b+c|}{|1+b|} \neq 1$ and we can interpolate all unary functions with $[1 + b, b + c]$. Therefore $\#\mathrm{CSP}([1, b, c])$ is #P-hard.
   One direction is obvious. In the other direction, because $|1 + b| = |b + c|$, $(1 + b)(1 + \bar{b}) = (b + c)(\bar{b} + \bar{c})$. Expanding this equation, we get $b + \bar{b} = c\bar{b} + \bar{c}b$, i.e., $\mathrm{Re}\, b = \mathrm{Re}\, c/b$. If $b = e^{i\alpha}$ and $c = e^{i\beta}$, then $\cos\alpha = \cos(\beta - \alpha)$, hence $\beta - \alpha = \pm\alpha \bmod 2\pi$. It follows that $c = 1$ or b^2. □

5.2. Non-affine functions

The following lemma generalizes Lemma 11 in [9] to complex weights. However the original proof in [9] does not work for complex weights, due to possible cancelations.

Lemma 5.6. *If R_F is not affine, then* #CSP($\{F\}$) *is* #P-*hard.*

Proof. We prove by induction on the arity of the function F.

All functions of arity one have affine support. The conclusion holds trivially for these functions.

We first consider a function F of arity two. Suppose F does not have affine support. This implies that exactly one of its four values is 0. F can be denoted by the matrix

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} F(0,0) & F(0,1) \\ F(1,0) & F(1,1) \end{bmatrix},$$

then in particular $\det(F) \neq 0$. By taking two copies of F sharing a free variable z in the appropriate order (x,z) and (z,y), we can realize the binary function $H(x,y) = \sum_z F(x,z)F(z,y)$, whose matrix form is $H = FF^T = \left[\begin{smallmatrix} a^2+b^2 & ac+bd \\ ac+bd & c^2+d^2 \end{smallmatrix}\right]$. This H is a symmetric binary function, which can also be denoted by $[a^2+b^2, ac+bd, c^2+d^2]$. We can apply Lemma 5.2 to H. Because F is non-singular, so is the matrix for H. Because exactly one entry of F is 0, $ac+bd \neq 0$ and H is not of the form $[x,0,y]$. Because either $a^2+b^2 \neq 0$ or $c^2+d^2 \neq 0$, H is not of the form $[0,x,0]$. So the only remaining possibility for $H \in \mathscr{A} \cup \mathscr{P}$ is that H is of the form $x[1,\pm i,1]$ or $x[1,\pm 1,-1]$. By symmetry, we only need to consider the cases $a=0$ and $bcd \neq 0$, or $b=0$ and $acd \neq 0$. If $a=0$, we can assume $b=1$ by dehomogenizing, and then the function H is $[1,d,c^2+d^2]$. If H is of the form $x[1,\pm i,1]$, we have $d=\pm i$ and $c=\pm\sqrt{2}$. Then we can realize another symmetric binary function by $H'(x,y) = F(x,y)F(y,x)$. So $H' = [a^2, bc, d^2] = [0, \pm\sqrt{2}, -1]$. #CSP($\{H'\}$) is #P-hard by Lemma 5.2. If H is of the form $x[1,\pm 1,-1]$, we have $d = \pm 1$ and $c = \pm\sqrt{2}i$. Then $H' = [a^2, bc, d^2] = [0, \pm\sqrt{2}i, 1]$ and #CSP($\{H'\}$) is #P-hard by Lemma 5.2 again. So we have completed the $a=0$ case. If $b=0$, we can assume $a=1$ and the function H is $[1,c,c^2+d^2]$. If H is of the form $x[1,\pm i,1]$, we have $c=\pm i$ and $d=\pm\sqrt{2}$. Then we can realize another binary function F' by $F'(x,y) = F(x,y)F(x,y)$. In matrix notation $F' = \left[\begin{smallmatrix} a^2 & b^2 \\ c^2 & d^2 \end{smallmatrix}\right] = \left[\begin{smallmatrix} 1 & 0 \\ -1 & 2 \end{smallmatrix}\right]$. Next we can simulate H' from F' as $H' = F'F'^T = \left[\begin{smallmatrix} 1 & -1 \\ -1 & 5 \end{smallmatrix}\right]$. In symmetric notation $H' = [1,-1,5]$. By Lemma 5.2 #CSP($\{H'\}$) is #P-hard. Finally if H is of the form $x[1,\pm 1,-1]$, we have $c = \pm 1$ and $d = \pm\sqrt{2}i$. Then by the same construction, $F' = \left[\begin{smallmatrix} 1 & 0 \\ 1 & -2 \end{smallmatrix}\right]$ and $H' = \left[\begin{smallmatrix} 1 & 1 \\ 1 & 5 \end{smallmatrix}\right]$, which in symmetric notation is $H' = [1,1,5]$. So again #CSP($\{H'\}$) is #P-hard. We have completed the proof for the case where the function F is of arity two.

Inductively we assume the lemma has been proved for functions with arity $<k$, for some $k \geqslant 3$, and now assume the function F has arity k. Since R_F is not affine, there exist $a,b,c \in R_F$ such that $d = a \oplus b \oplus c \notin R_F$. We only need to prove that we can use F to simulate a function of smaller arity that does not have affine support.

Divide the index set $[k] = \{1,\dots,k\}$ of input variables of F into 4 subsets according to the values of a,b,c as follows:

$$I = \{j \mid a_j = b_j \neq c_j\}, \qquad J = \{j \mid a_j = c_j \neq b_j\},$$
$$K = \{j \mid b_j = c_j \neq a_j\}, \quad \text{and} \quad L = \{j \mid a_j = b_j = c_j\}.$$

Since a_j, b_j and c_j take Boolean values, $\{I,J,K,L\}$ is a partition of $[k]$. For a Boolean variable a, we use $\bar{a}$ to denote its negation. We also remark that, if $j,l \in I$, then either $(a_l,b_l,c_l) = (a_j,b_j,c_j)$ or $(\overline{a_j},\overline{b_j},\overline{c_j})$. A similar statement holds for J, K and L.

Now we have the following four cases, and for each case, we prove our result.

- L is not empty. There exists j such that $a_j = b_j = c_j$.
  We pin the jth input of F to be a_j, and get a function $F^{x_j=a_j}$. Note that $d_j = a_j = b_j = c_j$. This function $F^{x_j=a_j}$ does not have affine support.
  Now we may assume $L = \emptyset$ and $[k] = I \cup J \cup K$.
- There are indices $l \neq j$, such that $(a_l,b_l,c_l) = (a_j,b_j,c_j)$.
  W.l.o.g., we assume $l=1$ and $j=2$. Define a function of arity $k-1$ by $H(x_1,x_3,\dots,x_k) = F(x_1,x_1,x_3,\dots,x_k)$. H can be simulated by F, and by the property that $a,b,c \in R_F$ and yet $d \notin R_F$, H does not have affine support.
- There are indices $l \neq j$, such that $(a_l,b_l,c_l) = (\overline{a_j},\overline{b_j},\overline{c_j})$.
  Clearly both l and j belong to the same set I or J or K. W.l.o.g., we assume $l = 1 \in I$ and $j = 2 \in I$. The proofs for J and K are the same. Then we have $a = (\alpha,\bar{\alpha},a')$, $b = (\alpha,\bar{\alpha},b')$, $c = (\bar{\alpha},\alpha,c')$, and $d = (\bar{\alpha},\alpha,d')$, where $\alpha \in \mathbb{Z}_2$, $a',b',c' \in \mathbb{Z}_2^{k-2}$, and $d' = a' \oplus b' \oplus c' \in \mathbb{Z}_2^{k-2}$. Assume for a contradiction that all functions of the forms $F^{x_i=\beta}$ and $F^{x_i=*}$ have affine support.
  Consider $F^{x_1=\alpha}$, whose underlying relation $R_{F^{x_1=\alpha}}$ is affine. Because $a,b \in R_F$, $(\bar{\alpha},a'),(\bar{\alpha},b') \in R_{F^{x_1=\alpha}}$. The summation of $(\bar{\alpha},a'),(\bar{\alpha},b'),(\alpha,c'),(\alpha,d')$ is the zero vector in $\mathbb{Z}_2^{k-1}$, so $(\alpha,c') \in R_{F^{x_1=\alpha}}$ iff $(\alpha,d') \in R_{F^{x_1=\alpha}}$. This implies that $(\alpha,\alpha,c') \in R_F$ iff $(\alpha,\alpha,d') \in R_F$.

Next consider $F^{x_2=\alpha}$. Because $c \in R_F$ and $d \notin R_F$, we have $(\bar{\alpha}, c') \in R_{F^{x_2=\alpha}}$, and $(\bar{\alpha}, d') \notin R_{F^{x_2=\alpha}}$. We just proved, R_F gives the same value to $(\alpha, \alpha, c'), (\alpha, \alpha, d')$. If R_F gives 1, then $(\alpha, c'), (\alpha, d') \in R_{F^{x_2=\alpha}}$, but this is impossible for the affine relation $R_{F^{x_2=\alpha}}$. So we must have $(\alpha, \alpha, c') \notin R_F$ and $(\alpha, \alpha, d') \notin R_F$.
Similarly, we can prove that both $(\bar{\alpha}, \bar{\alpha}, a')$ and $(\bar{\alpha}, \bar{\alpha}, b') \notin R_F$. More precisely, first consider $F^{x_2=\bar{\alpha}}$. Because $a, b \in R_F$, $(\alpha, a'), (\alpha, b') \in R_{F^{x_2=\bar{\alpha}}}$. Having an affine support, $R_{F^{x_2=\bar{\alpha}}}$ gives the same value to $(\bar{\alpha}, a')$ and $(\bar{\alpha}, b')$. Thus R_F gives the same value to $(\bar{\alpha}, \bar{\alpha}, a')$ and $(\bar{\alpha}, \bar{\alpha}, b')$.
Next consider $F^{x_1=\bar{\alpha}}$. It also has an affine support. Since $c \in R_F$ and $d \notin R_F$, we have $(\alpha, c') \in R_{F^{x_1=\bar{\alpha}}}$ and $(\alpha, d') \notin R_{F^{x_1=\bar{\alpha}}}$. If $(\bar{\alpha}, \bar{\alpha}, a'), (\bar{\alpha}, \bar{\alpha}, b') \in R_F$, then $(\bar{\alpha}, a'), (\bar{\alpha}, b') \in R_{F^{x_1=\bar{\alpha}}}$. This is impossible for an affine relation $R_{F^{x_1=\bar{\alpha}}}$. Thus it follows that $(\bar{\alpha}, \bar{\alpha}, a') \notin R_F$ and $(\bar{\alpha}, \bar{\alpha}, b') \notin R_F$.
To summarize we have all $(\alpha, \alpha, c'), (\alpha, \alpha, d'), (\bar{\alpha}, \bar{\alpha}, a'), (\bar{\alpha}, \bar{\alpha}, b') \notin R_F$.
Finally we consider $F^{x_1=*}$, and calculate as follows:

$$F^{x_1=*}(\bar{\alpha}, a') = F(a) + F(\bar{\alpha}, \bar{\alpha}, a') = F(a) \neq 0,$$
$$F^{x_1=*}(\bar{\alpha}, b') = F(b) + F(\bar{\alpha}, \bar{\alpha}, b') = F(b) \neq 0,$$
$$F^{x_1=*}(\alpha, c') = F(c) + F(\alpha, \alpha, c') = F(c) \neq 0,$$
$$F^{x_1=*}(\alpha, d') = F(d) + F(\alpha, \alpha, d') = F(d) = 0.$$

This is a contradiction with the assumption that $R_{F^{x_1=*}}$ is affine.

