The expressibility of functions on the Boolean domain, with applications to Counting CSPs[*]

Andrei A. Bulatov[1]   Martin Dyer[2]

Leslie Ann Goldberg[3]   Mark Jerrum[4]   Colin McQuillan[3]

22nd December 2012

Abstract

An important tool in the study of the complexity of Constraint Satisfaction Problems (CSPs) is the notion of a relational clone, which is the set of all relations expressible using primitive positive formulas over a particular set of base relations. Post's lattice gives a complete classification of all Boolean relational clones, and this has been used to classify the computational difficulty of CSPs. Motivated by a desire to understand the computational complexity of (weighted) counting CSPs, we develop an analogous notion of functional clones and study the landscape of these clones. One of these clones is the collection of log-supermodular (lsm) functions, which turns out to play a significant role in classifying counting CSPs. In the conservative case (where all nonnegative unary functions are available), we show that there are no functional clones lying strictly between the clone of lsm functions and the total clone (containing all functions). Thus, any counting CSP that contains a single nontrivial non-lsm function is computationally as hard to approximate as any problem in #P. Furthermore, we show that any non-trivial functional clone (in a sense that will be made precise) contains the binary function "implies". As a consequence, in the conservative case, all non-trivial counting CSPs are as hard as #BIS, the problem of counting independent sets in a bipartite graph. Given the complexity-theoretic results, it is natural to ask whether the "implies" clone is equivalent to the clone of lsm functions. We use the Möbius transform and the Fourier transform to show that these clones coincide precisely up to arity 3. It is an intriguing open question whether the lsm clone is finitely generated. Finally, we investigate functional clones in which only restricted classes of unary functions are available.

1   Introduction

In the classical setting, a (non-uniform) constraint satisfaction problem $\mathsf{CSP}(\Gamma)$ is specified by a finite domain D and a constraint language Γ, which is a set of relations of varying arities over D. For example, D might be the Boolean domain $\{0, 1\}$ and Γ might be the set containing the single relation $\text{NAND} = \{(0, 0), (0, 1), (1, 0)\}$. An instance of $\mathsf{CSP}(\Gamma)$ is a set of n variables taking values in D, together with a set of constraints on those variables. Each constraint is a relation R from Γ applied to a tuple of variables, which is called the "scope" of the constraint. The problem is to find an assignment of domain elements to the variables which satisfies all of the constraints. For example, the problem of finding an independent set in a graph can be represented as a CSP with $\Gamma = \{\text{NAND}\}$. The vertices of the graphs are the variables of the CSP instance. The instance has one NAND constraint for each edge of the graph. Vertices whose variables are mapped to domain element 1 are deemed to be in the independent set. Constraint satisfaction problems (CSPs) may be viewed as generalised satisfiability problems, among which usual satisfiability is a very special case.

The notion of expressibility is key to understanding the complexity of CSPs. A *primitive positive formula* (pp-formula) in variables $V = \{v_1, \ldots, v_n\}$ is a formula of the form

$$\exists v_{n+1} \ldots v_{n+m} \bigwedge_i \varphi_i,$$

where each atomic formula φ_i is either a relation R from Γ applied to some of the variables in $V' = \{v_1, \ldots, v_{n+m}\}$ or an equality relation of the form $v_i = v_j$, which we write as $\text{EQ}(v_i, v_j)$. For example, the formula

$$\exists v_3 \, \text{NAND}(v_1, v_2) \text{EQ}(v_1, v_3) \text{EQ}(v_2, v_3)$$

is a pp-formula in variables v_1 and v_2. This formula corresponds to the relation $\{(0, 0)\}$ since the only way to satisfy the constraints is to map both v_1 and v_2 to the domain element 0.

The *relational clone* $\langle \Gamma \rangle_\text{R}$ is the set of all relations expressible as pp-formulas over Γ. Relational clones have played a key role in the development of the complexity of CSPs because of the following important fact, which is described, for example, in the expository chapter of Cohen and Jeavons [14]: If two sets of relations Γ and Γ' generate the same relational clone, then the computational complexities of the corresponding CSPs, $\mathsf{CSP}(\Gamma)$ and $\mathsf{CSP}(\Gamma')$, are exactly the same. Thus, in order to understand the complexity of CSPs, one does not need to consider all sets of relations Γ. It suffices to consider those that are relational clones.

Recently, there has been considerable interest in the computational complexity of counting CSPs (see, for example [7, 10, 13, 19, 20, 35]). Here, the goal is to count the number of solutions of a CSP rather than merely to decide if a solution exists. In fact, in order to encompass the computation of partition functions of models from statistical physics and other generating functions, it is common (see, for example, [9]) to consider weighted sums, which can be expressed by replacing the relations in the constraint language by real-valued or complex-valued functions. In this case, the weight of an assignment (of domain values to the variables) is the product of the function values corresponding to that assignment, while the value of the CSP instance itself is the sum of the weights of all assignments. If

I is an instance of such a counting CSP then this weighted sum is called the "partition function of I" (by analogy with the concept in statistical physics) and is denoted $Z(I)$. For a finite set of functions Γ we are interested in the problem $\#\mathsf{CSP}(\Gamma)$, which is the problem of computing $Z(I)$, given an instance I which uses only functions from Γ.

Our first goal (see §2) is to determine the most useful analogues of pp-definability and relational clones in the context of (weighted) counting CSPs (#CSPs), and to see what insight this provides into the computational complexity of these problems. It is clear that, in order to adapt the concept of pp-definability to the counting setting, one should replace a conjunction of relations by a product of functions and replace existential quantification by summation. However, there are sensible alternatives for the detailed definitions, and these have ramifications for the complexity-theoretic consequences. There is at least one proposal in the literature for extending pp-definability to the algebraic/functional setting — that of Yamakami [35]. However, we find it useful to adopt a more liberal notion of pp-definability, including a limit operation. Without this, a functional clone could contain arbitrarily close approximations to a function F of interest, without including F itself. We call this analogue of pp-definability "pps_ω-definability". The notion of pps_ω-definability leads to a more inclusive functional clone than the one considered in [35].

Aside from a desire for tidiness, there is a good empirical motivation for introducing limits. Just as pp-definability is closely related to polynomial-time reductions between classical CSPs, so is pps_ω-definability related to approximation-preserving reductions between (weighted) counting CSPs. Lemma 18 is a precise statement of this connection. Many approximation-preserving reductions in the literature (for example, those in [21]) are based not on a fixed "gadget" but on sequences of increasingly-large gadgets that come arbitrarily close to some property without actually attaining it. Our notion of pps_ω-definability is intended to capture this phenomenon.

The second, more concrete goal of this paper (see §3–§9) is to explore the space of functional clones and to use what we learn about this space to classify the complexity of approximating #CSPs. We restrict attention to the Boolean situation so the domain is $\{0, 1\}$ and the allowed functions are of the form $\{0, 1\}^k \to \mathbb{R}^{\geq 0}$ for some integer k. We examine the landscape of functional clones for the case in which all nonnegative unary functions (weights) are available. This case is known as the *conservative* case. It is also studied in the context of decision and optimisation CSPs [6, 28] and in work related to counting CSPs such as Cai, Lu and Xia's work on classifying "Holant*" problems [11]. The conservative case is easier to to classify than the general case, so we are able to construct a useful map of the landscape of functional clones (see Theorem 16). Note that Yamakami [35] has considered an even more special case in which all unary weights (including negative weights) are available. In that case the landscape turns out to be less rich and more pessimistic — negative weights introduce cancellation, which tends to drive approximate counting CSPs in the direction of intractability.

An issue that turns out to be important in the classification of conservative functional clones is *log-supermodularity*. Roughly, a function with Boolean domain is said to be log-supermodular if its logarithm is supermodular. (A formal definition appears later.) It is a non-trivial fact (Lemma 7) that the set LSM of log-supermodular functions is a functional clone (using our notion of pps_ω-definability).

Conservative functional clones are classified as follows. A particularly simple functional

clone is the clone generated by the disequality relation. A counting CSP derived from this clone is trivial to solve exactly, as the partition function factorises. We say that functions from this clone are of "product form". Our main result (Theorem 16) is that any clone that contains a function F that is not of product form necessarily contains the binary relation IMP = $\{(0,0), (0,1), (1,1)\}$. This has important complexity-theoretic consequences, which will be discussed presently. Furthermore, (also Theorem 16), any non-trivial clone that contains a function F that is not log-supermodular actually contains all functions. Therefore a large part of the functional clone landscape is very simple. In particular, we have a complete understanding of the clones below the clone generated by IMP and of the clones above LSM. The complexity of the landscape of functional clones is thus sandwiched between the clone generated by IMP and the clone LSM.

In order to derive complexity-theoretic consequences (see Theorem 19), we also present an efficient version of pps_ω-definability, and a corresponding notion of functional clone. The complexity-theoretic consequences are the third contribution of the paper. In order to describe these, we need a quick digression into the complexity of approximate counting. The complexity class $\#\text{RH}\Pi_1$ of counting problems was introduced by Dyer, Goldberg, Greenhill and Jerrum [20] as a means to classify a wide class of approximate counting problems that were previously of indeterminate computational complexity. The problems in $\#\text{RH}\Pi_1$ are those that can be expressed in terms of counting the number of models of a logical formula from a certain syntactically restricted class. The complexity class $\#\text{RH}\Pi_1$ has a completeness class (with respect to approximation-preserving "AP-reductions") which includes a wide and ever-increasing range of natural counting problems, including: independent sets in a bipartite graph, downsets in a partial order, configurations in the Widom-Rowlinson model (all [20]) and stable matchings [12]. Either all of these problems admit a Fully Polynomial Randomised Approximation Scheme (FPRAS), or none do. The latter is conjectured. A typical complete problem in this class is #BIS, the problem of counting independent sets in a bipartite graph.

Our complexity-theoretic results are presented in §10. As noted above, $\#\mathsf{CSP}(\mathcal{F})$ is computationally easy if every function in $\mathcal{F}$ is of product form. We show that, in every other (conservative) case, it is as difficult to approximate as #BIS. If, in addition, $\mathcal{F}$ contains a function F which is not log-supermodular, then the counting problem $\#\mathsf{CSP}(\mathcal{F})$ turns out to be universal for Boolean counting CSPs and hence is provably NP-hard to approximate. As immediate corollaries, we recover existing results concerning the complexity of computing the partition function of the Ising model [21].

Given the above discussion, one might speculate that the IMP-clone and LSM are the same. In fact, they are not. In §11, we examine the classes $\langle \mathsf{LSM}_k \rangle$ generated by lsm functions of arity at most k. We show that $\langle \mathsf{LSM}_3 \rangle$ is equal to the IMP-clone, but we give a proof that $\langle \mathsf{LSM}_3 \rangle$ is strictly contained in $\langle \mathsf{LSM}_4 \rangle$. This mirrors the situation for VCSPs, where binary submodular functions can express all ternary submodular functions but not all arity 4 submodular functions [36]. However, we do not know whether there is a fixed k such that $\mathsf{LSM} = \langle \mathsf{LSM}_k \rangle$. We conjecture that this is not the case. If $\mathsf{LSM} = \langle \mathsf{LSM}_k \rangle$ for some k, there would still remain the question of whether LSM is finitely generated, i.e., whether it is the functional clone generated by some finite set of functions $\mathcal{F}$. We conjecture the opposite, that there is no such $\mathcal{F}$.

Finally, in §12 and §13, we step outside the conservative case, and study functional

clones in which only restricted classes of unary functions are available. As might be expected, this yields a richer structure of functional clones, including one that corresponds to the ferromagnetic Ising model with a consistent field, a problem that is tractable in the FPRAS sense [25]. We also exhibit in this setting two clones that are provably incomparable with respect to inclusion, even though their corresponding counting CSPs are related by approximation-preserving reducibility. These counting CSPs have natural interpretations as (a) evaluating the weight enumerator of a binary code, and (b) counting independent sets in a bipartite graph. This example shows that, in demonstrating intractability, it may sometimes be necessary to use reductions that go beyond pps_ω-definability.

Although we focus on approximation of the partition functions of (weighted) #CSPs in this paper, there is, of course, an extensive literature on exact evaluation. See, for example, the recent survey of Chen [13].

2 Functional clones

As usual, we denote the natural numbers by $\mathbb{N}$, the real numbers by $\mathbb{R}$ and the complex numbers by $\mathbb{C}$. For $n \in \mathbb{N}$, we denote the set $\{1, 2, \ldots, n\}$ by $[n]$.

Let $(\mathscr{C}, +, \times)$ be any subsemiring of $(\mathbb{C}, +, \times)$, and let D be a finite domain. For $n \in \mathbb{N}$, denote by $\mathcal{U}_n$ the set of all functions $D^n \to \mathscr{C}$; also denote by $\mathcal{U} = \mathcal{U}_0 \cup \mathcal{U}_1 \cup \mathcal{U}_2 \cup \cdots$ the set of functions of all arities. Note that we do not specify the domain, which we take to be understood from the context, in this notation. Suppose $\mathcal{F} \subseteq \mathcal{U}$ is some collection of functions, $V = \{v_1, \ldots, v_n\}$ is a set of variables and $\boldsymbol{x} : \{v_1, \ldots, v_n\} \to D$ is an assignment to those variables. An atomic formula has the form $\varphi = G(v_{i_1}, \ldots, v_{i_a})$ where $G \in \mathcal{F}$, $a = a(G)$ is the arity of G, and $(v_{i_1}, v_{i_2}, \ldots, v_{i_a}) \in V^a$ is a scope. Note that repeated variables are allowed. The function $F_\varphi : D^n \to \mathscr{C}$ represented by the atomic formula $\varphi = G(v_{i_1}, \ldots, v_{i_a})$ is just

$$F_\varphi(\boldsymbol{x}) = G(\boldsymbol{x}(v_{i_1}), \ldots, \boldsymbol{x}(v_{i_a})) = G(x_{i_1}, \ldots, x_{i_a}),$$

where from now on we write $x_j = \boldsymbol{x}(v_j)$.