- If there are more than one element in sets I or in J or in K, it is included in the previous two cases. The remaining case is that the sizes of I, J, K are all no more than 1 and L is empty. Because $k > 2$, the sizes of I, J, K are exactly 1, and so $k = 3$. W.l.o.g., let $I = \{1\}$, $J = \{2\}$ and $K = \{3\}$.
A moment reflection shows that we can write $a = (p, q, \bar{r})$, $b = (p, \bar{q}, r)$, $c = (\bar{p}, q, r)$, $d = (\bar{p}, \bar{q}, \bar{r})$, where $p, q, r \in \mathbb{Z}_2$.
First we consider $F^{x_1=p}$, which has an affine support, by arity. Let $u = (p, q, r)$, and suppose $u \in R_F$. Then $(q, r) \in R_{F^{x_1=p}}$. Because $a, b \in R_F$, then $(q, \bar{r})$ and $(\bar{q}, r)$ both belong to $R_{F^{x_1=p}}$. Then being affine, $(\bar{q}, \bar{r}) \in R_{F^{x_1=p}}$. Let $v = (p, \bar{q}, \bar{r})$, then $v \in R_F$.
Next we consider $R_{F^{x_2=q}}$. By $a, c \in R_F$, we get $(p, \bar{r}), (\bar{p}, r) \in R_{F^{x_2=q}}$. By assumption $u \in R_F$, then $(p, r) \in R_{F^{x_2=q}}$. By $R_{F^{x_2=q}}$ being affine, we get $(\bar{p}, \bar{r}) \in R_{F^{x_2=q}}$. Let $w = (\bar{p}, q, \bar{r})$, then $w \in R_F$.
Now $a, v, w \in R_F$. This gives us $(p, q), (p, \bar{q}), (\bar{p}, q) \in R_{F^{x_3=\bar{r}}}$. Since $R_{F^{x_3=\bar{r}}}$ is affine, $(\bar{p}, \bar{q}) \in R_{F^{x_3=\bar{r}}}$. This means that $d = (\bar{p}, \bar{q}, \bar{r}) \in R_F$, which is a contradiction.
We conclude that in fact $u \notin R_F$.
By tracing the above steps, under the new condition $u \notin R_F$, we get $v \notin R_F$, and also $w \notin R_F$.
Finally we consider $F^{x_3=r}$. By $b, c \in R_F$, we get $(p, \bar{q}), (\bar{p}, q) \in R_{F^{x_3=r}}$. By $u \notin R_F$, we have $(p, q) \notin R_{F^{x_3=r}}$. By $R_{F^{x_3=r}}$ being affine, we get $(\bar{p}, \bar{q}) \notin R_{F^{x_3=r}}$. i.e., $(\bar{p}, \bar{q}, r) \notin R_F$.
We have now accounted for all 8 points of the form $(\hat{p}, \hat{q}, \hat{r})$, where each bit $\hat{\beta} = \beta$ or $\bar{\beta}$. Exactly three of them a, b, c belong to R_F and the other five points do not. It can be directly verified that $R_{F^{x_1=*}}$ has exactly three points $(q, r), (q, \bar{r}), (\bar{q}, r)$, but not $(\bar{q}, \bar{r})$, which is a contradiction to $R_{F^{x_1=*}}$ being affine. This contradiction completes our proof. $\square$

5.3. Reducing arity

Now we come to the two key lemmas for the hardness proof. Both proofs inductively reduce the arity of a function. Suppose $\mathscr{F} \nsubseteq \mathscr{A}$ and $\mathscr{F} \nsubseteq \mathscr{P}$. Let $F \notin \mathscr{A}$ and $G \notin \mathscr{P}$, where $F, G \in \mathscr{F}$. (It is possible that $G = F$.) From F and G, we recursively simulate functions with smaller arities, keeping the property of being not in $\mathscr{A}$ and not in $\mathscr{P}$ respectively. After the two lemmas we handle the base case of the induction.

Lemma 5.7. *If $F \notin \mathscr{A}$, then either* #CSP($\{F\}$) *is* #P*-hard, or we can simulate a unary function $H \notin \mathscr{A}$, that is, there is a reduction from* #CSP($\{F, H\}$) *to* #CSP($\{F\}$).

Proof. We prove by induction on the arity of the function F. If F has arity one, then we are done since F itself is the unary function we want.

Inductively we assume the lemma has been proved for functions with arity less than k, for some $k \geqslant 2$. Now let F have arity k. In the following proof, for each case, we construct some functions that can be simulated in #CSP($\{F\}$), but have an arity less than k, and then *assume* they are in $\mathscr{A}$ (otherwise, it is proved by induction). Finally we prove that either the problem is #P-hard, or get a unary function $H \notin \mathscr{A}$ or reach a contradiction.

Since the constant function 0 is in $\mathscr{A}$, F has a non-empty support R_F. We first assume R_F is not the whole space $\mathbb{Z}_2^k$. By Lemma 5.6, either #CSP($\{F\}$) is #P-hard, or R_F is affine. Suppose $R_F = \chi_{AX}$, and $x_1, x_2, \ldots, x_s$ $(0 \leqslant s < k)$ are free variables of $AX = 0$. The function $F^{x_{s+1}=*, x_{s+2}=*, \ldots, x_k=*}$ can be simulated by F and has an arity less than k. Thus by our assumption $F^{x_{s+1}=*, x_{s+2}=*, \ldots, x_k=*} \in \mathscr{A}$. Then obviously $F = \chi_{AX} F^{x_{s+1}=*, x_{s+2}=*, \ldots, x_k=*} \in \mathscr{A}$. This is a contradiction to the assumption that $F \notin \mathscr{A}$.

So we may assume $R_F = \mathbb{Z}_2^k$. By our assumption both $F^{x_2=0}, F^{x_2=1} \in \mathscr{A}$, we can apply Lemma 4.2 to these two functions. Accordingly we have the following 3 cases.

1. Both $F^{x_2=0}$ and $F^{x_2=1}$ have Congruity. We will denote the function $F^{x_1=a,x_2=b}$ by F^{ab}. Let c_1 and $c_2 \in \mathbb{Z}_2$ be the two constants for the Congruity of $F^{x_2=0}$ and $F^{x_2=1}$. Thus $F^{10}/F^{00}(x_3,\dots,x_k) = c_1$ and $F^{11}/F^{01}(x_3,\dots,x_k) = c_2$.
   (a) $c_1 = c_2$.
   This means $F^{x_1=1}/F^{x_1=0}$ is a constant c in $\{1,-1,i,-i\}$. Suppose $c = i^r$. Then $F = (i^{x_1})^r F^{x_1=0}$. Since $F^{x_1=0}$ is in $\mathscr{A}$ by arity, this shows that F is also in $\mathscr{A}$. A contradiction.
   (b) $c_1 = -c_2$.
   We will use the notation $[\alpha(X)]$ to denote the 0–1 indicator function for an affine linear form $\alpha(X)$ over $\mathbb{Z}_2$. For any input X, it takes value $0 \in \mathbb{Z}$ if $\alpha(X) = 0$ in $\mathbb{Z}_2$, and it takes value $1 \in \mathbb{Z}$ if $\alpha(X) = 1$ in $\mathbb{Z}_2$.
   Since $c_1 \in \{1,-1,i,-i\}$, there exists an r such that $i^r = c_1/i$. Then we claim that
$$F = \left(i^{[x_1]}\right)^r \cdot i^{[x_1 \oplus x_2]+[x_2]+[x_2]+[x_2]} \cdot F^{x_1=0}.$$
   To verify this, first suppose $x_1 = 0$, then the RHS is $i^{4[x_2]} \cdot F^{x_1=0} = F^{x_1=0}$. Now let $x_1 = 1$, then the RHS is $i^r \cdot i^{1-[x_2]+3[x_2]} \cdot F^{x_1=0} = c_1(-1)^{[x_2]} F^{x_1=0}$. This is $c_1 F^{00} = F^{10}$, if $x_2 = 0$. For $x_2 = 1$, the expression is $-c_1 F^{01} = c_2 F^{01} = F^{11}$. Since $F^{x_1=0}$ has arity less than k, $F^{x_1=0} \in \mathscr{A}$. But then the claim implies that $F \in \mathscr{A}$ as well. A contradiction.
   (c) $c_1 = ic_2$ or $c_1 = -ic_2$.
   Assign an arbitrary assignment for $x_3,\dots,x_k$. Let P be the resulting function on x_1, x_2. In matrix form, where the rows are indexed by $x_1 = 0,1$ and columns are indexed by $x_2 = 0,1$, we have $P = \left[\begin{smallmatrix} u & v \\ \pm ic_2 u & c_2 v \end{smallmatrix}\right]$, for some u, v taking values in $\{\pm 1, \pm i\}$. Let $Q(x_1,x_2) = P^3(0,x_2)P(x_1,x_2)$, realizable by pinning. In matrix form, $Q = \left[\begin{smallmatrix} u^4 & v^4 \\ \pm ic_2 u^4 & c_2 v^4 \end{smallmatrix}\right] = \left[\begin{smallmatrix} 1 & 1 \\ \pm ic_2 & c_2 \end{smallmatrix}\right]$. Here we used the fact that the values of P are powers of i. Now $Q^{x_2=*}$ is a unary function $[2, (1 \pm i)c_2]$ which has unequal non-zero norms $2 \neq |(1 \pm i)c_2| = \sqrt{2}$ and hence not in $\mathscr{A}$.
2. One of $F^{x_2=0}$ and $F^{x_2=1}$ has Congruity and the other has Semi-congruity. Without loss of generality, assume $F^{x_2=0}$ has Congruity and $F^{x_2=1}$ has Semi-congruity. The other case is similar.
   By Congruity, there is a constant $c_1 \in \{1,-1,i,-i\}$, such that $F^{10}/F^{00}(x_3,\dots,x_k) = c_1$ for all $x_3,\dots,x_k \in \mathbb{Z}_2^{k-2}$. By Semi-congruity there is a constant $c_2 \in \{1,-1,i,-i\}$, and a $(k-3)$-dimensional affine linear subspace $S \subset \mathbb{Z}_2^{k-2}$, represented by an affine linear form $\alpha(x_3,\dots,x_k) = 0$, such that on S, $F^{11}/F^{01}(x_3,\dots,x_k) = c_2$ and on $\mathbb{Z}_2^{k-2} - S$, $F^{11}/F^{01}(x_3,\dots,x_k) = -c_2$. We note that to have Semi-congruity, it must be the case that $k \geqslant 3$, and one of the coefficients of $x_3,\dots,x_k$ in $\alpha(x_3,\dots,x_k)$ must be non-zero. W.l.o.g. let it be the coefficient of x_3.
   Fix an arbitrary assignment to $x_4,\dots,x_k$ (if $k = 3$ this step is vacuous), this gives a function $P(x_1,x_2,x_3)$. By changing the constant term in α and c_2 to $-c_2$ if necessary we may assume $x_3 = 0$ gives a point with $\alpha(x_3,\dots,x_k) = 0$.
   Now we will use a special notation to represent $P(x_1,x_2,x_3)$.
$$P = \begin{array}{cccc} z & & & c_1 z \\ & x & c_1 x & \\ & y & c_2 y & \\ w & & & -c_2 w \end{array}.$$
   This symbol is to suggest a cube and is to be read as follows: The left (right) 4 entries are function values with $x_1 = 0$ $(x_1 = 1)$; the top (bottom) 4 entries are function values with $x_2 = 0$ $(x_2 = 1)$; finally the inner (outer) 4 entries are values with $x_3 = 0$ $(x_3 = 1)$.
   Let $Q(x_1,x_2,x_3) = P(x_1,x_2,x_3)(P(0,x_2,x_3))^3$. This corresponds to taking the 3rd power of each of left 4 nodes (x,y,z,w) and multiplying to itself and the node to its right. We get
$$Q = \begin{array}{cccc} 1 & & & c_1 \\ & 1 & c_1 & \\ & 1 & c_2 & \\ 1 & & & -c_2 \end{array}$$
   since $x^4 = y^4 = z^4 = w^4 = 1$. Next let $R(x_1,x_2,x_3) = Q(x_1,x_2,x_3)(Q(x_1,x_2,0))^3$. This gives
$$R = \begin{array}{cccc} 1 & & & 1 \\ & 1 & 1 & \\ & 1 & 1 & \\ 1 & & & -1 \end{array},$$
   since $c_1^4 = c_2^4 = 1$. Then $R^{x_1=0} = \left[\begin{smallmatrix} 1 & 1 \\ 1 & 1 \end{smallmatrix}\right]$, $R^{x_1=1} = \left[\begin{smallmatrix} 1 & 1 \\ 1 & -1 \end{smallmatrix}\right]$. It follows that $R^{x_1=*} = \left[\begin{smallmatrix} 2 & 2 \\ 2 & 0 \end{smallmatrix}\right]$, and $R^{x_1=*,x_2=*} = [4,2]$. It has unequal non-zero norms, hence this unary function $R^{x_1=*,x_2=*} \notin \mathscr{A}$.
3. Both $F^{x_2=0}$ and $F^{x_2=1}$ have Semi-congruity. Let $F^{10}/F^{00} = c_1$ on $\alpha(x_3,\dots,x_k) = 0$ and $-c_1$ on $\alpha(x_3,\dots,x_k) = 1$. Similarly $F^{11}/F^{01} = c_2$ on $\beta(x_3,\dots,x_k) = 0$ and $-c_1$ on $\beta(x_3,\dots,x_k) = 1$. Here $c_1, c_2 \in \{1,-1,i,-i\}$, and α, β are two non-trivial affine linear forms.

(a) $c_1 \neq \pm c_2$. Since β is non-trivial, we may assume the coefficient of x_3 in β is non-zero. Fix any assignment to $x_4, \dots, x_k$, we may assume w.l.o.g. $x_3 = 0$ satisfies $\beta = 0$. We have the following function $P(x_1, x_2, x_3)$, which in our symbolic notation is

$$P = \begin{array}{cccc} z & & & \epsilon c_1 z \\ & x & \delta c_1 x & \\ & y & c_2 y & \\ w & & & -c_2 w \end{array},$$

where $\epsilon, \delta \in \{\pm 1\}$ depending on the assignment of $x_4, \dots, x_k$. If $\epsilon = \delta$, the two entries $\pm c_1 z$ and $\pm c_1 x$ both take the same $+c_1$ or $-c_1$ multiplier, then we have obtained the same ternary function in Case 2 and therefore we can continue in exactly the same way. So assume $\epsilon = -\delta$. By renaming c_1 as $-c_1$, we may assume the two entries are in fact $-c_1 z$ and $+c_1 x$ respectively. Now we take $Q(x_1, x_2, x_3) = P(0, x_2, x_3)^3 P(x_1, x_2, x_3)$. Then we have

$$Q = \begin{array}{cccc} 1 & & & -c_1 \\ & 1 & c_1 & \\ & 1 & c_2 & \\ 1 & & & -c_2 \end{array}.$$

Finally let $R(x_1, x_2, x_3) = Q(x_1, 0, x_3)^3 Q(x_1, x_2, x_3)$. Then

$$R = \begin{array}{cccc} 1 & & & 1 \\ & 1 & 1 & \\ & 1 & c_1^3 c_2 & \\ 1 & & & c_1^3 c_2 \end{array}.$$

Note that $c_1^3 c_2 = c_2 / c_1$, since $c_1^4 = 1$. It is easy to see that $R^{x_1 = *, x_3 = 0} = [2, 1 + c_2/c_1]$. Since $c_2/c_1 \neq \pm 1$ we have $c_2/c_1 = \pm i$. Then this unary function $[2, 1 \pm i] \notin \mathscr{A}$ since it has unequal non-zero norms $2 \neq |1 \pm i|$.

(b) $c_1 = \pm c_2 \in \{1, -1\}$. In this case $F^{x_1=1}/F^{x_1=0}$ only takes values ± 1. Then $R_{F^{x_1=*}}$ is precisely where $F^{x_1=1}/F^{x_1=0} = +1$. If it is not affine, we have #P-hardness by Lemma 5.6. So let $R_{F^{x_1=*}}$ be defined by an affine linear form $\gamma(x_2, \dots, x_k) = 0$. It can be directly verified that

$$F = F^{x_1=0} \cdot i^{[x_1]+[x_1]+[x_1]+[x_1 \oplus \gamma]+[\gamma]+[\gamma]+[\gamma]}.$$

Thus, $F^{x_1=0} \in \mathscr{A}$, which implies that $F \in \mathscr{A}$. A contradiction.

(c) $c_1 = \pm c_2 \in \{i, -i\}$. In this case $F^{x_1=1}/F^{x_1=0}$ only takes values $\pm i$. We may assume $c_1 = c_2 = i$ by changing α to $\alpha \oplus 1$ and/or β to $\beta \oplus 1$ if necessary. Consider the subset

$$\begin{aligned} S &= \left\{ (x_2, x_3, \dots, x_k) \mid F^{x_1=1}/F^{x_1=0} = i \right\} \\ &= \left\{ (0, x_3, \dots, x_k) \mid \alpha(x_3, \dots, x_k) = 0 \right\} \cup \left\{ (1, x_3, \dots, x_k) \mid \beta(x_3, \dots, x_k) = 0 \right\}. \end{aligned}$$

First suppose all coefficients of $x_3, \dots, x_k$ in α and β are the same. If $\alpha(x_3, \dots, x_k) = \sum_{i=3}^k \alpha_i x_i + a$ and $\beta(x_3, \dots, x_k) = \sum_{i=3}^k \alpha_i x_i + b$ over $\mathbb{Z}_2$, then $(a \oplus b)x_2 + \sum_{i=3}^k \alpha_i x_i + a = 0$ over $\mathbb{Z}_2$ defines the set S. Denote this affine linear form by γ, then it can be verified that

$$F = F^{x_1=0} \cdot i^{[x_1 \oplus \gamma]+[\gamma]+[\gamma]+[\gamma]}.$$

Thus, $F^{x_1=0} \in \mathscr{A}$, which implies that $F \in \mathscr{A}$. A contradiction.
Now suppose some coefficients of $x_3, \dots, x_k$ in α and β differ. W.l.o.g. suppose the coefficient of x_3 is 0 in α, and is 1 in β respectively. Fix any assignment to $x_4, \dots, x_k$, then the value of α is fixed, and yet by setting x_3 to 0 or 1, the value of β flips. Then we get a function

$$P(x_1, x_2, x_3) = \begin{array}{cccc} z & & & \epsilon z \\ & x & \epsilon x & \\ & y & \delta y & \\ w & & & -\delta w \end{array}$$

for some $\epsilon, \delta = \pm i$. From here the proof is completed as in Case 2. $\square$

Lemma 5.8. *For any function $F \notin \mathscr{P}$, either #CSP($\{F\}$) is #P-hard, or we can simulate, using F, a function $[a, 0, 1, 0]$ (or $[0, 1, 0, a]$), where $a \neq 0$, or a binary function $H \notin \mathscr{P}$ having no zero values.*

We note that $[a, 0, 1, 0]$ and $[0, 1, 0, a] \notin \mathscr{P}$, for $a \neq 0$.

Proof of Lemma 5.8. Suppose F has arity k. Since $\mathscr{P}$ contains all unary functions and $F \notin \mathscr{P}$, $k \geqslant 2$. Define an $|R_F| \times k$ $\{0, 1\}$-matrix whose rows list every element of R_F, and columns correspond to $x_1, \dots, x_k$. This matrix simply lists all the elements of R_F as rows in some order.