A pps-formula ("primitive product summation formula") is a summation[1] of a product of atomic formulas. A pps-formula ψ in variables $V \subseteq V' = \{v_1, \ldots, v_{n+m}\}$ over $\mathcal{F}$ has the form

(1)
$$\psi = \sum_{v_{n+1}, \ldots, v_{n+m}} \prod_{j=1}^{s} \varphi_j,$$

where φ_j are all atomic formulas over $\mathcal{F}$ in the variables V'. (The variables V are free, and the others, $V' \setminus V$, are bound.) The formula ψ specifies a function $F_\psi : D^n \to \mathscr{C}$ in the following way:

(2)
$$F_\psi(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in D^m} \prod_{j=1}^{s} F_{\varphi_j}(\boldsymbol{x}, \boldsymbol{y}),$$

---

[1]To avoid ambiguity, we will try to use "summation" of functions only with the meaning given here. Sums of different functions will be referred to as "addition".

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are assignments $V \to D$, $V' \setminus V \to D$. The functional clone $\langle \mathcal{F} \rangle$ generated by $\mathcal{F}$ is the set of all functions in $\mathcal{U}$ that can be represented by a pps-formula over $\mathcal{F} \cup \{\text{EQ}\}$ where EQ is the binary equality function defined by $\text{EQ}(x, x) = 1$ and $\text{EQ}(x, y) = 0$ for $x \neq y$. We refer to the pps-formula as an *implementation* of the function.

Since pps-formulas are defined using sums of products (with just one level of each), we need to check that functions that are pps-definable in terms of functions that are themselves pps-definable over $\mathcal{F}$ are actually directly pps-definable over $\mathcal{F}$. The following lemma ensures that this is the case.

Lemma 1. *If $G \in \langle \mathcal{F} \rangle$ then $\langle \mathcal{F}, G \rangle = \langle \mathcal{F} \rangle$.*

Note that, to simplify notation, we write $\langle \mathcal{F}, G \rangle$ in place of the more correct $\langle \mathcal{F} \cup \{G\} \rangle$. More generally, we shall often drop set-brackets, replace the union symbol $\cup$ by a comma, and confuse a singleton set with the element it contains.

Proof of Lemma 1. Let $\mathcal{F}' = \mathcal{F} \cup \{\text{EQ}\}$. Suppose that ψ is a pps-formula over $\mathcal{F}' \cup \{G\}$ given by

$$(3) \qquad \psi = \sum_{v_{n+1}, \ldots, v_{n+m}} \left(\prod_{i=1}^{r} \varphi_i \right) \left(\prod_{j=1}^{s} \chi_j \right),$$

where $\{\varphi_i\}$ are atomic $\mathcal{F}'$-formulas and $\{\chi_j\}$ are atomic G-formulas in the variables V'. Then

$$(4) \qquad F_\psi(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in D^m} \left(\prod_{i=1}^{r} F_{\varphi_i}(\boldsymbol{x}, \boldsymbol{y}) \right) \left(\prod_{j=1}^{s} F_{\chi_j}(\boldsymbol{x}, \boldsymbol{y}) \right),$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are assignments $\boldsymbol{x} : \{v_1, \ldots, v_n\} \to D$ and $\boldsymbol{y} : \{v_{n+1}, \ldots, v_{n+m}\} \to D$. Now, since G is pps-definable over $\mathcal{F}'$, and each χ_j is an atomic G-formula in the variables V', we can write each χ_j as

$$\chi_j = \sum_{v_{\nu_j+1}, \ldots, v_{\nu_j+\ell}} \prod_{k=1}^{t} \varphi_{j,k},$$

where ℓ is the number of bound variables used in the definition of χ_j (ℓ is independent of j), $\nu_j = n + m + (j-1)\ell$ is the number of free variables plus the number of bound variables that are "used up" by $\chi_1, \ldots, \chi_{j-1}$, and each $\varphi_{j,k}$ is an atomic $\mathcal{F}'$-formula over the variables $V' \cup \{v_{\nu_j+1}, \ldots, v_{\nu_j+\ell}\}$. We get

$$F_\psi(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in D^m} \left(\prod_{i=1}^{r} F_{\varphi_i}(\boldsymbol{x}, \boldsymbol{y}) \right) \left(\prod_{j=1}^{s} \sum_{\boldsymbol{z}^j \in D^\ell} \prod_{k=1}^{t} F_{\varphi_{j,k}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}^j) \right)$$

$$= \sum_{\boldsymbol{y} \in D^m} \sum_{\boldsymbol{z}^1 \in D^\ell} \cdots \sum_{\boldsymbol{z}^s \in D^\ell} \left(\prod_{i=1}^{r} F_{\varphi_i}(\boldsymbol{x}, \boldsymbol{y}) \right) \left(\prod_{j=1}^{s} \prod_{k=1}^{t} F_{\varphi_{j,k}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}^j) \right),$$

where each $\boldsymbol{z}^j$ is an assignment $\boldsymbol{z}^j : \{v_{\nu_j+1}, \ldots, v_{\nu_j+\ell}\} \to D$. So

$$\varphi = \sum_{v_{n+1},\ldots,v_{\nu_s+\ell}} \left(\prod_{i=1}^{r} \varphi_i \right) \left(\prod_{j=1}^{s} \prod_{k=1}^{t} \varphi_{j,k} \right)$$

is a pps-formula over $\mathcal{F}'$ for the function F_ψ. □

To extend the notion of definability, we allow limits as follows. We say that an a-ary function $F \in \mathcal{U}$ is pps$_\omega$-definable over $\mathcal{F}$ if there exists a finite subset S_F of $\mathcal{F}$, such that, for every $\varepsilon > 0$, there exists an a-ary function $\widehat{F}$, pps-definable over S_F, such that

$$\|\widehat{F} - F\|_\infty = \max_{\boldsymbol{x} \in D^a} |\widehat{F}(\boldsymbol{x}) - F(\boldsymbol{x})| < \varepsilon.$$

Denote the set of functions that are pps$_\omega$-definable over $\mathcal{F} \cup \{\mathrm{EQ}\}$ by $\langle \mathcal{F} \rangle_\omega$; we call this the pps$_\omega$-*definable functional clone* generated by $\mathcal{F}$. Observe that functions in $\langle \mathcal{F} \rangle_\omega$ are determined only by finite subsets of $\mathcal{F}$. This feature will be important when we come to define a computationally efficient version of the the limiting operation, but ultimately is not restrictive since $\#\mathsf{CSP}(\mathcal{F})$ is defined only for finite sets of functions $\mathcal{F}$. Also observe that some functions taking values outside $\mathscr{C}$ may be the limit of functions pps-definable over $\mathcal{F}$. But they are not pps$_\omega$-definable, since the function values of the limit must be in $\mathscr{C}$. The domain $\mathscr{C}$ of the universal class of functions $\mathcal{U}$ in operation at any time will be clear from the context.

The following lemma is an analogue of Lemma 1.

Lemma 2. *If $G \in \langle \mathcal{F} \rangle_\omega$ then $\langle \mathcal{F}, G \rangle_\omega = \langle \mathcal{F} \rangle_\omega$.*

Proof. Let $\mathcal{F}' = \mathcal{F} \cup \{\mathrm{EQ}\}$. Suppose that H is an a-ary function in $\langle \mathcal{F}, G \rangle_\omega$. Let S_H be a finite subset of $\mathcal{F}' \cup \{G\}$ such that the following is true: Given $\varepsilon > 0$, there exists an a-ary function $\widehat{H}$, pps-definable over S_H, such that $\|\widehat{H} - H\|_\infty < \varepsilon/2$. Let ψ be a pps-formula over S_H representing $\widehat{H}$. For any function $\widehat{G}$ with the same arity as G, denote by $\psi[G{:=}\widehat{G}]$ the formula obtained from ψ by replacing all occurrences of G by $\widehat{G}$. By continuity of the operators of pps-formulas, we know there exists $\delta > 0$ such that, for every function $\widehat{G}$ of the same arity as G, $\|\widehat{G} - G\|_\infty < \delta$ implies

$$\|F_{\psi[G:=\widehat{G}]} - F_\psi\|_\infty < \varepsilon/2.$$

This claim will be explicitly quantified in the proof of Lemma 4, but we don't need so much detail here. Of course, $\widehat{H} = F_\psi$ so for each such $\widehat{G}$ we have $\|F_{\psi[G:=\widehat{G}]} - \widehat{H}\|_\infty < \varepsilon/2$. Now let S_G be the finite subset of $\mathcal{F}'$ used to show that G is pps$_\omega$-definable over $\mathcal{F}'$. Let $S = S_G \cup S_H \setminus \{G\} \subseteq \mathcal{F}'$. Choose a function $\widehat{G}$, pps-definable over S_G, satisfying $\|\widehat{G} - G\|_\infty < \delta$. Notice that $\widehat{G} \in \langle S \rangle$ and $F_\psi \in \langle S, G \rangle$ so $F_{\psi[G:=\widehat{G}]} \in \langle S \rangle$ (by Lemma 1), and

$$\|F_{\psi[G:=\widehat{G}]} - H\|_\infty \leq \|F_{\psi[G:=\widehat{G}]} - \widehat{H}\|_\infty + \|\widehat{H} - H\|_\infty < \varepsilon.$$

Since $\varepsilon > 0$ is arbitrary, and $S \subseteq \mathcal{F}'$ is finite, we conclude that $H \in \langle \mathcal{F} \rangle_\omega$. □

That completes the setup for expressibility. In order to deduce complexity results, we need an efficient version of $\langle \mathcal{F} \rangle_\omega$. We say that a function F is *efficiently* pps_ω-definable over $\mathcal{F}$ if there is a finite subset S_F of $\mathcal{F}$, and a TM $\mathcal{M}_{F,S_F}$ with the following property: on input $\varepsilon > 0$, $\mathcal{M}_{F,S_F}$ computes a pps-formula ψ over S_F such that F_ψ has the same arity as F and $\|F_\psi - F\|_\infty < \varepsilon$. The running time of $\mathcal{M}_{F,S_F}$ is at most a polynomial in $\log \varepsilon^{-1}$. Denote the set of functions in $\mathcal{U}$ that are efficiently pps_ω-definable over $\mathcal{F} \cup \{\mathrm{EQ}\}$ by $\langle \mathcal{F} \rangle_{\omega,p}$; we call this the *efficient* pps_ω-*definable functional clone* generated by $\mathcal{F}$,

The following useful observation is immediate from the definition of $\langle \mathcal{F} \rangle_{\omega,p}$.

Observation 3. Suppose $F \in \langle \mathcal{F} \rangle_{\omega,p}$ (or $F \in \langle \mathcal{F} \rangle_\omega$). Then there is a finite subset S_F of $\mathcal{F}$ such that $F \in \langle S_F \rangle_{\omega,p}$ (resp. $F \in \langle S_F \rangle_\omega$).

The following lemma is an analogue of Lemma 2 .

Lemma 4. *If* $G \in \langle \mathcal{F} \rangle_{\omega,p}$ *then* $\langle \mathcal{F}, G \rangle_{\omega,p} = \langle \mathcal{F} \rangle_{\omega,p}$.

Proof. Let $\mathcal{F}' = \mathcal{F} \cup \{\mathrm{EQ}\}$. Suppose H is an a-ary function in $\langle \mathcal{F}', G \rangle_{\omega,p}$. Our goal is to specify a finite subset S of $\mathcal{F}'$ and to construct a TM $M_{H,S}$ with the following property: on input $\varepsilon > 0$, $M_{H,S}$ should compute an a-ary pps-formula φ over S such that $\|F_\varphi - H\|_\infty < \varepsilon$. The running time of $M_{H,S}$ should be at most a polynomial in $\log \varepsilon^{-1}$.

Let S_H be the finite subset of $\mathcal{F}' \cup \{G\}$ from the efficient pps_ω-definition of H over $\mathcal{F}' \cup \{G\}$. Given an input $\varepsilon/2$, the TM M_{H,S_H} computes an a-ary pps-formula ψ over S_H such that $\|F_\psi - H\|_\infty < \varepsilon/2$. Write ψ as in Equation (3) so F_ψ is written as in Equation (4). Suppose that, for $j \in [s]$ and $\boldsymbol{y} \in \{0,1\}^m$, $\delta_{j,\boldsymbol{y}}(\boldsymbol{x})$ is a function of $\boldsymbol{x}$. Consider the expression

$$\Upsilon(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in D^m} \left(\prod_{i=1}^r F_{\varphi_i}(\boldsymbol{x},\boldsymbol{y}) \right) \left(\prod_{j=1}^s (F_{\chi_j}(\boldsymbol{x},\boldsymbol{y}) + \delta_{j,\boldsymbol{y}}(\boldsymbol{x})) \right)$$
$$- \sum_{\boldsymbol{y} \in D^m} \left(\prod_{i=1}^r F_{\varphi_i}(\boldsymbol{x},\boldsymbol{y}) \right) \left(\prod_{j=1}^s F_{\chi_j}(\boldsymbol{x},\boldsymbol{y}) \right),$$

which can be expanded as

$$\Upsilon(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in D^m} \sum_{\emptyset \subset T \subseteq [s]} C_{\boldsymbol{y},T}(\boldsymbol{x}) \prod_{j \in T} \delta_{j,\boldsymbol{y}}(\boldsymbol{x}),$$

where

$$C_{\boldsymbol{y},T}(\boldsymbol{x}) = \prod_{i=1}^r F_{\varphi_i}(\boldsymbol{x},\boldsymbol{y}) \prod_{j \in [s] \setminus T} F_{\chi_j}(\boldsymbol{x},\boldsymbol{y}).$$

Let $C = \max_{\boldsymbol{x},\boldsymbol{y},T} |C_{\boldsymbol{y},T}(\boldsymbol{x})|$ and let $\delta = \varepsilon 2^{-(s+1)} |D|^{-m} C^{-1} < 1$.

Now let S_G be the finite subset of $\mathcal{F}'$ used to show that G is efficiently pps_ω-definable over $\mathcal{F}'$. Given the input δ, the TM M_{G,S_G} computes a pps-formula $\widehat{\chi}$ over S_G representing a function $F_{\widehat{\chi}}$ with the same arity as G such that $\|F_{\widehat{\chi}} - G\|_\infty < \delta$. Since each χ_j is an atomic G-formula in the variables V', we may appropriately name the variables of $\widehat{\chi}$ to obtain a pps-formula $\widehat{\chi}_j$ over S_G such that $\|F_{\widehat{\chi}_j} - F_{\chi_j}\|_\infty < \delta$.