We first remove any column which is all-0 or all-1. If we remove an all-0 column corresponding to x_i, then $X \in R_F \Longrightarrow x_i = 0$. The updated table corresponds to $R_{F^{x_i=0}}$. Similarly if we remove an all-1 column corresponding to x_i, then $X \in R_F \Longrightarrow x_i = 1$. If two columns are identical or are complementary in every bit, we remove one of them. If the columns at x_i and x_j are identical, then $X \in R_F \Longrightarrow x_i \oplus x_j = 0$. Then the updated table removing the column at x_j corresponds to $R_{F^{x_j=*}}$. Similarly for a pair of complementary columns at x_i and x_j we have $X \in R_F \Longrightarrow x_i \oplus x_j = 1$, and the removal of the column at x_j also corresponds to $R_{F^{x_j=*}}$.

We remove columns as long as possible. We claim that this removal process maintains the property of not belonging to $\mathscr{P}$. Suppose we removed an all-0 column at x_i, to get $G = F^{x_i=0}$. Since $X \in R_F \Longrightarrow x_i = 0$, we have $F = \Delta_0(x_i) \cdot G$, where $\Delta_0(x_i)$ is the unary function $[1, 0]$. Thus $G \in \mathscr{P} \Longrightarrow F \in \mathscr{P}$. The case with removing an all-1 column is similar, where we use the unary function $\Delta_1(x_i) = [0, 1]$ instead. If we removed the column at x_j identical to the column at x_i, then $G = F^{x_j=*}$ and $F = \chi_{x_i=x_j} \cdot G$. Finally for the removal of a complementary column at x_j we have $G = F^{x_j=*}$ and $F = \chi_{x_i \oplus x_j \oplus 1=0} \cdot G$. In every step, we maintain $G \notin \mathscr{P}$.

Now we suppose there is some $G \notin \mathscr{P}$ where no more columns can be removed by the above process. There must be some columns left in the table, otherwise the function just before the last column removal is a unary function, hence in $\mathscr{P}$. In fact G being not in $\mathscr{P}$, the arity of G is at least two. For simplicity we still denote it by k. Thus $k \geqslant 2$. We have two cases:

Case 1: $|R_G| < 2^k$. By Lemma 5.6, we may assume R_G is affine, given by a linear system $AX = 0$. We have that $|R_G| = |R_F| \neq 0$, as we never deleted any rows, so the number of rows remains the same. Since G is not unary, the table has more than one column. If $|R_G| = 1$, any two columns (of length one) must be identical or complementary and the removal process should have continued. Thus $|R_G| > 1$. W.l.o.g. assume $x_1, \dots, x_s$ are free variables in $AX = 0$ and $x_{s+1}, \dots, x_k$ are dependent variables. $|R_G| = 2^s$ is a power of 2. We have shown that $s \geqslant 1$. By $|R_G| < 2^k$, $s < k$. We claim $s \geqslant 2$. If instead $s = 1$, then every $x_2, \dots, x_k$ is dependent on x_1 on R_G, so the column at x_2 must be an all-0 or all-1 column, or be identical or complementary to x_1, and the removal process would have continued. The expression of x_k in terms of $x_1, \dots, x_s$ must involve at least two non-zero coefficients; otherwise the column at x_k must be an all-0 or all-1 column, or be identical or complementary to another column. W.l.o.g., say the coefficients of x_1, x_2 are non-zero.

Let $P(x_1, x_2, x_k) = G^{x_3=0,\dots,x_s=0,x_{s+1}=*,\dots,x_{k-1}=*}$ (these two lists of variables could be empty). It can be verified that $R_P = \chi_{x_1 \oplus x_2 \oplus x_k = c}$ for some $c \in \mathbb{Z}_2$.

The affine linear equation $x_1 \oplus x_2 \oplus x_k = c$ is symmetric. Now we define a "symmetrized" function $H(x_1, x_2, x_k) = \prod_{\sigma \in S_3} P(x_{\sigma(1)}, x_{\sigma(2)}, x_{\sigma(k)})$, where S_3 is the symmetry group on three letters $\{1, 2, k\}$. This H is a symmetric function on (x_1, x_2, x_k) and has support $R_H = R_P$. Thus, after normalizing, $H = [a, 0, 1, 0]$ or $[0, 1, 0, a]$ where $a \neq 0$. We remark that this ternary function $H \notin \mathscr{P}$. In fact if H were to be in $\mathscr{P}$, any expression as a member of $\mathscr{P}$ must not use any binary disequality function, since there is a non-zero value for $x_i = x_j = 0$, $i, j \in \{1, 2, k\}$, with the third variable set to c. Similarly it cannot use any binary equality function since there is a non-zero value for $(0, 1, \bar{c})$. But unary functions alone cannot work either.

Case 2: $|R_G| = 2^k$. If for all $1 \leqslant i \leqslant k$, the ratio $G^{x_i=1}/G^{x_i=0}$ is a constant function c_i (since $|R_G| = 2^k$ there are no divisions by zeros), then $G = c_0 \cdot \prod_{1 \leqslant i \leqslant k} U_i(c_i)$, where the constant $c_0 = G^{x_1=0,\dots,x_k=0}$, and $U_i(c_i)$ is the unary function $[1, c_i]$ on x_i. This gives $G \in \mathscr{P}$, a contradiction.

Now suppose for some i, $G^{x_i=1}/G^{x_i=0}$ is not a constant function. W.l.o.g., we assume $i = 1$. The Boolean hypercube on $(x_2, \dots, x_k) \in \{0, 1\}^{k-1}$ is connected by edges which flip just one bit. W.l.o.g., along some flip of x_2 we have two different values. Suppose $G^{x_1=1}/G^{x_1=0}(0, a_3, \dots, a_k) \neq G^{x_1=1}/G^{x_1=0}(1, a_3, \dots, a_k)$, for some $a_3, \dots, a_k$. Set $x_3 = a_3, \dots, x_k = a_k$, we get a binary function $H(x_1, x_2) = G(x_1, x_2, a_3, \dots, a_k)$. We have $H(1, 0)/H(0, 0) \neq H(1, 1)/H(0, 1)$, hence the rank of $H = \left[\begin{smallmatrix} H(0,0) & H(0,1) \\ H(1,0) & H(1,1) \end{smallmatrix}\right]$ is 2.

If H were in $\mathscr{P}$, then partition the variable set according to connectivity by binary equality and disequality functions. If any connected component has at least 2 variables, we can set values to these 2 variables so that $H = 0$. But H is never zero. Then each component must be a single variable and H is defined by a product of unary functions. But such a function has rank 1. This contradiction proves that $H \notin \mathscr{P}$ and completes our proof. $\square$

5.4. *Proof of Theorem 3.1*

Now we are ready to complete the proof for the main Theorem 3.1.

Proof of Theorem 3.1. By Theorem 4.1, #CSP($\mathscr{A}$) is computable in polynomial time. Also #CSP($\mathscr{P}$) is obviously computable in polynomial time.

If $\mathscr{F} \nsubseteq \mathscr{A}$, by Lemma 5.7, either #CSP($\mathscr{F}$) is #P-hard, or we can simulate a function $F = [1, \lambda] \notin \mathscr{A}$. In particular $\lambda \notin \{0, \pm 1, \pm i\}$. If $\mathscr{F} \nsubseteq \mathscr{P}$, by Lemma 5.8, either #CSP($\mathscr{F}$) is #P-hard, or we can simulate a function $P = [a, 0, 1, 0]$, or $P' = [0, 1, 0, a]$, where $a \neq 0$, or a binary function $H \notin \mathscr{P}$ having no zero values.

Firstly, we prove #CSP($\{F, P\}$) is #P-hard. Clearly $P^{x_1=*} = [a, 1, 1]$. If $a \notin \{1, -1\}$, it is #P-hard by Lemma 5.2. If $a \in \{1, -1\}$, we can construct $Q(x_1, x_2) = \sum_{x_3} P(x_1, x_2, x_3) F(x_3) = [a, \lambda, 1]$, which is $[\pm 1, \lambda, 1]$. Both of them are #P-hard by Lemma 5.2. The proof for #CSP($\{F, P'\}$) is the same.

Secondly, we prove #CSP($\{F, H\}$) is #P-hard. After normalizing, we may suppose $H = \left[\begin{smallmatrix} 1 & x \\ y & z \end{smallmatrix}\right]$, where $xyz \neq 0$, and $z \neq xy$. There are two cases, depending on whether $z = -xy$.

For the case $z \neq -xy$, we construct a symmetric function $H(x_1, x_2)H(x_2, x_1) = [1, xy, z^2]$. By the conditions $xyz \neq 0$, $z \neq xy$, $z \neq -xy$, it is impossible to be the first three tractable cases in Lemma 5.2. If it is the last two tractable cases, then xy is a power of i. Now we can form the function $H(x_1, x_2)H(x_2, x_1)F(x_1)F(x_2)$, which is $[1, \lambda xy, \lambda^2 z^2]$. This function has no zero entry and its 2×2 matrix form has rank 2, so it is not of the first three tractable cases in Lemma 5.2. If it were in the last two tractable cases, then λxy is a power of i, which implies that $\lambda = (\lambda xy)/(xy)$ itself is a power of i. However since $[1, \lambda] \notin \mathscr{A}$, we know λ is not a power of i.

For the case $z = -xy$, we construct some binary functions with an integer parameter s as follows:

$$\begin{aligned}\sum_{x_3} H(x_1, x_3)H(x_2, x_3)\bigl(F(x_3)\bigr)^s &= \bigl[1 + \lambda^s x^2, \bigl(y + \lambda^s xz\bigr), \bigl(y^2 + \lambda^s z^2\bigr)\bigr] \\ &= \bigl[1 + \lambda^s x^2, y\bigl(1 - \lambda^s x^2\bigr), y^2\bigl(1 + \lambda^s x^2\bigr)\bigr].\end{aligned}$$

As λ is not a power of i, at most one of the two values x^2 and λx^2 can be a power of i. Now we choose $s = 0$ or $s = 1$ above so that $\lambda^s x^2 \notin \{\pm 1, \pm i\}$.

After normalizing, we may write the function $[1 + \lambda^s x^2, y(1 - \lambda^s x^2), y^2(1 + \lambda^s x^2)]$ as $[1, y(1 - \lambda^s x^2)/(1 + \lambda^s x^2), y^2]$, noticing that $1 + \lambda^s x^2 \neq 0$. We claim that this function is not one of the five tractable cases from Lemma 5.2. Since there are no zero entries, clearly it is not the first two cases. Its matrix form has rank 2, therefore it is not the third case. If it were the fourth tractable case $[1, \pm i, 1]$, then $y = \pm 1$, and $(1 - \lambda^s x^2)/(1 + \lambda^s x^2) = \pm i$. This implies that $\lambda^s x^2 = \pm i$, which is impossible. If $[1, y(1 - \lambda^s x^2)/(1 + \lambda^s x^2), y^2] = [1, \pm 1, -1]$, the fifth tractable case, then $y = \pm i$, and again $(1 - \lambda^s x^2)/(1 + \lambda^s x^2) = \pm i$, also impossible.