For $\boldsymbol{y} \in D^m$, let $\delta_{j,\boldsymbol{y}}(\boldsymbol{x}) = F_{\widehat{\chi}_j}(\boldsymbol{x},\boldsymbol{y}) - F_{\chi_j}(\boldsymbol{x},\boldsymbol{y})$ and note that $|\delta_{j,\boldsymbol{y}}(\boldsymbol{x})| \le \delta$. Let $S = S_G \cup S_H \setminus \{G\} \subseteq \mathcal{F}'$. Let ψ' be the formula over S formed from ψ by substituting

each occurrence of χ_j with $\widehat{\chi}_j$. Convert ψ' to a pps-formula φ as in Lemma 1. From the calculation above,

$$
\begin{aligned}
\|F_\varphi - F_\psi\|_\infty &= \|F_{\psi'} - F_\psi\|_\infty \\
&= \max_{\boldsymbol{x}} |F_{\psi'}(\boldsymbol{x}) - F_\psi(\boldsymbol{x})| \\
&= \max_{\boldsymbol{x}} |\Upsilon(\boldsymbol{x})| \\
&\leq 2^s |D|^m C\delta = \varepsilon/2.
\end{aligned}
$$

Note that

$$
\|F_\varphi - H\|_\infty \leq \|F_\varphi - F_\psi\|_\infty + \|F_\psi - H\|_\infty < \varepsilon.
$$

Thus, the formula φ is an appropriate output for our TM $M_{H,S}$.

Finally, let us check how long the computation takes. The running time of M_{H,S_H} is at most $\mathrm{poly}(\log \varepsilon^{-1})$. Since this machine outputs the formula ψ, we conclude that m and r and s are bounded from above by polynomials in $\log \varepsilon^{-1}$. Let Δ be the ceiling of the maximum absolute value of any function in S_H. Note that $C \leq \Delta^{r+s}$. The running time of M_{G,S_G} is at most $\mathrm{poly}(\log(\delta^{-1}))$, which is at most a polynomial in $m + s + \log(C) + \log(\varepsilon^{-1})$ which is at most a polynomial in $\log(\varepsilon^{-1})$. Finally, the direct manipulation of the formulas that we did (renaming variables from $\widehat{\chi}$ to obtain $\widehat{\chi}_j$ and producing the pps-formula φ from ψ and the $\widehat{\chi}_j$ formulas) takes time at most polynomial in the size of ψ and $\widehat{\chi}$, which is at most a polynomial in $\log(\varepsilon^{-1})$. □

Lemma 4 may have wider applications in the study of approximate counting problems. Often, approximation-preserving reductions between counting problems are complicated to describe and difficult to analyse, owing to the need to track error estimates. Lemma 4 suggests breaking the reduction into smaller steps, and analysing each of them independently. This assumes, of course, that the reductions are pps$_\omega$-definable, but that often seems to be the case in practice.

3　Relational clones and nonnegative functions

A function $F \in \mathcal{U}$ is *Boolean*[2] if its range is a subset of $\{0,1\}$. Then F encodes a relation R as follows: $\boldsymbol{x}$ is in the relation R iff $F(\boldsymbol{x}) = 1$. We will not distinguish between relations and the Boolean functions that define them. Suppose that $\mathcal{R} \subseteq \mathcal{U}$ is a set of relations/Boolean functions. A pp-formula over $\mathcal{R}$ is an existentially quantified product of atomic formulas (this is called an $\exists \mathrm{CNF}(\mathcal{R})$-formula in [16]). More precisely, a pp-formula ψ over $\mathcal{R}$ in variables $V' = \{v_1, \ldots, v_{n+m}\}$ has the form

$$
\psi = \exists\, v_{n+1}, \ldots, v_{n+m} \bigwedge_{j=1}^{s} \varphi_j,
$$

---

[2]Note that "Boolean" applies to the codomain here, not the domain. All of the functions that we consider from now on have Boolean domains. This usage of "Boolean function" is unfortunate, but is well established in the literature. When the range is not a subset of $\{0,1\}$ we emphasise this fact by referring to the function as a "pseudo-Boolean" function.

where φ_j are all atomic formulas over $\mathcal{R}$ in the variables V'. As before, the variables $V = \{v_1, \ldots, v_n\}$ are called "free", and the others, $V' \setminus V$, are called "bound". The formula ψ specifies a Boolean function $R_\psi : D^n \to \{0,1\}$ in the following way. $R_\psi(\boldsymbol{x}) = 1$ if there is a vector $\boldsymbol{y} \in D^m$ such that $\bigwedge_{j=1}^s R_{\varphi_j}(\boldsymbol{x}, \boldsymbol{y})$ evaluates to "1", where $\boldsymbol{x}$ and $\boldsymbol{y}$ are assignments $\boldsymbol{x} : \{v_1, \ldots, v_n\} \to D$ and $\boldsymbol{y} : \{v_{n+1}, \ldots, v_{n+m}\} \to D$; $R_\psi(\boldsymbol{x}) = 0$ otherwise. We call the pp-formula an *implementation* of R_ψ.

A *relational clone* (often called a "co-clone") is a set of relations containing the equality relation and closed under finite Cartesian products, projections, and identification of variables. A *basis* [16] for the relational clone I is a set $\mathcal{R}$ of Boolean relations such that the relations in I are exactly the relations that can be implemented with a pp-formula over $\mathcal{R}$. Every relational clone has such a basis.

For every set $\mathcal{R}$ of Boolean relations, let $\langle \mathcal{R} \rangle_{\mathrm{R}}$ denote the set of relations that can be represented by a pp-formula over $\mathcal{R} \cup \{\mathrm{EQ}\}$. It is well-known that if $R \in \langle \mathcal{R} \rangle_{\mathrm{R}}$ then $\langle \mathcal{R} \cup \{R\} \rangle_{\mathrm{R}} = \langle \mathcal{R} \rangle_{\mathrm{R}}$ (This can be proved similarly to the proof of Lemma 1.) Thus, $\langle \mathcal{R} \rangle_{\mathrm{R}}$ is in fact a relational clone with basis $\mathcal{R}$.

A basis $\mathcal{R}$ for a relational clone $\langle \mathcal{R} \rangle_{\mathrm{R}}$ is called a "plain basis" [16, Definition 1] if every member of $\langle \mathcal{R} \rangle_{\mathrm{R}}$ is definable by a $\mathrm{CNF}(\mathcal{R})$-formula (a pp-formula over $\mathcal{R}$ with no $\exists$).

Pseudo-Boolean functions [5] are defined on the Boolean domain $D = \{0,1\}$, and have codomain $\mathscr{C} = \mathbb{R}$, the real numbers. For $n \in \mathbb{N}$, denote by $\mathscr{B}_n$ the set of all functions $\{0,1\}^n \to \mathbb{R}$, and denote the set of functions of all arities by $\mathscr{B} = \mathscr{B}_0 \cup \mathscr{B}_1 \cup \mathscr{B}_2 \cup \cdots$. Note that any tuple $\boldsymbol{x} \in \{0,1\}^n$ is the indicator function of a subset of $[n]$. We write $|\boldsymbol{x}|$ for the cardinality of this set, i.e. $|\boldsymbol{x}| = \sum_{j=1}^n x_j$.

For most of this paper, we restrict attention to the codomain $\mathscr{C} = \mathbb{R}^{\geq 0}$ of nonnegative real numbers. Then $\mathcal{B}_n$ is the set given by replacing $\mathbb{R}$ by $\mathbb{R}^{\geq 0}$ in the definition of $\mathscr{B}_n$, and then $\mathcal{B}$ is defined analogously to $\mathscr{B}$. We will also need to consider the *permissive* functions in $\mathcal{B}$. These are functions which are positive everywhere, so the codomain $\mathscr{C} = \mathbb{R}^{>0}$, the positive real numbers. Thus $\mathcal{B}_n^{>0}$ and $\mathcal{B}^{>0}$ are given by replacing $\mathbb{R}$ by $\mathbb{R}^{>0}$ in the definitions of $\mathscr{B}_n$ and $\mathscr{B}$.

The advantage of working with the Boolean domain is (i) it has a well-developed theory of relational clones, and (ii) the concept of a log-supermodular function exists (see §4). As explained in the introduction, the advantage of working with nonnegative real numbers is that we disallow cancellation, and potentially obtain a more nuanced expressibility/complexity landscape.

Given a function $F \in \mathcal{B}$, let R_F be the function corresponding to the relation underlying F. That is, $R_F(\boldsymbol{x}) = 0$ if $F(\boldsymbol{x}) = 0$ and $R_F(\boldsymbol{x}) = 1$ if $F(\boldsymbol{x}) > 0$. The following straightforward lemma will be useful.

Lemma 5. *Suppose $\mathcal{F} \subseteq \mathcal{B}$. Then*

$$\langle \{R_F \mid F \in \mathcal{F}\} \rangle_{\mathrm{R}} = \{R_F \mid F \in \langle \mathcal{F} \rangle\}.$$

Proof. Let $\mathcal{F}$ be a subset of $\mathcal{B}$. First, we must show that, for any $R \in \langle \{R_F \mid F \in \mathcal{F}\} \rangle_{\mathrm{R}}$, R is in $\{R_F \mid F \in \langle \mathcal{F} \rangle\}$.

Let ψ be the pp-formula over $\{R_F \mid F \in \mathcal{F}\} \cup \{\mathrm{EQ}\}$ that is used to represent R. Write

ψ as

$$\psi = \exists\, v_{n+1}, \ldots, v_{n+m} \bigwedge_{j=1}^{s} R_{F_j}(v_{i(j,1)}, \ldots, v_{i(j,a_j)}),$$

where F_j is an arity-a_j function in $\mathcal{F} \cup \{\text{EQ}\}$, and the index function $i(\cdot, \cdot)$ picks out an index in the range $[1, n+m]$, and hence a variable from $V' = \{v_1, \ldots, v_{n+m}\}$. Let ψ' be the pps-formula over $\mathcal{F} \cup \{\text{EQ}\}$ given by

$$\psi' = \sum_{v_{n+1}, \ldots, v_{n+m}} \prod_{j=1}^{s} F_j(v_{i(j,1)}, \ldots, v_{i(j,a_j)}).$$

Let $F' = F_{\psi'}$. Note that $F' \in \langle \mathcal{F} \rangle$ and that $R_{F'} = R$.

By reversing this construction, we can show that, for any $R \in \{R_F \mid F \in \langle \mathcal{F} \rangle\}$, R is in $\langle \{R_F \mid F \in \mathcal{F}\} \rangle_{\mathrm{R}}$. $\qquad\square$

4   Log-supermodular functions

A function $F \in \mathcal{B}_n$ is *log-supermodular* (lsm) if $F(\boldsymbol{x} \vee \boldsymbol{y})F(\boldsymbol{x} \wedge \boldsymbol{y}) \geq F(\boldsymbol{x})F(\boldsymbol{y})$ for all $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$. The terminology is justified by the observation that $F \in \mathcal{B}_n^{>0}$ is lsm if and only if $f = \log F$ is *supermodular*. We denote by $\mathsf{LSM} \subset \mathcal{B}$ the class of all lsm functions. The second part of our main result (Theorem 16) says that, in terms of expressibility, everything of interest takes place within the class LSM. Consequently, in §11, we will investigate the internal structure of LSM.

Note that $\mathcal{B}_0, \mathcal{B}_1 \subset \mathsf{LSM}$, since log-supermodularity is trivial for nullary or unary functions, and hence the class LSM is conservative. And it fits naturally into our study of expressibility because of the following closure property: functions that are pps_ω-definable from lsm functions are lsm. The non-trivial step in showing this is encapsulated in the following lemma. It is a special case of the Ahlswede-Daykin "four functions" theorem [1]. However [1] is a much stronger result than is required, so we give an easier proof, using an argument similar to the base case of the induction in [1].

Lemma 6. *If $G \in \mathcal{B}_{n+m}$, let $G' \in \mathcal{B}_n$ be defined by $G'(\boldsymbol{x}) = \sum_{\boldsymbol{z} \in \{0,1\}^m} G(\boldsymbol{x}, \boldsymbol{z})$. Then $G \in \mathsf{LSM}$ implies $G' \in \mathsf{LSM}$.*

Proof. By symmetry and induction, it suffices to consider summation on the last variable. Thus, let $(\boldsymbol{x}, x_{n+1}), (\boldsymbol{y}, y_{n+1}) \in \{0,1\}^{n+1}$, and let

$$\alpha_z = G(\boldsymbol{x}, z), \ \beta_z = G(\boldsymbol{y}, z), \ \gamma_z = G(\boldsymbol{x} \vee \boldsymbol{y}, z), \ \delta_z = G(\boldsymbol{x} \wedge \boldsymbol{y}, z) \quad (z \in \{0,1\}).$$

Then we must show that $G \in \mathsf{LSM}$ implies

$$(\alpha_0 + \alpha_1)(\beta_0 + \beta_1) \leq (\gamma_0 + \gamma_1)(\delta_0 + \delta_1),$$

which expands to

$$(5) \qquad \alpha_0\beta_0 + \alpha_1\beta_0 + \alpha_0\beta_1 + \alpha_1\beta_1 \leq \gamma_0\delta_0 + \gamma_0\delta_1 + \gamma_1\delta_0 + \gamma_1\delta_1.$$

Since $G \in \mathsf{LSM}$, we have the following four inequalities,

$$(6) \quad \alpha_0 \beta_0 \leq \gamma_0 \delta_0, \qquad (7) \quad \alpha_0 \beta_1 \leq \gamma_1 \delta_0, \qquad (8) \quad \alpha_1 \beta_0 \leq \gamma_1 \delta_0, \qquad (9) \quad \alpha_1 \beta_1 \leq \gamma_1 \delta_1.$$

We will complete the proof by showing that (6) to (9) imply (5) for arbitrary nonnegative real numbers $\alpha_z, \beta_z, \gamma_z, \delta_z$ ($z \in \{0, 1\}$).

Using (6) and (9), it follows that (5) is implied by

$$(10) \qquad \qquad \alpha_1 \beta_0 + \alpha_0 \beta_1 \ \leq \ \gamma_0 \delta_1 + \gamma_1 \delta_0.$$

Observe that $\gamma_1 \delta_0 = 0$ implies (10), since the left side is zero by (7) and (8). Thus we may assume $\gamma_1 \delta_0 > 0$.

Now, using (6) and (9) again, (10) is implied by

$$(11) \qquad \alpha_1 \beta_0 + \alpha_0 \beta_1 \ \leq \ \frac{(\alpha_0 \beta_0)(\alpha_1 \beta_1)}{\delta_0 \gamma_1} + \gamma_1 \delta_0 \ = \ \frac{(\alpha_1 \beta_0)(\alpha_0 \beta_1)}{\gamma_1 \delta_0} + \gamma_1 \delta_0.$$

Now (11) can be rewritten as

$$0 \ \leq \ (\gamma_1 \delta_0 - \alpha_1 \beta_0)(\gamma_1 \delta_0 - \alpha_0 \beta_1),$$

which is implied by (7) and (8). □

Lemma 7. *If $\mathcal{F} \subseteq \mathsf{LSM}$ then $\langle \mathcal{F} \rangle_\omega \subseteq \mathsf{LSM}$.*

Proof. We just need to show that each level in the definition of pps_ω-definable function preserves lsm: first that every atomic formula over $\mathcal{F} \cup \{\mathrm{EQ}\}$ defines an lsm function, then that a product of lsm functions is lsm, then that a summation of an lsm function is lsm, and finally that a limit of lsm functions is lsm. As we shall see below, only the third step is non-trivial, and it is covered by Lemma 6.

First, note that the EQ is lsm, so every function in $\mathcal{F} \cup \{\mathrm{EQ}\}$ is lsm, An atomic formula $\varphi = G(v_{i_1}, \ldots, v_{i_a})$ defines a function $F_\varphi(\boldsymbol{x}) = G(x_{i_1}, \ldots, x_{i_a})$ which is lsm:

$$
\begin{aligned}
F_\varphi(\boldsymbol{x} \vee \boldsymbol{y}) F_\varphi(\boldsymbol{x} \wedge \boldsymbol{y}) &= G(x_{i_1} \vee y_{i_1}, \ldots, x_{i_a} \vee y_{i_a}) \, G(x_{i_1} \wedge y_{i_1}, \ldots, x_{i_a} \wedge y_{i_a}) \\
&\geq G(x_{i_1}, \ldots, x_{i_a}) \, G(y_{i_1}, \ldots, y_{i_a}) \\
&= F_\varphi(\boldsymbol{x}) F_\varphi(\boldsymbol{y}).
\end{aligned}
$$

Note that we do not need to assume that $i_1, \ldots, i_a$ are all distinct.

It is immediate that the product of two lsm functions (and hence the product of any number) is lsm. Thus the product $\prod_{j=1}^{s} F_{\varphi_j}$ appearing in (2) is lsm. Then, by Lemma 6, the pps-definable function F_ψ in (2) is lsm.

Finally, we will show that any function that is approximated by lsm functions is lsm. Suppose that a function $F \in \mathcal{B}_n$ has the property that, for every $\varepsilon > 0$, there is an arity-n lsm function $\widehat{F}$ satisfying

$$\|\widehat{F} - F\|_\infty = \max_{\boldsymbol{x} \in \{0,1\}^a} |\widehat{F}(\boldsymbol{x}) - F(\boldsymbol{x})| < \varepsilon.$$

We wish to show that F is lsm. Let $F_{\max} = \max_{\boldsymbol{x}} F(\boldsymbol{x})$. Suppose for contradiction that F is not lsm, so there is a $\delta > 0$ and $\boldsymbol{x}, \boldsymbol{y} \in \{0, 1\}^n$ such that

$$F(\boldsymbol{x} \vee \boldsymbol{y}) F(\boldsymbol{x} \wedge \boldsymbol{y}) \leq F(\boldsymbol{x}) F(\boldsymbol{y}) - \delta.$$

Let $\varepsilon > 0$ be sufficiently small that $\varepsilon \max(F_{\max}, 1)$ is tiny compared to δ. Then

$$
\begin{aligned}
\widehat{F}(\boldsymbol{x} \vee \boldsymbol{y})\widehat{F}(\boldsymbol{x} \wedge \boldsymbol{y}) &\leq (F(\boldsymbol{x} \vee \boldsymbol{y}) + \varepsilon)(F(\boldsymbol{x} \wedge \boldsymbol{y}) + \varepsilon) \\
&\leq F(\boldsymbol{x} \vee \boldsymbol{y})F(\boldsymbol{x} \wedge \boldsymbol{y}) + 2\varepsilon F_{\max} + \varepsilon^2 \\
&\leq F(\boldsymbol{x})F(\boldsymbol{y}) - \delta + 2\varepsilon F_{\max} + \varepsilon^2 \\
&\leq (\widehat{F}(\boldsymbol{x}) + \varepsilon)(\widehat{F}(\boldsymbol{y}) + \varepsilon) - \delta + 2\varepsilon F_{\max} + \varepsilon^2 \\
&\leq \widehat{F}(\boldsymbol{x})\widehat{F}(\boldsymbol{y}) + 2\varepsilon(F_{\max} + \varepsilon) - \delta + 2\varepsilon F_{\max} + 2\varepsilon^2 \\
&< \widehat{F}(\boldsymbol{x})\widehat{F}(\boldsymbol{y}),
\end{aligned}
$$

so $\widehat{F}$ is not lsm, giving a contradiction.   □

An important example of an lsm function is the 0,1-function "implies",

$$
\mathrm{IMP}(x, y) = \begin{cases} 0, & \text{if } (x, y) = (1, 0); \\ 1, & \text{otherwise.} \end{cases}
$$

We also think of this as a binary relation $\mathrm{IMP} = \{(0,0), (0,1), (1,1)\}$. Complexity-theoretic issues will be treated in detail in §10. However, it may be helpful to give a pointer here to the importance of IMP in the study of approximate counting problems.

The problem #BIS is that of counting independent sets in a bipartite graph. Dyer et al. [18] exhibited a class of counting problems, including #BIS, which are interreducible via approximation-preserving reductions. Further natural problems have been shown to lie in this class, providing compelling evidence that is of intermediate complexity between counting problems that are tractable (admit a polynomial-time approximation algorithm) and those that are NP-hard to approximate. We will see in due course (Theorem 19 and Proposition 37) that #BIS and #CSP(IMP) are interreducible via approximation-preserving reductions, and hence are of equivalent difficulty.

We know from Lemma 7 that $\langle \mathrm{IMP}, \mathcal{B}_1 \rangle_\omega \subseteq \mathsf{LSM}$, and one might ask whether this inclusion is strict. We will address this question in §11.

5   Pinnings and modular functions

Let δ_0 be the unary function with $\delta_0(0) = 1$ and $\delta_0(1) = 0$ and let δ_1 be the unary function with $\delta_1(0) = 0$ and $\delta_1(1) = 1$.

Let $S \subseteq [n]$, let $\boldsymbol{x}' = (x_j)_{j \notin S}$, and $\boldsymbol{x}'' = (x_j)_{j \in S}$, and partition $\boldsymbol{x} \in \{0,1\}^n$ as $(\boldsymbol{x}'; \boldsymbol{x}'')$. Then, if $F \in \mathcal{B}_n$, the function $F(\boldsymbol{x}'; \boldsymbol{c})$ given by setting $x''_j = c_j$ for constants $c_j \in \{0,1\}$ ($j \in S$) is a *pinning* of F. Note that we allow the empty pinning $S = \emptyset$, which is F itself, and the pinning of all variables $S = [n]$, which is a nullary function.

Clearly, every pinning of F is in $\langle F, \delta_0, \delta_1 \rangle$, since a constant $c \in \{0,1\}$ can be implemented using either δ_0 or δ_1, i.e. we add $\delta_c(x_i)$ to the constraint set. We will use the notation $i \leftarrow c$ to indicate that the ith variable has been pinned to c.

If $n \geq 2$ then a *2-pinning* of a function $F \in \mathcal{B}_n$ is a function $F(x_i, x_j; \boldsymbol{c})$ which pins *all but* 2 of the variables. Thus $[n] \setminus S = \{i, j\}$, where i and j are distinct indices in $[n]$, and $\boldsymbol{c} \in \{0,1\}^{n-2}$. Clearly, every 2-pinning of F is in $\langle F, \mathcal{B}_1 \rangle$, since it is a pinning of F.

We say that a function $F \in \mathcal{B}_n$ is log-modular if $F(\boldsymbol{x} \vee \boldsymbol{y})F(\boldsymbol{x} \wedge \boldsymbol{y}) = F(\boldsymbol{x})F(\boldsymbol{y})$ for all $\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n$.

It is a fact that LSM and the class of log-modular functions are closed under pinning. This is a consequence of the following lemma about 2-pinnings of lsm and log-modular functions. It is due, in essence, to Topkis [31], but we provide a short proof for completeness.

Lemma 8 (Topkis). *A function $F \in \mathcal{B}^{>0}$ is lsm iff every 2-pinning is lsm, and is log-modular iff every 2-pinning is log-modular.*

Proof. The necessity of the 2-pinning condition is obvious, but we must prove sufficiency. We need only show $F(\boldsymbol{x})F(\boldsymbol{y}) \leq F(\boldsymbol{x} \vee \boldsymbol{y})F(\boldsymbol{x} \wedge \boldsymbol{y})$ for $\boldsymbol{x}, \boldsymbol{y}$ such that $x_i \neq y_i$ $(i \in [n])$. All other cases follow from this. Note that log-supermodularity is preserved under arbitrary permutation of variables. Thus, if 0^r, 1^r denote r-tuples of 0's and 1's respectively, we must show that, for all $r, s > 0$ with $r + s = n$,

$$(12) \qquad F(0^r, 1^s)F(1^r, 0^s) \leq F(1^r, 1^s)F(0^r, 0^s).$$

We will prove this by induction, assuming it is true for all $r', s' > 0$ such that $r' + s' < n$. The base case, $r' = s' = 1$, is the 2-pinning assumption. If $r > 1$, then we have

$$(13) \qquad F(0^r, 1^s)F(1^{r-1}, 0^{s+1}) \leq F(1^{r-1}, 0, 1^s)F(0^r, 0^s),$$

by induction, after pinning the rth position to 0,

$$(14) \qquad F(1^{r-1}, 0, 1^s)F(1^r, 0^s) \leq F(1^r, 1^s)F(1^{r-1}, 0^{s+1})$$

by induction, after pinning the first $r - 1 > 0$ positions to 1.

Now, multiplying (13) and (14) gives (12) after cancellation, which is valid since F is permissive. If $r = 1$, we do not have the induction giving (14), so we use instead

$$(15) \qquad F(1, 0^s)F(0, 1, 0^{s-1}) \leq F(0, 0^s)F(1, 1, 0^{s-1}),$$

by induction, using the base case after pinning the last $s - 1$ positions to 0,

$$(16) \qquad F(1, 1, 0^{s-1})F(0, 1^s) \leq F(1, 1^s)F(0, 1, 0^{s-1}).$$

by induction, after pinning the second position to 1.

Now, multiplying (15) and (16) gives (12) after cancellation, completing the proof for log-supermodularity. The proof for log-modularity is identical, except that every "$\leq$" must be replaced by "$=$". $\qquad \square$

Remark 9. We have proved Lemma 8 only for permissive functions because, in fact, it is false more generally. Consider, for example, the function $F \in \mathcal{B}_4$ such that $F(1,1,0,0) = F(0,0,1,1) = 1$, $F(\boldsymbol{x}) = 0$ otherwise. It is easy to see that all the 2-pinnings F' of F have $F'(x, y) > 0$ for at most one $(x, y) \in \{0, 1\}^2$. It follows that every 2-pinning of F is log-modular. But F is not even lsm, since $F(1,1,0,0)F(0,0,1,1) > F(1,1,1,1)F(0,0,0,0)$.

6   Computable real numbers

Since we want to be able to derive computational results, we will now focus attention on functions whose co-domain is restricted to efficiently-computable real numbers. We will say that a real number is polynomial-time computable if the n most significant bits of its binary expansion can be computed in time polynomial in n. This is essentially the definition given in [27]. Let $\mathbb{R}^p$ denote the set of *nonnegative* real numbers that are polynomial-time computable. For $n \in \mathbb{N}$, denote by $\mathcal{B}_n^p$ the set of all functions $\{0,1\}^n \to \mathbb{R}^p$; also denote by $\mathcal{B}^p = \mathcal{B}_0^p \cup \mathcal{B}_1^p \cup \mathcal{B}_2^p \cup \cdots$ the set of functions of all arities.

Remark 10. As we have defined them, it is known that the polynomial-time computable numbers form a field [27], and hence a subsemiring $\mathscr{C}$ of $\mathbb{C}$, as we require. Thus, in our definitions, there is no difficulty with pps-definability. However, there could be a problem with pps$_\omega$-definability, since the limit of a sequence of polynomial-time computable reals may not be polynomial-time computable. Polynomial-time computability is ensured only by placing restrictions on speed of convergence. See [27, 32] for details. However, observe that our definition of efficient pps$_\omega$-definability avoids this difficulty entirely, by insisting that the limit of a sequence of reals will be permitted only if the limit is itself polynomial-time computable.

Remark 11. The polynomial-time computable real numbers are a proper subclass of the *efficiently approximable* real numbers, defined in [22]. (This fact can be deduced from [27].) We have made this restriction since it results in a more uniform treatment of limits when we discuss efficient pps$_\omega$-definability for functions in $\mathcal{B}^p$.

7   Binary functions

We begin the study of the conservative case in the simplest nontrivial situation. We consider the functional clones $\langle F, \mathcal{B}_1^p \rangle_{\omega,p}$, where F is a single binary function.