The proof of Theorem 3.1 is complete. □

6. Maximum degree 3

In this section we prove Theorem 3.2. This theorem states that our dichotomy theorem holds even when restricted to #CSP problems where every variable appears at most three times. Of course the tractability still applies. The claim is that, for any $\mathscr{F}$ such that $\mathscr{F} \nsubseteq \mathscr{A}$ and $\mathscr{F} \nsubseteq \mathscr{P}$, the #CSP problem on $\mathscr{F}$ over these restricted input instances remains #P-hard.

Assume $\mathscr{F} \nsubseteq \mathscr{A}$ and $\mathscr{F} \nsubseteq \mathscr{P}$, we want to prove the following sequence of reductions:

$$\begin{aligned}\#\mathrm{CSP}(\mathscr{F}) &\leqslant_T \#R_3\text{-CSP}\bigl(\mathscr{F} \cup \{=_2\}\bigr) \\ &\leqslant_T \#R_3\text{-CSP}\bigl(\mathscr{F} \cup \{H\}\bigr) \\ &\leqslant_T \#R_3\text{-CSP}(\mathscr{F}),\end{aligned}$$

where H is a non-degenerate binary function. The first reduction is easy and will be given shortly. We note here that binary equality is for free in general #CSP but it is not free when we consider #CSP with bounded occurrence of variables. In Lemma 6.1 we give the second reduction above. In Theorem 2.1 we give a preparation theorem in which we introduce a localized form of holographic reductions using orthogonal matrices. This theorem is used in the proof of the third reduction above, in Lemma 6.5.

To prove the first reduction, consider an arbitrary #CSP($\mathscr{F}$) instance. Suppose a variable x appears in $\ell > 3$ constraint functions. Our reduction is as follows. We introduce a new variable x' and a new constraint $(=_2)(x, x')$, which requires that the variables x and x' take the same value. Then we replace two appearances of x by x'. After the modification, x' appears 3 times, and x appears $\ell - 1$ times. Repeating this substitution, we can make x, and its copies $x', \ldots$ all appear only 3 times. This modification does not change the value of the #CSP problem. We can do this for every variable by introducing more new variables, and the size of the problem stays polynomially bounded.

Our first key lemma is to show that if we have any non-degenerate binary function $H \in \mathscr{F}$ (this means that the matrix $\left[\begin{smallmatrix} H(0,0) & H(0,1) \\ H(1,0) & H(1,1) \end{smallmatrix}\right]$ is non-degenerate), we can interpolate $(=_2)$. For readers who are familiar with holographic reductions, the use of holographic reductions is unmistakable but implicit here. Dyer and Greenhill [4] proved such a result for a *symmetric* binary function H. We adapt their idea from the symmetric case using the Jordan normal form.

Lemma 6.1. *Let $H : \{0, 1\}^2 \to \mathbb{C}$ be a non-degenerate binary function. Then for any $\mathscr{F}$ containing H, we have*

$$\#R_3\text{-CSP}\bigl(\mathscr{F} \cup \{=_2\}\bigr) \leqslant_T \#R_3\text{-CSP}(\mathscr{F}).$$

Proof. Consider the Jordan normal form of H. There are two cases: either there exist T and $\Lambda = \left(\begin{smallmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{smallmatrix}\right)$, such that $H = T\Lambda T^{-1}$, or there exist T and $\Lambda = \left(\begin{smallmatrix} \lambda & 1 \\ 0 & \lambda \end{smallmatrix}\right)$, such that $H = T\Lambda T^{-1}$.

For the first case, consider an instance I of #R_3-CSP($\mathscr{F} \cup \{=_2\}$). Suppose the function $(=_2)$ appears m times. Replace each occurrence of $(=_2)$ by a chain of T, $(=_2)$, T^{-1}. More precisely, we replace any occurrence of $(=_2)(x,y)$ by $T(x,z) \cdot (=_2)(z,w) \cdot T^{-1}(w,y)$, where z, w are new variables. This defines a new instance I'. Since $TI_2T^{-1} = I_2$, where I_2 denotes the 2×2 identity matrix, the #CSP value of the instance I and I' are the same. We can stratify the CSP sum defining the value on I' according to how many $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$ assignments are given to the occurrences of the new Equality constraints of the form $(=_2)(z,w)$. Clearly any assignment assigning a value $(0,1)$ or $(1,0)$ to some $(=_2)(z,w)$ has a 0 contribution to the sum. Thus we only need to consider those assignments which assign i many times $(0,0)$, and $m-i$ many times $(1,1)$. Let the sum over all such assignments of the evaluation (including those of $T(x,z)$ and $T^{-1}(w,y)$) on I' be ρ_i. Then the CSP value on the instance I' can be written as $\sum_{i=0}^{m} \rho_i$.

Now we construct from I a sequence of instances I'_k indexed by k. Replace each occurrence of $(=_2)(x,y)$ by a chain of k functions H to get an instance I'_k of #R_3-CSP($\mathscr{F}$). More precisely, each occurrence of $(=_2)(x,y)$ is replaced by $H(x,x_1)H(x_1,x_2)\cdots H(x_{k-1},y)$, where $x_1,x_2,\ldots,x_{k-1}$ are new variables (only for this occurrence of $(=_2)(x,y)$). The function of this chain is $H^k = T\Lambda^kT^{-1}$. A moment's reflection shows that the value of the instance I'_k is

$$\sum_{i=0}^{m} \rho_i \lambda_1^{ki} \lambda_2^{k(m-i)} = \lambda_2^{mk} \sum_{i=0}^{m} \rho_i (\lambda_1/\lambda_2)^{ik}.$$

If λ_1/λ_2 is a root of unity, then take a k such that $(\lambda_1/\lambda_2)^k = 1$. (Input size is measured by the number of variables and constraints. The functions in $\mathscr{F}$ are considered constants. Thus this k is a constant.) We have the value $\sum_{i=0}^{m} \rho_i \lambda_1^{ki} \lambda_2^{k(m-i)} = \lambda_2^{mk} \sum_{i=0}^{m} \rho_i$. Since H is non-degenerate, $\lambda_2 \neq 0$, we can compute the value of I from the value of I'_k.

If λ_1/λ_2 is not a root of unity, $(\lambda_1/\lambda_2)^i$, $i = 1,2,\ldots$ never repeat. We can take $k = 1,\ldots,m+1$ and get a system of linear equations about ρ_i. Because the coefficient matrix is Vandermonde in $(\lambda_1/\lambda_2)^i$, $i = 1,2,\ldots,m+1$, we can solve ρ_i and get the value of I.

For the second case, the construction is the same, so we only show the difference with the proof in the first case. Again we can stratify the #CSP value for I' according to how many $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$ assignments are given to the occurrences of the new Equality constraints of the form $(=_2)(z,w)$. We cluster all assignments according to exactly how many times out of m the new Equality constraints of the form $(=_2)(z,w)$ are assigned $(0,0)$ *or* $(1,1)$, and the remaining ones are assigned $(0,1)$. Note that any assignment with a non-zero number of $(1,0)$'s will produce a 0 contribution in the #CSP value for I'_k, after the substitution of each $(=_2)(x,y)$ in I by $H(x,x_1)H(x_1,x_2)\cdots H(x_{k-1},y)$. This is because, by this substitution, effectively each $(=_2)(z,w)$ in I' is replaced by $\Lambda^k = \begin{pmatrix} \lambda^k & k\lambda^{k-1} \\ 0 & \lambda^k \end{pmatrix}$. Again let the sum over all assignments with i many $(0,0)$ *or* $(1,1)$, and $m-i$ many $(0,1)$ of the evaluation (including those of $T(x,z)$ and $T^{-1}(w,y)$) on I' be ρ_i. Then the #CSP value on the instance I' (and on I) is just ρ_m.

The value of I'_k is

$$\sum_{i=0}^{m} \rho_i \lambda^{ki} \bigl(k\lambda^{k-1}\bigr)^{m-i} = \lambda^{(k-1)m} \sum_{i=0}^{m} \bigl(\lambda^i \rho_i\bigr) k^{m-i}.$$

We can take $k = 1,\ldots,m+1$ and get a system of linear equations on $\lambda^i\rho_i$. Because the coefficient matrix is a Vandermonde matrix of full rank, we can solve $\lambda^i\rho_i$. Since H is non-degenerate, $\lambda \neq 0$, we can compute ρ_m, which is the value on I. $\square$

In the following argument, it is easier if we view #R_3-CSP($\mathscr{F}$) as Holant($\{=_1,=_2,=_3\} \mid \mathscr{F}$). Lemma 6.1 shows how to interpolate $(=_2)$ on the RHS in the setting of Holant($\{=_1,=_2,=_3\} \mid \mathscr{F}$), when $\mathscr{F}$ contains a non-degenerate binary function H.

Our next step is to realize a non-degenerate binary function H from $\mathscr{F}$ which is assumed to be neither a subset of $\mathscr{A}$ nor a subset of $\mathscr{P}$.

As $\mathscr{F} \nsubseteq \mathscr{P}$, certainly $\mathscr{F} \nsubseteq \mathscr{D}$, therefore there exists some $F \in \mathscr{F}$ and $F \notin \mathscr{D}$. We will prove that if $F \notin \mathscr{D}$, then we can use F to construct a non-degenerate binary function H. That is, Holant($\{=_1,=_2,=_3\} \mid \mathscr{F} \cup \{H\}$) $\leqslant_T$ Holant($\{=_1,=_2,=_3\} \mid \mathscr{F}$). In fact, we prove a stronger statement

$$\text{Holant}\bigl(\{=_1,=_2\} \bigm| \mathscr{F} \cup \{H\}\bigr) \leqslant_T \text{Holant}\bigl(\{=_1,=_2\} \bigm| \mathscr{F}\bigr).$$

This result may be of independent interest in the study of Holant problems. (The unary Equality function $(=_1) = [1,1]$ is the function on one variable that always evaluates to the constant value 1, and is available in all #CSP problems, because adding any number of constraints of the form $(=_1)(x)$, for any variable x, the answer is unchanged.) Another advantage of this restricted construction is that we can use the technique of a *local* holographic reductions without considering its effect on $(=_3)$.