Recall that EQ is the binary relation EQ $= \{(0,0),(1,1)\}$. (We used the name "EQ" to denote the equivalent binary function, but it will do no harm to use the same symbol for the relation and the function.) Denote by OR, NEQ, and NAND the binary relations OR $= \{(0,1),(1,0),(1,1)\}$, NEQ $= \{(0,1),(1,0)\}$, and NAND $= \{(0,0),(0,1),(1,0)\}$. When we write a function $F \in \mathcal{B}_2$, we will identify the arguments by writing $F(x_1, x_2)$. We may represent F by a 2×2 matrix

$$M(F) \;=\; \begin{bmatrix} F(0,0) & F(0,1) \\ F(1,0) & F(1,1) \end{bmatrix} \;=\; \begin{bmatrix} f_{00} & f_{01} \\ f_{10} & f_{11} \end{bmatrix},$$

say, with rows indexed by $x_1 \in \{0,1\}$ and columns by $x_2 \in \{0,1\}$. We will assume $f_{01} \geq f_{10}$, since otherwise we may consider the function F^T, such that $F^T(x_1, x_2) = F(x_2, x_1)$, represented by the matrix $M(F)^T$. Clearly $\langle F^T \rangle = \langle F \rangle$.

If U is a unary function, we will write $U = (U(0), U(1)) = (u_0, u_1)$, say. Then we have

$$M\big(U(x_1)F(x_1,x_2)\big) = \begin{bmatrix} u_0 f_{00} & u_0 f_{01} \\ u_1 f_{10} & u_1 f_{11} \end{bmatrix}, \quad M\big(U(x_2)F(x_1,x_2)\big) = \begin{bmatrix} u_0 f_{00} & u_1 f_{01} \\ u_0 f_{10} & u_1 f_{11} \end{bmatrix},$$

where both $U(x_1)F(x_1, x_2)$ and $U(x_2)F(x_1, x_2)$ are clearly in $\langle F, U \rangle$.

If $F_1, F_2 \in \mathcal{B}_2$, then $M(F_1)M(F_2) = M(F)$, where $F \in \langle F_1, F_2 \rangle$ is such that

$$F(x_1, x_2) \;=\; \sum_{y=0}^{1} F_1(x_1, y)F_2(y, x_2).$$

Lemma 12. *Let $F \in \mathcal{B}_2^p$. Assuming $f_{01} \geq f_{10}$,*

(i) *if $f_{00}f_{11} = f_{01}f_{10}$, then $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathcal{B}_1^p \rangle_{\omega,p}$;*

(ii) *if $f_{01}, f_{10} = 0$ and $f_{00}, f_{11} > 0$, then $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathcal{B}_1^p \rangle_{\omega,p}$;*

(iii) *if $f_{00}, f_{11} = 0$ and $f_{01}, f_{10} > 0$, then $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle_{\omega,p}$;*

(iv) *if $f_{00}, f_{01}, f_{11} > 0$ and $f_{00}f_{11} > f_{01}f_{10}$, then $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathrm{IMP}, \mathcal{B}_1^p \rangle_{\omega,p}$;*

(v) *otherwise, $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$.*

The non-efficient version — with $\mathcal{B}_1, \mathcal{B}_2$ replacing $\mathcal{B}_1^p, \mathcal{B}_2^p$, and $\langle \cdot \rangle_\omega$ replacing $\langle \cdot \rangle_{\omega,p}$ — also holds.

Proof. To prove $\langle F_1, \mathcal{B}_1^p \rangle_{\omega,p} = \langle F_2, \mathcal{B}_1^p \rangle_{\omega,p}$, it suffices to show that $F_2 \in \langle F_1, \mathcal{B}_1^p \rangle_{\omega,p}$ and $F_1 \in \langle F_2, \mathcal{B}_1^p \rangle_{\omega,p}$. We will verify this in each of the five cases.

(i) Suppose $f_{00}f_{11} = f_{01}f_{10}$. If $f_{00}, f_{01} = 0$, then

$$F(x_1, x_2) = U_1(x_1)U_2(x_2)$$

with $U_1 = (0, 1)$ and $U_2 = (f_{10}, f_{11})$. Similarly if $f_{00}, f_{10} = 0$, $f_{01}, f_{11} = 0$, or $f_{10}, f_{11} = 0$. In the remaining case $f_{00}, f_{01}, f_{10}, f_{11} > 0$. Then choose $U_1 = (1, f_{10}/f_{00})$, $U_2 = (f_{00}, f_{01})$. In all cases $F \in \langle U_1, U_2 \rangle$, so $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathcal{B}_1^p \rangle_{\omega,p}$.

(ii) If $f_{01}, f_{10} = 0$ and $f_{00}, f_{11} > 0$, then $F(x_1, x_2) = U(x_1)\mathrm{EQ}(x_1, x_2)$, where $U = (f_{00}, f_{11})$, so $F \in \langle U \rangle$. Hence $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathcal{B}_1^p \rangle_{\omega,p}$.

(iii) If $f_{00}, f_{11} = 0$ and $f_{01}, f_{10} > 0$, then $F(x_1, x_2) = U(x_1)\mathrm{NEQ}(x_1, x_2)$, where $U = (f_{01}, f_{10})$, so $F \in \langle \mathrm{NEQ}, U \rangle$. Similarly $\mathrm{NEQ}(x_1, x_2) = U'(x_1)F(x_1, x_2)$, where $U' = (1/f_{01}, 1/f_{10})$, so $\mathrm{NEQ} \in \langle F, U' \rangle$. So $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle_{\omega,p}$.

(iv) If $f_{00}, f_{01}, f_{11} > 0$, $f_{00}f_{11} > f_{01}f_{10}$, we can apply unary weights U_1, U_2, where $U_1 = (1/f_{00}, f_{01}/f_{00}f_{11})$, $U_2 = (1, f_{00}/f_{01})$, to implement $\mathrm{IMP}_\alpha(x_1, x_2) = U_1(x_1)U_2(x_2)F(x_1, x_2)$, where

$$M(\mathrm{IMP}_\alpha) \;=\; \begin{bmatrix} 1 & 1 \\ \alpha & 1 \end{bmatrix},$$

where $\alpha = f_{01}f_{10}/f_{00}f_{11} < 1$. Then we have $\mathrm{IMP}_\alpha \in \langle F, U_1, U_2 \rangle$. Note that $\mathrm{IMP}_0 = \mathrm{IMP}$. If $\alpha > 0$, consider the function IMP_α^k, with matrix

$$M(\mathrm{IMP}_\alpha^k) \;=\; \begin{bmatrix} 1 & 1 \\ \alpha^k & 1 \end{bmatrix}.$$

Now IMP_α^k can be implemented as $\mathrm{IMP}_\alpha^k(x_1, x_2) = U_1^k(x_1)U_2^k(x_2)F^k(x_1, x_2)$, by taking k copies of U_1, U_2 and F. Since $\alpha < 1$, we see that $\lim_{k \to \infty} \mathrm{IMP}_\alpha^k = \mathrm{IMP}_0 = \mathrm{IMP}$.

Moreover, the limit is efficient, since $\|\mathrm{IMP} - \mathrm{IMP}_\alpha^k\|_\infty < \varepsilon$ if $k = O(\log \varepsilon^{-1})$ (for fixed $\alpha < 1$), and so an ε-approximation to IMP can be computed in $O(\log \varepsilon^{-1})$ time. Hence $\mathrm{IMP} \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$.

Note that "powering" limits like that used here will be employed below without further discussion of their efficiency.

Conversely, from IMP, we first implement IMP_α. If $\alpha = 0$, we do nothing. Otherwise, we use unary weights U_1, U_2 such that $U_1 = (1/\alpha - 1, 1)$, $U_2 = (\alpha, 1)$, to implement $F_1(x_1, x_2) = U_1(x_1)U_2(x_2)\mathrm{IMP}(x_2, x_1)$, where

$$M(F_1) = \begin{bmatrix} 1 - \alpha & 0 \\ \alpha & 1 \end{bmatrix}.$$

Then $M(\mathrm{IMP}_\alpha) = M(\mathrm{IMP})M(F_1)$, so $\mathrm{IMP}_\alpha \in \langle \mathrm{IMP}, U_1, U_2 \rangle$. Now we can recover $F(x_1, x_2) = U_3(x_1)U_4(x_2)\mathrm{IMP}_\alpha(x_1, x_2)$, where $U_3 = (f_{00}, f_{00}f_{11}/f_{01})$, $U_4 = (1, f_{01}/f_{00})$, so we have $F \in \langle \mathrm{IMP}, U_1, U_2, U_3, U_4 \rangle$. Hence $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \langle \mathrm{IMP}, \mathcal{B}_1^p \rangle_{\omega,p}$.

(v) The remaining cases are (a) $f_{01}, f_{10}, f_{11} > 0$, $f_{00}f_{11} < f_{01}f_{10}$ and (b) $f_{00}, f_{01}, f_{10} > 0$, $f_{11} = 0$.

First, we deal with part (a): If $f_{01}, f_{10}, f_{11} > 0$ and $f_{00}f_{11} < f_{01}f_{10}$, we apply unary weights U_1, U_2, where $U_1 = (f_{11}/f_{01}, 1)$, $U_2 = (1/f_{10}, 1/f_{11})$, to implement $\mathrm{OR}_\alpha(x_1, x_2) = U_1(x_1)U_2(x_2)F(x_1, x_2)$, where $\alpha = f_{00}f_{11}/f_{01}f_{10} < 1$, and

$$M(\mathrm{OR}_\alpha) = \begin{bmatrix} \alpha & 1 \\ 1 & 1 \end{bmatrix}$$

If $\alpha = 0$, $\mathrm{OR}_0 = \mathrm{OR}$, so we have $\mathrm{OR} \in \langle F, \mathcal{B}_1^p \rangle$. Otherwise $\lim_{k \to \infty} \mathrm{OR}_\alpha^k = \mathrm{OR}_0 = \mathrm{OR}$, so we have $\mathrm{OR} \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$.

Conversely, from OR, we first express NEQ. Use the unary function $U = (2, 1/2)$ to implement $F_1 = U(x_1)U(x_2)\mathrm{OR}(x_1, x_2)$, where

$$M(F_1) = \begin{bmatrix} 0 & 1 \\ 1 & 1/4 \end{bmatrix}.$$

Then $\lim_{k \to \infty} F_1^k = \mathrm{NEQ}$, so $\mathrm{NEQ} \in \langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$. Now we observe that $M(\mathrm{IMP}) = M(\mathrm{NEQ})M(\mathrm{OR})$, which implies $\mathrm{IMP} \in \langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$. Also, we have $\mathrm{IMP}_\alpha \in \langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$, as in (iv) above. Finally, $M(\mathrm{OR}_\alpha) = M(\mathrm{NEQ})M(\mathrm{IMP}_\alpha)$, so $\mathrm{OR}_\alpha \in \langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$. Now we can reverse the transformation from F to OR_α to recover F.

Now, we consider part (b): If $f_{00}, f_{01}, f_{10} > 0$ and $f_{11} = 0$, we apply unary weights U_1, U_2, where $U_1 = (1/f_{00}, 1/f_{10})$, $U_2 = (1, f_{00}/f_{01})$, to implement $\mathrm{NAND}(x_1, x_2) = U_1(x_1)U_2(x_2)F(x_1, x_2)$, where

$$M(\mathrm{NAND}) = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix},$$

so we have NAND $\in \langle F, \mathcal{B}_1^p \rangle$. We now use unary weight $U = (1/2, 2)$ to implement $F_1(x_1) = U(x_1)U(x_1)\text{NAND}(x_1, x_2)$ with

$$M(F_1) = \begin{bmatrix} 1/4 & 1 \\ 1 & 0 \end{bmatrix}.$$

Then we have $\lim_{k \to \infty} F_1^k = \text{NEQ}$, so again $\text{NEQ} \in \langle F, \mathcal{B}_1^p \rangle_{\omega, p}$. Then we observe that $M(\text{OR}) = M(\text{NEQ})M(\text{NAND})M(\text{NEQ})$, so $\text{OR} \in \langle F, \mathcal{B}_1^p \rangle_{\omega, p}$.

Conversely, from OR, we have $\text{NEQ} \in \langle \text{OR}, \mathcal{B}_1^p \rangle_{\omega, p}$ from the above. Then we have $M(\text{NAND}) = M(\text{NEQ})M(\text{OR})M(\text{NEQ})$, so $\text{NAND} \in \langle \text{OR}, \mathcal{B}_1^p \rangle_{\omega, p}$. Now we reverse the transformation above from F to NAND to recover F. Thus $F \in \langle \text{OR}, \mathcal{B}_1^p \rangle_{\omega, p}$. □

Remark 13. From Lemma 12, we see that IMP does not really occupy a special position in $\langle \text{IMP}, \mathcal{B}_1^p \rangle_{\omega, p}$, in the sense that there are other functions F with $\langle F, \mathcal{B}_1^p \rangle_{\omega, p} = \langle \text{IMP}, \mathcal{B}_1^p \rangle_{\omega, p}$. Similarly, OR does not occupy a special position in $\langle \text{OR}, \mathcal{B}_1^p \rangle_{\omega, p}$. Nevertheless, it is useful to label the classes this way, and we will do so.

Remark 14. From the proof of Lemma 12, we have the following inclusions between the four classes involved.

$$\langle \mathcal{B}_1^p \rangle_{\omega, p} \subseteq \begin{array}{c} \langle \text{NEQ}, \mathcal{B}_1^p \rangle_{\omega, p} \\ \langle \text{IMP}, \mathcal{B}_1^p \rangle_{\omega, p} \end{array} \subseteq \langle \text{OR}, \mathcal{B}_1^p \rangle_{\omega, p}.$$

In fact, $\langle \text{NEQ}, \mathcal{B}_1^p \rangle_{\omega, p}$ and $\langle \text{IMP}, \mathcal{B}_1^p \rangle_{\omega, p}$ are incomparable, and hence all the inclusions are actually strict. For one non-inclusion, note the clone $\langle \text{IMP}, \mathcal{B}_1^p \rangle_{\omega, p}$ contains only lsm functions, and hence does not contain NEQ. For the other, we claim that any binary function in the clone $\langle \text{NEQ}, \mathcal{B}_1^p \rangle_{\omega, p}$ has one of three forms, $U_1(x)U_2(y)$, $U(x)\text{EQ}(x, y)$ or $U(x)\text{NEQ}(x, y)$, and then observe that IMP matches none of these. The claim is a special case of a more general one, namely that any function in $\langle \text{NEQ}, \mathcal{B}_1^p \rangle_{\omega, p}$ is of the form F_φ, where φ is a product of atomic formulas involving only unary functions, EQ and NEQ. To show this, we need only consider summing over a single variable. By induction, assume we have a pps-formula of the form $\sum_{y \in \{0,1\}} F_\varphi(\boldsymbol{x}, \boldsymbol{z}) F_\psi(\boldsymbol{x}, y)$, where F_ψ is a product of atomic formulas involving y (and certain other variables $\boldsymbol{x}$), and F_φ is a product not involving y. Then

$$\sum_{y \in \{0,1\}} F_\varphi(\boldsymbol{x}, \boldsymbol{z}) F_\psi(\boldsymbol{x}, y) = F_\varphi(\boldsymbol{x}, \boldsymbol{z}) \sum_{y \in \{0,1\}} U(y) \prod_{i=1}^{k} \text{NEQ}(x_i, y)$$

$$= F_\varphi(\boldsymbol{x}, \boldsymbol{z}) \bar{U}(x_1) \prod_{i=1}^{k} \text{EQ}(x_i, x_1),$$

where $\bar{U}(x_1) = U(1 - x_1)$. The product of equalities can be removed by substituting x_j $(j = 2, \ldots, k)$ by x_1 in $\varphi(\boldsymbol{x}, \boldsymbol{z})$, continuing the induction.

Finally, it is straightforward to show that this class is closed under limits. That is, the limit of a sequence of functions which are products of unary and NEQ functions must itself be of this form. To see this, note that every k-ary function in this class can be written as a product of $O(k^2)$ unary and NEQ functions. Then the conclusion follows by a standard compactness argument. The efficient version also follows.

8   The class $\langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$

In the conservative case, we show that, somewhat surprisingly, the clone generated by the single binary function OR contains every function in $\mathcal{B}^p$.

Lemma 15. $\langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p} = \mathcal{B}^p$.

Proof. Suppose $F \in \mathcal{B}_n^p$. Suppose $x_1, \ldots, x_n$ are variables. For each $A \subseteq [n]$, let $\mathbf{1}_A$ be the assignment to $x_1, \ldots, x_n$ in which $x_i = 1$ if $i \in A$ and $x_i = 0$ otherwise. Then we will use the notation $F(A)$ as shorthand for $F(\mathbf{1}_A)$. Let $\mathcal{A} = \{A : F(A) > 0\}$, and let $\mu = \min_{A \in \mathcal{A}} F(A)$. For any $A \subseteq [n]$, let $u_A \in \mathcal{B}_1^p$ be the function such that $u_A(0) = 1$ and $u_A(1) = 2F(A)/\mu - 1 \geq 1$. Note that every function u_A is in $\mathcal{B}_1^p$ and we have $\mathrm{IMP}, \mathrm{NAND} \in \langle \mathrm{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$, from the proof of Lemma 12.

Our goal will be to show that there is a finite subset S_F of $\{\mathrm{IMP}, \mathrm{NAND}\} \cup \mathcal{B}_1^p$ and a TM M_{F,S_F} with the following property: on input $\varepsilon > 0$, M_{F,S_F} computes an arity-n pps-formula ψ over S_F such that $\|F_\psi - F\|_\infty < \varepsilon$. The running time of M_{F,S_F} should be at most a polynomial in $\log \varepsilon^{-1}$. To define S_F, we will use two unary functions U_1 and U_2 (both of which are actually constant functions). We define these by $U_1(0) = U_1(1) = 1/2$ and $U_2(0) = U_2(1) = \mu/2$. Then $S_F = \{\mathrm{IMP}, \mathrm{NAND}, U_1, U_2\} \cup \bigcup_{A \in \mathcal{A}} \{u_A\}$.

Let $V = \{v_1, \ldots, v_n\}$. For $A \in \mathcal{A}$, introduce a new variable z_A. Let $V'' = \{z_A \mid A \in \mathcal{A}\}$ and

$$\psi_1 = \sum_{V''} \left(\prod_{A \in \mathcal{A}} u_A(z_A) \right) \left(\prod_{i \in A} \mathrm{IMP}(z_A, x_i) \right) \left(\prod_{i \notin A} \mathrm{NAND}(z_A, x_i) \right).$$

For every $A \in \mathcal{A}$ the assignment $\boldsymbol{x} = \mathbf{1}_A$ can be extended in two ways (both with $z_A = 0$ and with $z_A = 1$) to satisfy

$$(17) \qquad \left(\prod_{i \in A} \mathrm{IMP}(z_A, x_i) \right) \left(\prod_{i \notin A} \mathrm{NAND}(z_A, x_i) \right) = 1.$$

Any other assignment $\boldsymbol{x}$ can be extended in only one way ($z_A = 0$) to satisfy (17). So if $A \in \mathcal{A}$ then

$$F_{\psi_1}(A) = (2F(A)/\mu - 1) + 1 = 2F(A)/\mu.$$

On the other hand, if $A \notin \mathcal{A}$ then

$$F_{\psi_1}(A) = 1.$$

We have shown that $F_{\psi_1} \in \langle S_F \rangle$. Let us now define

$$\psi_2 = \sum_{V''} \left(\prod_{A \in \mathcal{A}} \prod_{i \in A} \mathrm{IMP}(z_A, x_i) \right) \left(\prod_{i \notin A} \mathrm{NAND}(z_A, x_i) \right).$$

As before, for every $A \in \mathcal{A}$ the assignment $\boldsymbol{x} = \mathbf{1}_A$ can be extended in two ways ($z_A = 0$ and $z_A = 1$) to satisfy (17), and any other assignment $\boldsymbol{x}$ can be extended in only one way ($z_A = 0$) to satisfy it. So

$$F_{\psi_2}(A) = 2 \quad (A \in \mathcal{A}), \qquad F_{\psi_2}(A) = 1 \quad (A \notin \mathcal{A}).$$

Thus $F_{\psi_2} \in \langle S_F \rangle$. Now define F_3 by $F_3(A) = U_1(x_1)F_{\psi_2}(A)$, so $F_3 \in \langle S_F \rangle$, where

$$F_3(A) = 1 \quad (A \in \mathcal{A}), \qquad F_3(A) = \textup{\textonehalf} \quad (A \notin \mathcal{A}).$$

Now $\lim_{k \to \infty} F_3^k = F_0$, where

$$F_0(A) = 1 \quad (A \in \mathcal{A}), \qquad F_0(A) = 0 \quad (A \notin \mathcal{A}),$$

and thus $F_0 \in \langle S_F \rangle_{\omega,p}$.

Note that $F_0 = R_F$, the underlying relation of F. Now define $F_4 = F_{\psi_1}F_0$, so that

$$F_4(A) = 2F(A)/\mu \quad (A \in \mathcal{A}), \qquad F_4(A) = 0 \quad (A \notin \mathcal{A}),$$

Thus, by Lemma 4, $F_4 \in \langle S_F \rangle_{\omega,p}$. Now define F_5 by $F_5(A) = U_2(x_1)F_4(A)$, so $F_5 \in \langle S_F \rangle_{\omega,p}$, where

$$F_5(A) = F(A) \quad (A \in \mathcal{A}), \qquad F_5(A) = 0 \quad (A \notin \mathcal{A}).$$

Since $F_5 = F$, the proof is complete. $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad \square$

9   The main theorem

In this section we prove our main structural result, which characterises, in the conservative case, the clones generated by a single pseudo-Boolean function. Since it is known that any clone generated by a finite set of functions can be generated by a single function [7], Theorem 16 implicitly gives a characterisation of all finitely generated functional clones. This characterisation is given explicitly in Corollary 17 below.

Theorem 16. *Suppose $F \in \mathcal{B}^p$.*

- *If $F \notin \langle \textup{NEQ}, \mathcal{B}_1^p \rangle$ then $\textup{IMP} \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$, and hence $\langle \textup{IMP}, \mathcal{B}_1^p \rangle_{\omega,p} \subseteq \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$*

- *If, in addition, $F \notin \textup{LSM}$ then $\langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \mathcal{B}^p$.*

The non-efficient version — with $\mathcal{B}, \mathcal{B}_1$ replacing $\mathcal{B}^p, \mathcal{B}_1^p$, and $\langle \cdot \rangle_\omega$ replacing $\langle \cdot \rangle_{\omega,p}$ — also holds.

Proof. We start with the first part of the theorem. The aim is to show that either $\textup{IMP} \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ or $F \in \langle \textup{NEQ}, \mathcal{B}_1^p \rangle$. Let C be the relational clone $\langle R_F, \delta_0, \delta_1 \rangle_R$. Since $\{R_F, \delta_0, \delta_1\} \subseteq \{R_{F'} \mid F' \in \{F\} \cup \mathcal{B}_1^p\}$, $C \subseteq \langle R_{F'} \mid F' \in \{F\} \cup \mathcal{B}_1^p \rangle_R$, so by Lemma 5, $C \subseteq \{R_{F'} \mid F' \in \langle F, \mathcal{B}_1^p \rangle\}$.

First, suppose $\textup{IMP} \in C$. Then $\langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ contains a function F' such that $R_{F'} = \textup{IMP}$. The function F' falls into parts (iv) or (v) of Lemma 12, so by this lemma, $\langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ is either $\langle \textup{IMP}, \mathcal{B}_1^p \rangle_{\omega,p}$ or $\langle \textup{OR}, \mathcal{B}_1^p \rangle_{\omega,p}$. Either way, $\langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ contains $\textup{IMP}$ (as noted in Remark 14). Similarly, if $\textup{OR} \in C$ or $\textup{NAND} \in C$ then $\textup{IMP} \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$.

We now consider the possibilities. If R_F is not affine, then Creignou, Khanna and Sudan [15, Lemma 5.30] have shown that one of IMP, OR and NAND is in C. This is also stated and proved as [20, Lemma 15]. In fact, the set of all relational clones (also called "co-clones") is well understood. They are listed in [16, Table 2], which gives a plain basis for each relational
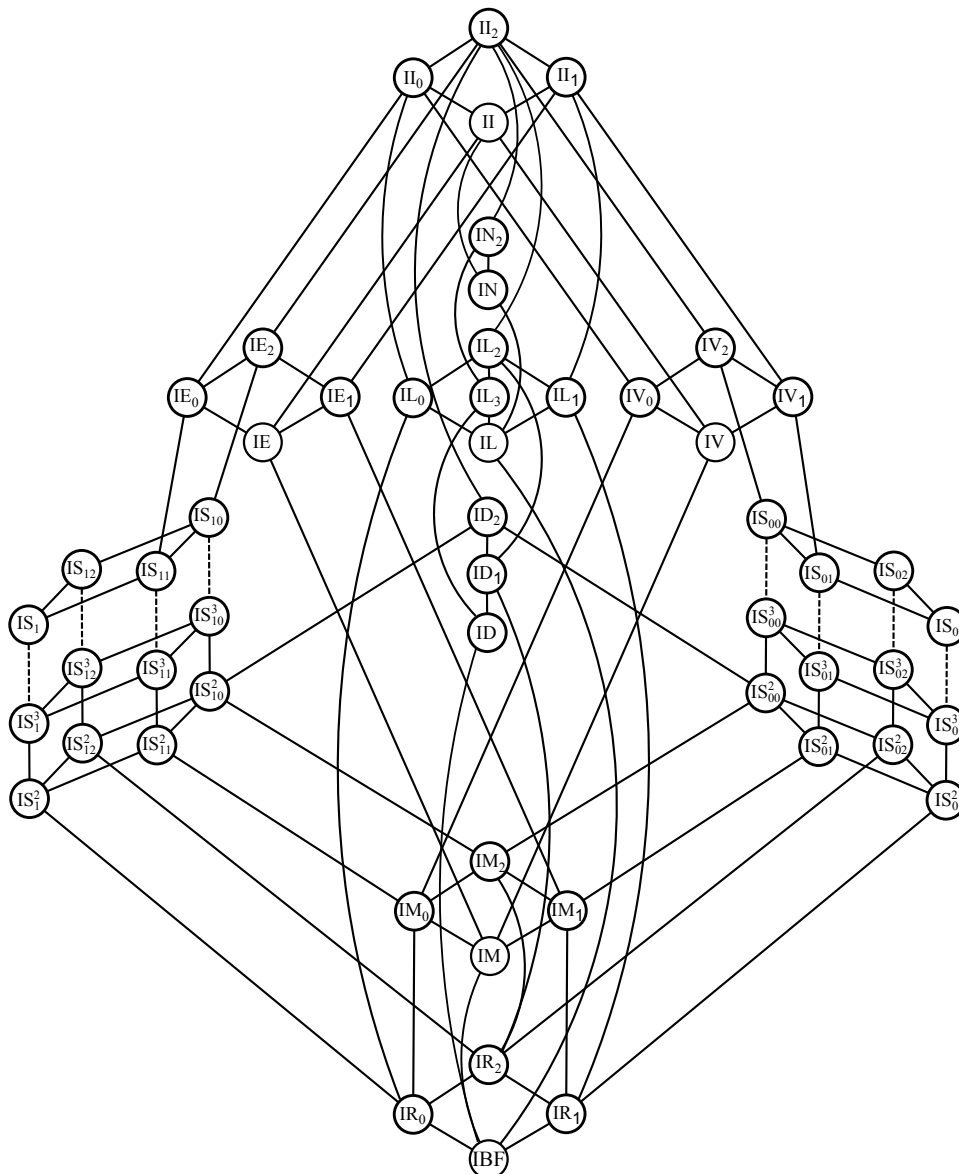
Figure 1: Post's lattice from [3, Fig. 2].

clone. There is a similar table in [4] (though the bases given there are not plain). A Hasse diagram illustrating the inclusions between the relational clones is depicted in [3, Figure 2]. This diagram is reproduced here as Figure 1. A downwards edge from one clone to another indicates that the lower clone is a subset of the higher one. For example, since there is a path (in this case, an edge) from ID_1 down to IR_2 in Figure 1, we deduce that $IR_2 \subset ID_1$. We will not require bases for all relational clones, but we have reproduced the part of [16, Table 2] that we use here as Table 1.

If R_F is affine then the relations in C are given by linear equations, so C is either the relational clone IL_2 (whose plain basis the set of all Boolean linear equations) or C is some subset of IL_2, in which case it is below IL_2 in Figure 1.