As a first step, we prove

Lemma 6.2. *If a function $F \in \mathscr{F} - \mathscr{D}$, then we can use F, Δ_0, Δ_1 and $(=_2)$ to simulate a non-degenerate binary function H. That is, there exists a non-degenerate binary function H, such that*

$$\text{Holant}\big(\{\Delta_0, \Delta_1, =_2\} \,\big|\, \mathscr{F} \cup \{H\}\big) \leqslant_T \text{Holant}\big(\{\Delta_0, \Delta_1, =_2\} \,\big|\, \mathscr{F}\big).$$

Proof. Suppose the arity of F is k. All functions of arity one are in $\mathscr{D}$, so $k \neq 1$. If $k = 2$, we let $H = F$.

Suppose $k \geqslant 3$ and the conclusion holds for arity less than k. In the following, whenever we construct (simulate) a function of arity less than k, we always assume the function is in $\mathscr{D}$, for otherwise the lemma is proved by induction. We eventually will reach a contradiction.

We note that, since we can use the unary functions Δ_0 and Δ_1, from the given F we can construct any $F^{x_i=c}$ on $k-1$ variables, where $1 \leqslant i \leqslant k$, $c = 0$ or 1.

If $F^{x_1=0}$ is identically 0, then obviously $F = \Delta_1 \otimes F^{x_1=1} \in \mathscr{D}$. A contradiction.

Now we assume $F^{x_1=0}$ is not identically 0, and suppose $F^{x_1=0}(Y) \neq 0$, for some $Y = y_2 \cdots y_k \in \{0,1\}^{k-1}$. Let $\overline{Y}$ denote $\overline{y_2} \cdots \overline{y_k}$, where $\overline{y_j} = 1 - y_j$. Let $Z = z_2 \cdots z_k$ be any assignment in $\{0,1\}^{k-1}$ such that $Z \neq Y$ and $Z \neq \overline{Y}$. We want to show that,

$$\begin{aligned}&\text{either } F^{x_1=1}(Z) = F^{x_1=0}(Z) = 0,\\ &\text{or } \left[F^{x_1=0}(Z) \neq 0 \text{ and } \frac{F^{x_1=1}(Y)}{F^{x_1=0}(Y)} = \frac{F^{x_1=1}(Z)}{F^{x_1=0}(Z)}\right].\end{aligned} \tag{1}$$

A consequence of (1) is that

$$F^{x_1=1}(Z) = \frac{F^{x_1=1}(Y)}{F^{x_1=0}(Y)} F^{x_1=0}(Z) \tag{2}$$

for all $Z \neq Y$ and $Z \neq \overline{Y}$. Clearly (2) also holds for $Z = Y$.

To prove (1), w.l.o.g., since $Z \neq \overline{Y}$, we suppose $y_2 = z_2 = c$.

Because $F^{x_2=c} \in \mathscr{D}$, and $F^{x_2=c}(0y_3 \cdots y_k) = F^{x_1=0}(Y) \neq 0$, $F^{x_2=c}$ has the form $[1, \lambda] \otimes [a_3, b_3] \otimes \cdots \otimes [a_k, b_k]$. As $y_2 = z_2 = c$, it follows that $F^{x_1=1}(Z) = \lambda F^{x_1=0}(Z)$, and $F^{x_1=1}(Y) = \lambda F^{x_1=0}(Y)$. In particular, $F^{x_1=0}(Z) = 0 \Longrightarrow F^{x_1=1}(Z) = 0$. Thus, either $F^{x_1=1}(Z) = F^{x_1=0}(Z) = 0$, or $[F^{x_1=0}(Z) \neq 0$ and $F^{x_1=1}(Z)/F^{x_1=0}(Z) = \lambda = F^{x_1=1}(Y)/F^{x_1=0}(Y)]$.

Consider all $Z \in \{0,1\}^{k-1}$ such that $Z \neq Y$ and $Z \neq \overline{Y}$. There are two cases:

1. There exists a Z_0, satisfying $Z_0 \neq Y$ and $Z_0 \neq \overline{Y}$ such that $F^{x_1=0}(Z_0) \neq 0$.
   We can substitute Y by Z_0 in the above proof, and since $\overline{Y} \neq Z_0$ and $\overline{Y} \neq \overline{Z_0}$, (1) applies to the pair Z_0 and $\overline{Y}$. Thus either $F^{x_1=1}(\overline{Y}) = F^{x_1=0}(\overline{Y}) = 0$, or $[F^{x_1=0}(\overline{Y}) \neq 0$ and $F^{x_1=1}(Z_0)/F^{x_1=0}(Z_0) = F^{x_1=1}(\overline{Y})/F^{x_1=0}(\overline{Y})]$. It follows that (2) is valid for all $Z \in \{0,1\}^{k-1}$. Hence $F = [1, F^{x_1=1}(Y)/F^{x_1=0}(Y)] \otimes F^{x_1=0} \in \mathscr{D}$. A contradiction.
   Note that for the remaining cases from this case 1 we can assume the support of $F^{x_1=0}$ is contained in $\{Y, \overline{Y}\}$. We observe that if there exists some $Z \neq Y$ and $Z \neq \overline{Y}$, such that $F^{x_1=1}(Z) \neq 0$, then the second alternative of (1) holds. Thus $F^{x_1=0}(Z) \neq 0$ as well, and we are in case 1. Thus, for the remaining case we have:
2. F is zero at all points other than the following four inputs: $(0Y), (1Y), (0\overline{Y}), (1\overline{Y})$.
   By induction $F^{x_1=0} \in \mathscr{D}$ and therefore it has the form $F^{x_1=0} = [a_2, b_2] \otimes \cdots \otimes [a_k, b_k]$. It is zero everywhere except possibly at Y and $\overline{Y}$. If it is non-zero at both points, then $F^{x_1=0}(Y)F^{x_1=0}(\overline{Y}) = a_2 b_2 \cdots a_k b_k \neq 0$. This implies that $F^{x_1=0}$ is non-zero everywhere. Since $k \geqslant 3$, $|\{Y, \overline{Y}\}| = 2 < 2^{k-1}$, this is impossible. Since $F^{x_1=0}(Y) \neq 0$, it must be that $F(0\overline{Y}) = 0$.
   Similarly, because $F^{x_1=1} \in \mathscr{D}$, at most one of $F(1Y)$ and $F(1\overline{Y})$ is non-zero. If $F(1\overline{Y}) = 0$, then F is non-zero only possibly at $(0Y), (1Y)$. Thus $F^{x_2=\overline{y_2}}$ is identically 0, and $F = \Delta(x_2) \cdot F^{x_2=y_2}$, where $\Delta(x_2)$ is a unary function on x_2, such that it takes value 1 if input $x_2 = y_2$ and 0 otherwise. Note that $F^{x_2=y_2} \in \mathscr{D}$ by induction, and $\Delta(x_2)$ is just $[0,1]$ or $[1,0]$ on x_2, it follows that $F \in \mathscr{D}$. A contradiction.
   Hence $F(1\overline{Y}) \neq 0$. Therefore $F(1Y) = 0$. We conclude that F is zero everywhere except at inputs $(0Y), (1\overline{Y})$, where it is non-zero. Now we construct $H(x,y) = \sum_{x_2,\ldots,x_k} F(x, x_2, \ldots, x_k)F(y, x_2, \ldots, x_k)$. Then $H = \left[\begin{smallmatrix} F^2(0Y) & 0 \\ 0 & F^2(1\overline{Y}) \end{smallmatrix}\right] \notin \mathscr{D}$. □

Lemma 6.2 shows that if we had available $[0,1]$ and $[1,0]$ then we can construct a non-degenerate binary function H. One way to get $[0,1]$ and $[1,0]$ is via a form of pinning lemma, which usually requires Equality functions. However, not all Equality functions are freely available for $\#R_3$-CSP problems, therefore it is not clear how to get $[0,1]$ and $[1,0]$ by some pinning lemma. The only available unary function is $(=_1) = [1,1]$, but we find $[1,0]$ is easier to analyze than $[1,1]$, so we use a holographic reduction (in fact, an orthogonal holographic reduction) to turn $[1,1]$ into $\Delta_0 = [1,0]$.

It is an algebraic fact that $(=_2)$ is unchanged under an orthogonal holographic transformation: $(=_2)$ can also be written as a row vector $[1,0]^{\otimes 2}+[0,1]^{\otimes 2}$. Then for any orthogonal matrix M, the holographic transformation is

$$\left([1,0]^{\otimes 2}+[0,1]^{\otimes 2}\right)M^{\otimes 2}=\left([1,0]M\right)^{\otimes 2}+\left([0,1]M\right)^{\otimes 2},$$

which is equal to $[1,0]^{\otimes 2}+[0,1]^{\otimes 2}$, as can be easily checked. (We are not claiming $[1,0]M=[1,0]$ and $[0,1]M=[1,0]$, but the equality holds for the *sum* of the tensor products.) Thus $M^{\otimes 2}$ turns $(=_2)$ into $(=_2)$.

We now introduce a technique of *local* holographic reductions. Instead of applying a holographic reduction on the whole signature grid instance, implicitly taken as the default in Theorem 2.1, we can apply it locally for a fragment of an input signature grid instance, which represents a gadget construction. We will construct such a gadget that realizes a non-degenerate binary function using a function not in $\mathscr{D}$. For the *local* orthogonal holographic transformation, the function of gadgets will be changed according to M. In $\operatorname{Holant}(\mathscr{G}\mid\mathscr{F})$, if we construct a gadget, respecting the bipartiteness of the allowed connections by $\operatorname{Holant}(\mathscr{G}\mid\mathscr{F})$, to realize a function H whose inputs are those of functions in $\mathscr{F}$, then we call this gadget a RHS gadget. This gives us a reduction from $\operatorname{Holant}(\mathscr{G}\mid\mathscr{F}\cup\{H\})$ to $\operatorname{Holant}(\mathscr{G}\mid\mathscr{F})$.