Now, EQ, δ_0 and δ_1 are in C. The relational clone containing these relations (and nothing

	Plain Basis
IR_2	$\{\text{EQ}, \delta_0, \delta_1\}$
ID_1	$\{\text{EQ}, \text{NEQ}, \delta_0, \delta_1\}$
IL_2	$\{\{(x_1, \ldots, x_k) \in \{0,1\}^k \mid x_1 + \ldots + x_k = c \pmod 2\} \mid k \in \mathbb{N}, c \in \{0,1\}\}$

Table 1: The relevant portion of Table 2 of [16]: Some relational clones and their plain bases.

else) is IR_2, so C is a (not necessarily proper) superset of IR_2. Thus, C is (not necessarily strictly) above IR_2 in Figure 1. From the figure, it is clear that the only possibilities are that C is one of the relational clones IL_2, ID_1 and IR_2.

Now $\text{IR}_2 \subset \text{ID}_1$ and the plain basis of ID_1 is $\{\text{EQ}, \text{NEQ}, \delta_0, \delta_1\}$. Therefore if $C = \text{IR}_2$ or $C = \text{ID}_1$, then R_F is definable by a CNF formula over $\{\text{EQ}, \text{NEQ}, \delta_0, \delta_1\}$.

Suppose that $F(\boldsymbol{x})$ has arity n and, to avoid trivialities, that R_F is not the empty relation. Suppose that $\psi(v_1, \ldots, v_n)$ is a CNF formula over $\{\text{EQ}, \text{NEQ}, \delta_0, \delta_1\}$ implementing the relation $R_\psi = R_F$.

Let $V = \{v_1, \ldots, v_n\}$. Let ψ_i be the projection of ψ onto variable v_i. ψ_i is one of the three unary relations $\{(0)\}$, $\{(1)\}$, and $\{(0),(1)\}$. Let $V' = \{v_i \in V \mid \psi_i = \{(0),(1)\}\}$. ($V'$ is the set of variables that are not pinned in R_F.) For $v_i \in V'$ and $v_j \in V'$, let $\psi_{i,j}$ be the projection of ψ onto variables v_i and v_j. $\psi_{i,j}$ is a binary relation. Of the 16 possible binary relations, the only ones that can occur are EQ, NEQ and $\{0,1\}^2$. The empty relation is ruled out since R_F is not empty. The four single-tuple binary relations are ruled out since v_i and v_j are in V'. For the same reason, the other four two-tuple binary relations are ruled out. The three-tuple binary relations are ruled out since $\psi_{i,j} \in \text{ID}_1$.

We define an equivalence relation $\sim$ on V' in which $v_i \sim v_j$ iff $\psi_{i,j} \in \{\text{EQ}, \text{NEQ}\}$. Let V'' contain exactly one variable from each equivalence class in V'. Let $k = |V''|$. For convenience, we will assume $V'' = \{v_1, \ldots, v_k\}$. (This can be achieved by renaming variables.)

Now, for every assignment $\boldsymbol{x} : \{v_1, \ldots, v_k\} \to \{0,1\}$ there is exactly one assignment $\boldsymbol{y} : \{v_{k+1}, \ldots, v_n\} \to \{0,1\}$ such that $R_F(\boldsymbol{x}, \boldsymbol{y}) = 1$. Let $\sigma(\boldsymbol{x})$ be this assignment $\boldsymbol{y}$. Now, define the arity-k function G by $G(\boldsymbol{x}) = F(\boldsymbol{x}, \sigma(\boldsymbol{x}))$. Note that

$$(18) \qquad G(\boldsymbol{x}) = \sum_{\boldsymbol{y} \in \{0,1\}^{n-k}} F(\boldsymbol{x}, \boldsymbol{y}),$$

where $\boldsymbol{y}$ is an assignment $\boldsymbol{y} : \{v_{k+1}, \ldots, v_n\} \to \{0,1\}$. Clearly, from (18), $G \in \langle F, \mathcal{B}_1^p \rangle_{\omega, p}$. Also, by construction, $G(\boldsymbol{x})$ is a permissive function so we can apply Lemma 8. We finish with two cases.

Case 1. Every 2-pinning of G is log-modular. Then G is log-modular, by Lemma 8. This means (see, for example, [5, Proposition 24]) that $g = \log_2 G$ is an affine function of $x_1, \ldots, x_k$ and $\bar{x}_1, \ldots, \bar{x}_k$ so $G \in \langle \text{NEQ}, \mathcal{B}_1^p \rangle$. For example, if $g = a_0 + a_1 x_1 + a_2 x_2 + a_3 \bar{x}_3$ then G can be written as

$$G(x_1, x_2, x_3) = \sum_{y_3} U_0 \, U_1(x_1) U_2(x_2) U_3(y_3) \text{NEQ}(x_3, y_3),$$

where $U_0 = 2^{a_0} \in \mathcal{B}_0^p$ and $U_i(x) = 2^{a_i x} \in \mathcal{B}_1^p$. (Note that each quantity 2^{a_i} may be expressed as the quotient of two evaluations of the permissive function $G \in \mathcal{B}^p$, and hence is in $\mathcal{B}_0^p$.) Since $F(\boldsymbol{x}, \boldsymbol{y}) = R_F(\boldsymbol{x}, \boldsymbol{y}) G(\boldsymbol{x})$, we conclude that $F \in \langle \text{NEQ}, \mathcal{B}_1^p \rangle$.

Case 2. There is a 2-pinning G' of G that is not log-modular. Since G is strictly positive, so is G'. Since $G \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$, so is G'. By Lemma 12, (parts (iv) or (v)), IMP $\in \langle G', \mathcal{B}_1^p \rangle_{\omega,p}$. By Lemma 4, IMP $\in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$.

Finally, we consider the case in which $C = \mathrm{IL}_2$. Let $\oplus_3$ be the relation $\{(0,0,0),(0,1,1), (1,0,1),(1,1,0)\}$ containing all triples whose Boolean sums are 0. From the plain basis of IL_2 (Table 1), we see that the relation $\oplus_3$ is in C, so $\langle F, \mathcal{B}_1^p \rangle$ contains a function F' with $R_{F'} = \oplus_3$. Let F'' be the symmetrisation of F' implemented by

$$F''(x,y,z) = F'(x,y,z)F'(x,z,y)F'(y,x,z)F'(y,z,x)F'(z,x,y)F'(z,y,x).$$

Now let $\mu_0 = F''(0,0,0)$ and $\mu_2 = F''(0,1,1)$. Let U be the unary function with $U(0) = \mu_0^{-1/3}$ and $U(1) = \mu_0^{1/6}\mu_2^{-1/2}$. Note that since $F \in \mathcal{B}^p$, the appropriate roots of μ_0 and μ_2 are efficiently computable, so $U \in \mathcal{B}_1^p$. Now $\oplus_3(x,y,z) = U(x)U(y)U(z)F''(x,y,z)$, so $\oplus_3 \in \langle F, \mathcal{B}_1^p \rangle$. Finally, let U' be the unary function defined by $U'(0) = 1$ and $U'(1) = 2$ and let $G(x,z) = \sum_y \oplus_3(x,y,z)U'(y)$. Note that $G(0,0) = G(1,1) = 1$ and $G(0,1) = G(1,0) = 2$. By Lemma 1, G is in $\langle F, \mathcal{B}_1^p \rangle$. But by Lemma 12, IMP $\in \langle G, \mathcal{B}_1^p \rangle_{\omega,p}$ so by Lemma 4, IMP $\in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$.

We now prove Part 2 of the theorem. Suppose that F is not lsm and that $F \notin \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$ so, by Part 1 of the theorem, we have IMP $\in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$. Let

$$H(x_1, x_2) = \sum_{y_1, y_2} \mathrm{IMP}(y_1, x_1)\mathrm{IMP}(y_1, x_2)\mathrm{IMP}(x_1, y_2)\mathrm{IMP}(x_2, y_2).$$

Note that $H(0,0) = H(1,1) = 2$ and $H(0,1) = H(1,0) = 1$. Now for any integer k, let

$$H_k(x_1, \ldots, x_n) = \sum_{y_1, \ldots, y_n} F(y_1, \ldots, y_n) \prod_{i=1}^{n} H(x_i, y_i)^k.$$

By construction, H_k is strictly positive. Also, as k gets large, $H_k(x_1, \ldots, x_n)$ gets closer and closer to $2^{kn}F(x_1, \ldots, x_n)$. Thus, for sufficiently large k, H_k is not lsm. By Lemma 1, $H \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ so $H_k \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$. Applying Lemma 8 to H_k, there is a binary function $F_1 \in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ that is not lsm so

$$F_1(0,0)F_1(1,1) < F_1(0,1)F_1(1,0).$$

By Parts (iii) and (v) of Lemma 12, we either have NEQ $\in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$ or OR $\in \langle F, \mathcal{B}_1^p \rangle_{\omega,p}$. In the latter case, we are finished by Lemma 15. In the former case, we are also finished since (in the notation of the proof of Lemma 12) $M(\mathrm{NEQ})M(\mathrm{IMP}) = M(\mathrm{OR})$ so OR $\in \langle \mathrm{IMP}, \mathrm{NEQ} \rangle$. □

Corollary 17. *Suppose $\mathcal{F} \subseteq \mathcal{B}^p$.*

- *If $\mathcal{F} \not\subseteq \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$ then $\langle \mathrm{IMP}, \mathcal{B}_1^p \rangle_{\omega,p} \subseteq \langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p}$*

- *If, in addition, $\mathcal{F} \not\subseteq \mathsf{LSM}$ then $\langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p} = \mathcal{B}^p$.*

The non-efficient version — with $\mathcal{B}, \mathcal{B}_1$ replacing $\mathcal{B}^p, \mathcal{B}_1^p$, and $\langle \cdot \rangle_\omega$ replacing $\langle \cdot \rangle_{\omega,p}$ — also holds.

Proof. If $\mathcal{F} \not\subseteq \langle \mathrm{NEQ}, \mathcal{B}^p \rangle$ then there exists $F \in \mathcal{F}$ such that $F \notin \langle \mathrm{NEQ}, \mathcal{B}^p \rangle$, so by Part 1 of Theorem 16, $\langle \mathrm{IMP}, \mathcal{B}_1^p \rangle_{\omega,p} \subseteq \langle F, \mathcal{B}_1^p \rangle_{\omega,p} \subseteq \langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p}$.

If, in addition, $\mathcal{F} \not\subseteq \mathsf{LSM}$ then suppose $F_1 \in \mathcal{F}$ and $F_2 \in \mathcal{F}$ where $F_1 \notin \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$ and $F_2 \notin \mathsf{LSM}$. (F_1 and F_2 might be the same function or might not be.) Suppose F_1 and F_2 have arities a_1 and a_2. Let F be the "direct product" of F_1 and F_2 defined by $F(\boldsymbol{x}, \boldsymbol{y}) = F_1(\boldsymbol{x})F_2(\boldsymbol{y})$ where $\boldsymbol{x} \in \{0,1\}^{a_1}$ and $\boldsymbol{y} \in \{0,1\}^{a_2}$. Then $F \in \langle \mathcal{F} \rangle$ and $F \notin \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$ and $F \notin \mathsf{LSM}$. Then, by Theorem 16, $\langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p} \supseteq \langle F, \mathcal{B}_1^p \rangle_{\omega,p} = \mathcal{B}^p$. $\square$

10 Complexity-theoretic consequences

In order to explore the consequences of Theorem 16 for the computational complexity of approximately evaluating #CSPs, we recall the following definitions of FPRASes and AP-reductions from [18].

For our purposes, a counting problem is a function Π from instances w (encoded as a word over some alphabet Σ) to a number $\Pi(w) \in \mathbb{R}^{\geq 0}$. For example, w might encode an instance I of a counting CSP problem #CSP(Γ), in which case $\Pi(w)$ would be the partition function $Z(I)$ associated with I. A *randomised approximation scheme* for Π is a randomised algorithm that takes an instance w and returns an approximation Y to $\Pi(w)$. The approximation scheme has a parameter $\varepsilon > 0$ which specifies the error tolerance. Since the algorithm is randomised, the output Y is a random variable depending on the "coin tosses" made by the algorithm. We require that, for every instance w and every $\varepsilon > 0$,

(19) $$\Pr\left[e^{-\varepsilon} \Pi(w) \leq Y \leq e^{\varepsilon} \Pi(w) \right] \geq {}^3\!/_4 \,.$$

The randomised approximation scheme is said to be a *fully polynomial randomised approximation scheme*, or *FPRAS*, if it runs in time bounded by a polynomial in $|w|$ (the length of the word w) and ε^{-1}. See Mitzenmacher and Upfal [30, Definition 10.2]. Note that the quantity ${}^3\!/_4$ in Equation (19) could be changed to any value in the open interval $({}^1\!/_2, 1)$ without changing the set of problems that have randomised approximation schemes [26, Lemma 6.1].

Suppose that Π_1 and Π_2 are functions from Σ^* to $\mathbb{R}^{\geq 0}$. An "approximation-preserving reduction" (AP-reduction) [18] from Π_1 to Π_2 gives a way to turn an FPRAS for Π_2 into an FPRAS for Π_1. Specifically, an *AP-reduction from Π_1 to Π_2* is a randomised algorithm $\mathcal{A}$ for computing Π_1 using an oracle[3] for Π_2. The algorithm $\mathcal{A}$ takes as input a pair $(w, \varepsilon) \in \Sigma^* \times (0, 1)$, and satisfies the following three conditions: (i) every oracle call made by $\mathcal{A}$ is of the form (v, δ), where $v \in \Sigma^*$ is an instance of Π_2, and $0 < \delta < 1$ is an error bound satisfying $\delta^{-1} \leq \mathrm{poly}(|w|, \varepsilon^{-1})$; (ii) the algorithm $\mathcal{A}$ meets the specification for being a randomised approximation scheme for Π_1 (as described above) whenever the oracle meets the specification for being a randomised approximation scheme for Π_2; and (iii) the run-time of $\mathcal{A}$ is polynomial in $|w|$ and ε^{-1}. Note that the class of functions computable by an FPRAS is closed under AP-reducibility. Informally, AP-reducibility is the most liberal notion of reduction meeting this requirement. If an AP-reduction from Π_1 to Π_2 exists we write $\Pi_1 \leq_{\mathrm{AP}} \Pi_2$. If $\Pi_1 \leq_{\mathrm{AP}} \Pi_2$ and $\Pi_2 \leq_{\mathrm{AP}} \Pi_1$ then we say that *Π_1 and Π_2 are AP-interreducible*, and write $\Pi_1 =_{\mathrm{AP}} \Pi_2$.