Lemma 6.3. *Let H be a function with k variables constructed by a RHS gadget in problem* $\operatorname{Holant}(\mathscr{G}\mid\mathscr{F})$. *Let M be orthogonal. Then we can construct a function $\widetilde{H}=M^{\otimes k}H$ if we use the same gadget in problem* $\operatorname{Holant}(\widetilde{\mathscr{G}}\mid\widetilde{\mathscr{F}})$, *where $\widetilde{\mathscr{F}}=\{M^{\otimes k}F\mid F\in\mathscr{F},\ F \text{ has arity } k\}$ and $\widetilde{\mathscr{G}}=\{FM^{\mathrm{T}^{\otimes \ell}}\mid F\in\mathscr{G},\ F \text{ has arity } \ell\}$.*

Proof. The proof is a simple adaptation of the proof for Theorem 2.1. We insert a binary equality function $(=_2)$ on every edge in the gadget for H. We also insert this $(=_2)$ on the k dangling edges, which are the external input edges for the gadget of H. This will not change the function of the gadget. Then we can apply the holographic reduction defined by M. As mentioned before, the binary equality function $(=_2)$ is invariant under this orthogonal transformation. Every function in $\mathscr{F}$ is transformed to the corresponding function in $\widetilde{\mathscr{F}}$, and similarly for $\mathscr{G}$. The only difference is that the $(=_2)$ in the k dangling edges transform to a binary function with the matrix M, because they only receive one copy of M each. This gives the final $\widetilde{H}=M^{\otimes k}H$. □

Now we return to our problem. We have some $F\notin\mathscr{D}$. We want to prove that there exists a binary function $H\notin\mathscr{D}$ (i.e., H is non-degenerate) such that $\operatorname{Holant}(\{=_1,=_2\}\mid\{F,H\})\leqslant_T\operatorname{Holant}(\{=_1,=_2\}\mid\{F\})$ by constructing a gadget to realize H using F. We choose the orthogonal matrix $M=\frac{1}{\sqrt{2}}\left[\begin{smallmatrix}1&1\\1&-1\end{smallmatrix}\right]$ to do a local holographic reduction. This changes $[1,1]$ to $[1,0]$ after a scaling, and changes $(=_2)$ into $(=_2)$. By Lemma 6.3, if we can construct a RHS binary gadget $\widetilde{H}\notin\mathscr{D}$ in $\operatorname{Holant}(\{\Delta_0,=_2\}\mid\{\widetilde{F}\})$, then we can construct a RHS gadget H in $\operatorname{Holant}(\{=_1,=_2\}\mid\{F\})$, where $\widetilde{H}=M^{\otimes 2}H$.

In problem $\operatorname{Holant}(\{=_1,=_2\}\mid\{F\})$, the holographic reduction is restricted in the gadgets realizing H. This local holographic reduction is illustrated in the following commutative diagram.

$$\begin{tikzcd}
\#\{=_1,=_2\}\mid\{F\} \arrow[r] \arrow[d, leftrightarrow, "M^{\otimes k}"] & H \arrow[d, leftrightarrow, "M^{\otimes 2}"] \\
\#\{\Delta_0,=_2\}\mid\{\widetilde{F}\} \arrow[r] & \widetilde{H}
\end{tikzcd}$$

A simple observation here is

Lemma 6.4. *Let M be a non-singular 2×2 matrix, then a function F with arity k is in $\mathscr{D}$ iff $M^{\otimes k}F$ is in $\mathscr{D}$.*

Applying a local holographic reduction and Lemma 6.4 to the pair $F,\widetilde{F}$ and the pair $H,\widetilde{H}$, we conclude that to construct a gadget realizing a binary function $H\notin\mathscr{D}$ in $\operatorname{Holant}(\{=_1,=_2\}\mid\{F\})$, is equivalent to construct a gadget realizing a binary function $\widetilde{H}\notin\mathscr{D}$ in $\operatorname{Holant}(\{\Delta_0,=_2\}\mid\{\widetilde{F}\})$. This is the following lemma.

Lemma 6.5. *If function $F\notin\mathscr{D}$, then we can use F, $[1,0]$ and $=_2$ to simulate a non-degenerate binary function H. That is,* $\operatorname{Holant}(\{\Delta_0,=_2\}\mid\{F,H\})\leqslant_T\operatorname{Holant}(\{\Delta_0,=_2\}\mid\{F\})$.

Proof. Suppose the arity of F is k. All functions of arity one belong to $\mathscr{D}$. Hence $k\geqslant 2$. If $k=2$, we let $H=F$. Now suppose $k\geqslant 3$ and the conclusion holds for arity less than k.

We have $\Delta_0=[1,0]$. This allows us to fix some inputs to the value 0. If we construct some function not in $\mathscr{D}$ with arity less than k, then the proof is completed by induction; so we always assume any constructed function of arity less than k is in $\mathscr{D}$.

With $[1,0]$ we can construct $F^{x_1=0}$. Therefore we may assume $F^{x_1=0}=[a_2,b_2]\otimes[a_3,b_3]\otimes\cdots\otimes[a_k,b_k]\in\mathscr{D}$. There are three cases:

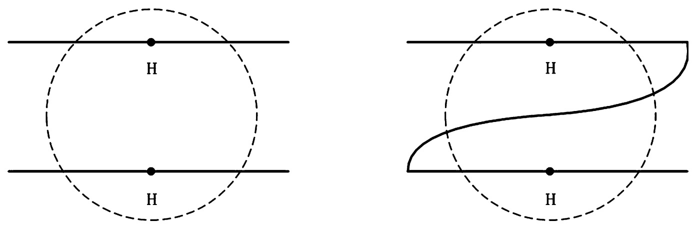

Fig. 1. From $H^{\otimes 2}$ to H^2.

1. $F^{x_1=0}$ is identically zero. This means that there exists some $j \in \{2, \dots, k\}$, such that $a_j = b_j = 0$.
If $F^{x_1=1} \in \mathscr{D}$, then $F = [0, 1] \otimes F^{x_1=1} \in \mathscr{D}$. A contradiction.
Now suppose $F^{x_1=1} \notin \mathscr{D}$. We define the following function $P(x_2, \dots, x_k, y_2, \dots, y_k) = \sum_{x_1,y_1} F(x_1, \dots, x_k)F(y_1, \dots, y_k)I_2(x_1, y_1) = F(1, x_2, \dots, x_k)F(1, y_2, \dots, y_k)$. This function can be obtained by taking two copies of F and connecting the respective two first variables x_1 and y_1 by $(=_2)$. This connecting edge labeled by $(=_2)$ maintains the bipartite structure. Since $F^{x_1=0}$ is identically zero, $P = (F^{x_1=1})^{\otimes 2}$, and we can use it as two individual copies of the function $F^{x_1=1}$, on two sets of disjoint variables $(x_2, \dots, x_k)$ and $(y_2, \dots, y_k)$. Since $F^{x_1=1} \notin \mathscr{D}$, by induction hypothesis, we have a construction for simulating a binary non-degenerate H using $F^{x_1=1}$, Δ_0 and $(=_2)$. Take two copies of this construction, and replace each two copies of the function $F^{x_1=1}$ by one copy of P. This realizes $H^{\otimes 2}$. Let the variables of $H^{\otimes 2}$ be (x_1, y_1, x_2, y_2), where (x_1, y_1) are the variables of the first copy of H and (x_2, y_2) are the variables of the second copy of H. Then connecting two inputs y_1 and x_2 of the four inputs of $H^{\otimes 2}$ by $(=_2)$, we get a non-degenerate binary function whose matrix form is the matrix product H^2. This is illustrated in Fig. 1.
Now we assume $F^{x_1=0}$ is not identically zero. Thus, for all $j \in \{2, \dots, k\}$, if $a_j = 0$ then $b_j \neq 0$.
2. There exists some $j \in \{2, \dots, k\}$, such that $a_j = 0$, and for all $j' \in \{2, \dots, k\}$, if $a_{j'} = 0$ then $b_{j'} \neq 0$.
For all j such that $a_j \neq 0$, fix the value of x_j to be 0 (we have Δ_0) and we get a function Q of the form $[0, c_1] \otimes \cdots \otimes [0, c_m]$, where $m \geqslant 1$ and all $c_j \neq 0$. This is a constant multiple of $(\Delta_1)^{\otimes m}$, which allows us to effectively apply Δ_1 on m separate variables at once. Take m copies of the construction from Lemma 6.2, and then replace every m occurrences of Δ_1 by the $(\Delta_1)^{\otimes m}$ constructed above. We get some $H^{\otimes m}$, for a non-degenerate binary H. Then by the same connection technique we can get H^m, which is a non-degenerate binary function.
3. All a_j are non-zero, for $j \in \{2, \dots, k\}$.
$F^{x_1=0}(0, \dots, 0) = \prod_{j=2}^{k} a_j \neq 0$. Factoring out a global non-zero constant $F^{x_1=0}(0 \cdots 0)$ from $F^{x_1=0}$, we can get $[1, b_2/a_2] \otimes [1, b_3/a_3] \otimes \cdots \otimes [1, b_k/a_k]$. For convenience, we may assume all $a_j = 1$, and the function $F^{x_1=0}$ is $[1, b_2] \otimes [1, b_3] \otimes \cdots \otimes [1, b_k]$.
Consider an arbitrary $Y = y_2 \cdots y_k \neq 1 \cdots 1$. W.l.o.g. suppose $y_2 = 0$. Because we have Δ_0 we can get $F^{x_2=0}$. So $F^{x_2=0} \in \mathscr{D}$ by induction. Let $F^{x_2=0} = [a'_1, b'_1] \otimes G(x_3, \dots, x_k)$. As $F^{x_2=0}(0 \cdots 0) = F^{x_1=0}(0 \cdots 0) \neq 0$, we get $a'_1 \neq 0$. For any $y_3 \cdots y_k \in \{0, 1\}^{k-2}$, $F^{x_2=0}(1y_3 \cdots y_k) = b'_1 G(y_3 \cdots y_k)$, $F^{x_2=0}(0y_3 \cdots y_k) = a'_1 G(y_3 \cdots y_k)$, and so
$$\begin{aligned} F^{x_1=1}(0y_3 \cdots y_k) &= F^{x_2=0}(1y_3 \cdots y_k) \\ &= b'_1 G(y_3 \cdots y_k) \\ &= \frac{b'_1}{a'_1} F^{x_2=0}(0y_3 \cdots y_k) \\ &= \frac{b'_1}{a'_1} F^{x_1=0}(0y_3 \cdots y_k). \end{aligned}$$
So if $F^{x_1=0}(Y) = 0$ then $F^{x_1=1}(Y) = 0$, and if $F^{x_1=0}(Y) \neq 0$ then
$$\frac{F^{x_1=1}(Y)}{F^{x_1=0}(Y)} = \frac{b'_1}{a'_1} = \frac{F^{x_1=1}(0 \cdots 0)}{F^{x_1=0}(0 \cdots 0)},$$
as the expression $\frac{b'_1}{a'_1}$ does not depend on $y_3 \cdots y_k$.
It follows that
$$\begin{aligned} &\text{either } F^{x_1=1}(Y) = F^{x_1=0}(Y) = 0, \\ &\text{or } \left[F^{x_1=0}(Y) \neq 0 \text{ and } \frac{F^{x_1=1}(Y)}{F^{x_1=0}(Y)} = \frac{F^{x_1=1}(0 \cdots 0)}{F^{x_1=0}(0 \cdots 0)} \right]. \end{aligned} \tag{3}$$
For both cases,
$$F^{x_1=1}(Y) = \frac{F^{x_1=1}(0 \cdots 0)}{F^{x_1=0}(0 \cdots 0)} F^{x_1=0}(Y)$$

holds. Hence,

$$F = [1, b_1] \otimes \cdots \otimes [1, b_k] + \delta[0, 1]^{\otimes k} \tag{4}$$

for some number δ, where $b_1 = F^{x_1=1}(0\cdots0)/F^{x_1=0}(0\cdots0)$, and $[0, 1]^{\otimes k}$ is a vector of length 2^k which only affects the value $F(1, \ldots, 1)$. Note that $\delta \neq 0$, for otherwise $F \in \mathscr{D}$.
Next we construct a function P. Note that $F^{x_2=x_3=\cdots=x_k=0} = [1, b_1]$. By Δ_0 we can realize this unary function. Applying the function $[1, b_1]$ to the first input of F (connected by $(=_2)$), we get the following function P

$$P(x_2, \ldots, x_k) = \sum_{x_1}[1, b_1](x_1) \cdot F(x_1, x_2, \ldots, x_k).$$

If $(x_2, \ldots, x_k) \neq (1, \ldots, 1)$,

$$\begin{aligned} P(x_2, \ldots, x_k) &= F(0, x_2, \ldots, x_k) + b_1 F(1, x_2, \ldots, x_k) \\ &= \prod_{j:\, x_j=1} b_j + b_1^2 \prod_{j:\, x_j=1} b_j \\ &= \left(1 + b_1^2\right) \prod_{j:\, x_j=1} b_j, \end{aligned}$$

and

$$P(1, \ldots, 1) = \prod_{j=2}^{k} b_j + b_1\left(\prod_{j=1}^{k} b_j + \delta\right) = \left(1 + b_1^2\right)\prod_{j=2}^{k} b_j + b_1\delta,$$

so

$$P = \left(1 + b_1^2\right)[1, b_2] \otimes \cdots \otimes [1, b_k] + b_1\delta[0, 1]^{\otimes(k-1)}.$$

We note that, starting from the expression (4) the construction of P above can be applied to any x_s, for $1 \leqslant s \leqslant k$.
There are two subcases:

(a) For some $1 \leqslant s \leqslant k$, $b_s = \pm i$.
W.l.o.g. assume $s = 1$. In this case $b_1^2 = -1$ implies that P is identically 0 except on input $(1, \ldots, 1)$, where it is non-zero. So we can use P as $k - 1$ copies of Δ_1, and by a similar argument to the second case, we can apply Lemma 6.2. The conclusion holds.

(b) For all $1 \leqslant s \leqslant k$, $b_s \neq \pm i$.
From F, we get P. We may replace F by P, and repeat this construction until only two variables are left. At this point we get a function of the form, up to a non-zero factor, $Q = [1, c_1] \otimes [1, c_2] + (0, 0, 0, \delta') = \left(\begin{smallmatrix} 1 & c_1 \\ c_2 & c_1c_2+\delta' \end{smallmatrix}\right)$ for some $\delta' \neq 0$. This is a non-degenerate binary function. □

Corollary 6.1. *If F is a function such that $F \notin \mathscr{D}$, then*

$$\operatorname{Holant}\left(\{=_1, =_2\} \mid \{F, H\}\right) \leqslant_T \operatorname{Holant}\left(\{=_1, =_2\} \mid \{F\}\right),$$

for some non-degenerate binary function H.

We can complete the proof of Theorem 3.2 now.

Proof. If $\mathscr{F} \nsubseteq \mathscr{A}$ and $\mathscr{F} \nsubseteq \mathscr{P}$, #CSP($\mathscr{F}$) is #P-hard by Theorem 3.1.
Because $\mathscr{D} \subseteq \mathscr{P}$, then $\mathscr{F} \nsubseteq \mathscr{D}$, by Corollary 6.1 of Lemma 6.5, we can simulate a non-degenerate binary function H. That is, we can reduce #R_3-CSP($\mathscr{F} \cup \{H\}$) to #R_3-CSP($\mathscr{F}$).
Then by Lemma 6.1, we can reduce #R_3-CSP($\mathscr{F} \cup \{=_2\}$) to #R_3-CSP($\mathscr{F} \cup \{H\}$).
Finally, we can reduce #CSP($\mathscr{F}$) to #R_3-CSP($\mathscr{F} \cup \{=_2\}$) as discussed before Lemma 6.1. □

Acknowledgments

We thank many colleagues who gave us valuable feedbacks on this work, with special thanks to Martin Dyer, Leslie Ann Goldberg, Mark Jerrum, Leslie Valiant, Tyson Williams and the anonymous referees for our conference paper at STOC 2009 [1], which contained some of the results reported here.
Jin-Yi Cai is supported by NSF CCF-0830488 and CCF-0914969. Mingji Xia is supported by NSFC 61003030.

References

[1] J.-Y. Cai, P. Lu, M. Xia, Holant problems and counting CSP, in: M. Mitzenmacher (Ed.), STOC, ACM, 2009, pp. 715–724.
[2] L.G. Valiant, The complexity of enumeration and reliability problems, SIAM J. Comput. 8 (1979) 410–421.
[3] N. Creignou, M. Hermann, Complexity of generalized satisfiability counting problems, Inf. Comput. 125 (1996) 1–12.
[4] M. Dyer, C. Greenhill, The complexity of counting graph homomorphisms, in: Proceedings of the 9th International Conference on Random Structures and Algorithms, 2000, pp. 260–289.
[5] M.E. Dyer, L.A. Goldberg, M. Paterson, On counting homomorphisms to directed acyclic graphs, J. ACM 54 (2007).
[6] A.A. Bulatov, M. Grohe, The complexity of partition functions, in: J. Díaz, J. Karhumäki, A. Lepistö, D. Sannella (Eds.), ICALP, in: Lecture Notes in Computer Science, vol. 3142, Springer, 2004, pp. 294–306.
[7] L.A. Goldberg, M. Grohe, M. Jerrum, M. Thurley, A complexity dichotomy for partition functions with mixed signs, SIAM J. Comput. 39 (2010) 3336–3402.
[8] A.A. Bulatov, The complexity of the counting constraint satisfaction problem, in: L. Aceto, I. Damgård, L.A. Goldberg, M.M. Halldórsson, A. Ingólfsdóttir, I. Walukiewicz (Eds.), ICALP (1), in: Lecture Notes in Computer Science, vol. 5125, Springer, 2008, pp. 646–661.
[9] M.E. Dyer, L.A. Goldberg, M. Jerrum, The complexity of weighted Boolean CSP, SIAM J. Comput. 38 (2009) 1970–1986.
[10] T.J. Schaefer, The complexity of satisfiability problems, in: STOC '78: Proceedings of the Tenth Annual ACM Symposium on Theory of Computing, ACM, New York, NY, USA, 1978, pp. 216–226.
[11] N. Creignou, S. Khanna, M. Sudan, Complexity Classifications of Boolean Constraint Satisfaction Problems, SIAM Monographs on Discrete Mathematics and Applications, 2001.
[12] A.A. Bulatov, V. Dalmau, Towards a dichotomy theorem for the counting constraint satisfaction problem, in: FOCS, IEEE Computer Society, 2003, pp. 562–571.
[13] A.A. Bulatov, A dichotomy theorem for constraint satisfaction problems on a 3-element set, J. ACM 53 (2006) 66–120.
[14] L.G. Valiant, Holographic algorithms, SIAM J. Comput. 37 (2008) 1565–1594.
[15] L.G. Valiant, Accidental algorithms, in: FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, IEEE Computer Society, Washington, DC, USA, 2006, pp. 509–517.
[16] J.-Y. Cai, P. Lu, Holographic algorithms: From art to science, J. Comput. Syst. Sci. 77 (2011) 41–61.
[17] J.-Y. Cai, P. Lu, M. Xia, Holographic algorithms by Fibonacci gates and holographic reductions for hardness, in: FOCS '08: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science, IEEE Computer Society, Washington, DC, USA, 2008.
[18] J.-Y. Cai, P. Lu, M. Xia, Computational complexity of Holant problems, SIAM J. Comput. 40 (2011) 1101–1132.
[19] A.A. Bulatov, M.E. Dyer, L.A. Goldberg, M. Jalsenius, D. Richerby, The complexity of weighted Boolean #CSP with mixed signs, Theor. Comput. Sci. 410 (2009) 3949–3961.
[20] S.P. Vadhan, The complexity of counting in sparse, regular, and planar graphs, SIAM J. Comput. 31 (2001) 398–427.
[21] J.-Y. Cai, X. Chen, P. Lu, Graph homomorphisms with complex values: A dichotomy theorem, in: S. Abramsky, C. Gavoille, C. Kirchner, F.M. auf der Heide, P.G. Spirakis (Eds.), ICALP (1), in: Lecture Notes in Computer Science, vol. 6198, Springer, 2010, pp. 275–286.
[22] M. Dyer, D. Richerby, On the complexity of #CSP, in: Proceedings of the 42nd ACM Symposium on Theory of Computing, 2010, pp. 725–734.
[23] J.-Y. Cai, X. Chen, P. Lu, Non-negatively weighted #CSP: An effective complexity dichotomy, in: IEEE Conference on Computational Complexity, IEEE Computer Society, 2011, pp. 45–54.
[24] J.-Y. Cai, X. Chen, Complexity of counting CSP with complex weights, in: STOC, 2012, pp. 909–920.
[25] H. Guo, S. Huang, P. Lu, M. Xia, The complexity of weighted Boolean #CSP modulo *k*, in: J.-Y. Marion, T. Schwentick (Eds.), STACS, in: LIPIcs, vol. 5, Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2011, pp. 249–260.
[26] J.-Y. Cai, S. Huang, P. Lu, From Holant to #CSP and back: Dichotomy for Holant problems, in: O. Cheong, K.-Y. Chwa, K. Park (Eds.), ISAAC (1), in: Lecture Notes in Computer Science, vol. 6506, Springer, 2010, pp. 253–265.
[27] S. Huang, P. Lu, A dichotomy for real weighted Holant problems, in: IEEE Conference on Computational Complexity, IEEE, 2012, pp. 96–106.
[28] J.-Y. Cai, H. Guo, T. Williams, A complete dichotomy rises from the capture of vanishing signatures, arXiv:1204.6445 [cs.CC], 2012.