---

[3]The reader who is not familiar with oracle Turing machines can just think of this as an imaginary (unwritten) subroutine for computing Π_2.

A word of warning about terminology. Subsequent to [18] the notation $\leq_{AP}$ has been used to denote a different type of approximation-preserving reduction which applies to optimisation problems. We will not study optimisation problems in this paper, so hopefully this will not cause confusion.

The complexity of approximating Boolean #CSPs in the unweighted case, where the functions in Γ have codomain $\{0, 1\}$, was studied earlier [20] by some of the authors of this paper. Two counting problems played a special role there, and in previous work in the complexity of approximate counting [18]. They also play a key role here.

Name #SAT

Instance A Boolean formula φ in conjunctive normal form.

Output The number of satisfying assignments of φ.

Name #BIS.

Instance A bipartite graph B.

Output The number of independent sets in B.

An FPRAS for #SAT would, in particular, have to decide with high probability between a formula having some satisfying assignments or having none. Thus #SAT cannot have an FPRAS unless $NP = RP$.[4] The same is true of any problem to which #SAT is AP-reducible. As far as we are aware, the complexity of approximating #BIS does not relate to any of the standard complexity theoretic assumptions, such as $NP \neq RP$. Nevertheless, there is increasing empirical evidence that no FPRAS for #BIS exists, and we adopt this as a working hypothesis. Of course, this hypothesis implies that no #BIS-hard problem (problem to which #BIS is AP-reducible) admits an FPRAS.

Finally, a precise statement of the computational task we are interested in. A (weighted) #CSP problem is parameterised by a finite subset $\mathcal{F}$ of $\mathcal{B}^p$ and defined as follows.

Name #CSP($\mathcal{F}$)

Instance A pps-formula ψ consisting of a product of m atomic $\mathcal{F}$-formulas over n free variables $\boldsymbol{x}$. (Thus, ψ has no bound variables.)

Output The value $\sum_{\boldsymbol{x} \in \{0,1\}^n} F_\psi(\boldsymbol{x})$ where F_ψ is the function defined by that formula.

Officially, the input size $|w|$ is the length of the encoding of the instance. However, we shall take the size of a #CSP($\mathcal{F}$) instance to be $n + m$, where n is the number of (free) variables and m is the number of constraints (atomic formulas). This is acceptable, as we are only concerned to measure the input size within a polynomial factor; moreover, we have restricted $\mathcal{F}$ to be finite, thereby avoiding the issue of how to encode the constraint functions $\mathcal{F}$. We typically denote an instance of #CSP($\mathcal{F}$) by I and the output by $Z(I)$; by analogy with systems in statistical physics we refer to $Z(I)$ as the partition function.

---

[4]The supposed FPRAS would provide a polynomial-time decision procedure for satisfiability with two-sided error; however, there is a standard trick for converting two-sided error to the one-sided error demanded by the definition of RP [34, Thm 10.5.9].

Aside from simplifying the representation of problem instances, there is another, more important reason for decreeing that $\mathcal{F}$ is finite, namely, that it allows us to prove the following basic lemma relating functional clones and computational complexity. It is, of course, based on a similar result for classical decision CSPs.

Lemma 18. *Suppose that $\mathcal{F}$ is a finite subset of $\mathcal{B}^p$. If $F \in \langle \mathcal{F} \rangle_{\omega,p}$ then*

$$\#\mathsf{CSP}(F, \mathcal{F}) \leq_{\mathrm{AP}} \#\mathsf{CSP}(\mathcal{F}).$$

Proof. Let k be the arity of F. Let $\mathcal{M}$ be a TM which, on input $\varepsilon' > 0$, computes a k-ary pps-formula ψ over $\mathcal{F} \cup \mathrm{EQ}$ such that $\|F_\psi - F\|_\infty < \varepsilon'$. We can assume without loss of generality that no function in $\{F\} \cup \mathcal{F}$ is identically zero (otherwise every $\#\mathsf{CSP}$ instance using this function has partition function 0). Let $\mu_{\max}$ be the maximum value in the range of F and let $\mu_{\min}$ be the minimum of 1 and the minimum non-zero value in the range of F. Similarly, let S be the set of non-zero values in the range of functions in $\mathcal{F} \cup \{\mathrm{EQ}\}$. Let $\nu_{\max}$ be the maximum value in S and let $\nu_{\min}$ be the minimum of 1 and the minimum value in S.

Consider an input (I, ε) where I is an instance of $\#\mathsf{CSP}(F, \mathcal{F})$ and ε is an accuracy parameter. Suppose that I has n variables, m F-constraints, and m' other constraints. We can assume without loss of generality that $m > 0$ (otherwise, I is an instance of $\#\mathsf{CSP}(\mathcal{F})$).

The key idea of the proof is to construct an instance I' of $\#\mathsf{CSP}(\mathcal{F})$ by replacing each F-constraint in I with the set of constraints and extra (bound) variables in the formula ψ that is output by $\mathcal{M}$ with input ε'. We determine how small to make ε' in terms of the following quantities. Let

$$A = \frac{4m}{\mu_{\min}} 2^n \mu_{\max}^m \nu_{\max}^{m'}$$
$$B = 2^n (\mu_{\max} + 1)^{m-1} \nu_{\max}^{m'}$$
$$C = \mu_{\min}^m \nu_{\min}^{m'}.$$

Let $\varepsilon' = \frac{\varepsilon}{4} \frac{C}{A+B}$. The time needed to construct ψ for a given $\varepsilon' > 0$ is at most $\mathrm{poly}(\log{(\varepsilon'^{-1})})$, which is at most a polynomial in n, m, m' and ε^{-1}, as required by the definition of AP-reduction. We shall see that $(I', \varepsilon/2)$ is the sought-for instance/tolerance pair required by our reduction.

Let I_ψ be the instance formed from I by replacing every F-constraint with an F_ψ-constraint. Note that $Z(I_\psi) = Z(I')$, since I', an instance of $\#\mathsf{CSP}(\mathcal{F})$, is an implementation of I_ψ. We want to show that if an oracle produces a sufficiently accurate approximation to $Z(I')$ (and hence to $Z(I_\psi)$) then we can deduce a sufficiently accurate approximation to $Z(I)$. Observe that the definition of FPRAS allows no margin of error when $Z(I) = 0$, and our reduction must give the correct result, namely 0, in this case. Therefore we need to treat separately the cases $Z(I) = 0$ and $Z(I) > 0$. We will show that

(20)
$$Z(I) = 0 \quad \text{implies} \quad Z(I_\psi) < C/3,$$

and

(21)
$$Z(I) > 0 \quad \text{implies} \quad Z(I_\psi) > 2C/3;$$

moreover, in the latter case,

$$(22) \qquad e^{-\varepsilon/2}Z(I) \le Z(I_\psi) \le e^{\varepsilon/2}Z(I).$$

These estimates are enough to ensure correctness of the reduction. For a call to an oracle for $\#\mathsf{CSP}(\mathcal{F})$ with instance I' and accuracy parameter $\varepsilon/2$ would return a result in the range $[e^{-\varepsilon/2}Z(I_\psi), e^{\varepsilon/2}Z(I_\psi)]$ with high probability. Observe that this estimate is sufficient to distinguish between cases (20) and (21). In the former case, we are able to return the exact result, namely 0. In the latter case, we return the result given by the oracle, which by (22) satisfies the conditions for an FPRAS.

To establish (20–22), let Y' be the set of assignments to the variables of instance I which make a non-zero contribution to $Z(I)$ and let Y'' be the remaining assignments to the variables of instance I. Let $Z'(I_\psi)$ be the contribution to $Z(I_\psi)$ due to assignments in Y' and $Z''(I_\psi)$ be the contribution to $Z(I_\psi)$ due to assignments in Y'' (so $Z(I_\psi) = Z'(I_\psi) + Z''(I_\psi)$). We can similarly write $Z(I) = Z'(I) + Z''(I)$, though of course $Z''(I) = 0$.

First, note that if $|F_\psi(\boldsymbol{x}) - F(\boldsymbol{x})| \le \varepsilon'$ and $F(\boldsymbol{x}) > 0$ then

$$|F_\psi(\boldsymbol{x})/F(\boldsymbol{x}) - 1| \le \varepsilon'/F(\boldsymbol{x}) \le \varepsilon'/\mu_{\min},$$

so

$$e^{-2\varepsilon'/\mu_{\min}} \le \frac{F_\psi(\boldsymbol{x})}{F(\boldsymbol{x})} \le e^{2\varepsilon'/\mu_{\min}}.$$

We conclude that

$$e^{-2\varepsilon'm/\mu_{\min}}Z'(I) \le Z'(I_\psi) \le e^{2\varepsilon'm/\mu_{\min}}Z'(I),$$

so

$$|Z'(I) - Z'(I_\psi)| \le \frac{4\varepsilon'm}{\mu_{\min}}Z(I) \le \varepsilon'A.$$

Furthermore,

$$|Z''(I) - Z''(I_\psi)| = Z''(I_\psi) \le \varepsilon'B.$$

Here we use $\|F_\psi - F\|_\infty < \varepsilon' < 1$; the "$+1$" in the definition of B absorbs the discrepancy between F_ψ and F. Combining these two inequalities yields

$$(23) \qquad |Z(I) - Z(I_\psi)| \le \varepsilon'(A + B) \le \frac{\varepsilon C}{4}.$$

Now, $Z(I) > 0$ implies $Z(I) \ge C$, and hence (20) and (21) follow directly from (23). If $Z(I) > 0$ we further have

$$\left|\frac{Z(I_\psi)}{Z(I)} - 1\right| \le \frac{\varepsilon'(A+B)}{Z(I)} \le \frac{\varepsilon'(A+B)}{C} \le \varepsilon/3.$$

This establishes (22) and completes the verification of the reduction. $\square$

Theorem 19. *Suppose $\mathcal{F}$ is a finite subset of $\mathcal{B}^p$.*

- *If $\mathcal{F} \subseteq \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$ then, for any finite subset S of $\mathcal{B}_1^p$, there is an FPRAS for $\#\mathsf{CSP}(\mathcal{F}, S)$.*

- *Otherwise,*

    ○ *There is a finite subset S of $\mathcal{B}_1^p$ such that $\#\mathsf{BIS} \leq_{\mathrm{AP}} \#\mathsf{CSP}(\mathcal{F}, S)$.*

    ○ *If $\mathcal{F} \not\subseteq \mathsf{LSM}$ then there is a finite subset S of $\mathcal{B}_1^p$ such that $\#\mathsf{SAT} =_{\mathrm{AP}} \#\mathsf{CSP}(\mathcal{F}, S)$.*

Proof. First, suppose that $\mathcal{F} \subseteq \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$. Let S be a finite subset of $\mathcal{B}_1^p$. Given an m-constraint input I of $\#\mathsf{CSP}(\mathcal{F}, S)$ and an accuracy parameter ε, we first approximate each arity-k function $F \in \mathcal{F}$ used in I with a function $\widehat{F} : \{0,1\}^k \to \mathbb{Q}^{\geq 0}$ such that $R_{\widehat{F}} = R_F$, and for every $\boldsymbol{x}$ for which $F(\boldsymbol{x}) > 0$,

$$(24) \qquad e^{-\varepsilon/m} \leq \frac{\widehat{F}(\boldsymbol{x})}{F(\boldsymbol{x})} \leq e^{\varepsilon/m}.$$

Let $\widehat{\mathcal{F}} = \{\widehat{F} \mid F \in \mathcal{F}\}$ and let $\widehat{I}$ be the instance of $\#\mathsf{CSP}(\widehat{\mathcal{F}}, S)$ formed from I by replacing each F-constraint with $\widehat{F}$. [19, Theorem 4] gives a polynomial-time algorithm for computing the partition function $Z(\widehat{I})$, which satisfies

$$(25) \qquad e^{-\varepsilon} Z(I) \leq Z(\widehat{I}) \leq e^{\varepsilon} Z(I).$$

Second, suppose that $\mathcal{F} \not\subseteq \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle$. By Theorem 16, $\mathrm{IMP} \in \langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p}$. By Observation 3, there is a finite subset S of $\mathcal{B}_1^p$ such that $\mathrm{IMP} \in \langle \mathcal{F}, S \rangle_{\omega,p}$. Thus, $\#\mathsf{CSP}(\mathrm{IMP}) \leq_{\mathrm{AP}} \#\mathsf{CSP}(\mathcal{F}, S)$, by Lemma 18. However, $\#\mathsf{BIS} =_{\mathrm{AP}} \#\mathsf{CSP}(\mathrm{IMP})$ by [20, Theorem 3].

Finally, suppose that $\mathcal{F} \not\subseteq \mathsf{LSM}$. By Corollary 17, $\langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p} = \mathcal{B}^p$ so $\mathrm{OR} \in \langle \mathcal{F}, \mathcal{B}_1^p \rangle_{\omega,p}$. By Observation 3, there is a finite subset S of $\mathcal{B}_1^p$ such that $\mathrm{OR} \in \langle \mathcal{F}, S \rangle_{\omega,p}$, so by Lemma 18, $\#\mathsf{CSP}(\mathrm{OR}) \leq_{\mathrm{AP}} \#\mathsf{CSP}(\mathcal{F}, S)$. However, by [20, Lemma 7] $\#\mathsf{SAT} \leq_{\mathrm{AP}} \#\mathsf{CSP}(\mathrm{OR})$. To see that $\#\mathsf{CSP}(\mathcal{F}, S) \leq_{\mathrm{AP}} \#\mathsf{SAT}$, let I be an m-constraint instance of $\#\mathsf{CSP}(\mathcal{F}, S)$. For each function $G \in \mathcal{F} \cup S$, define $\widehat{G}$ as in (24). Let $\widehat{I}$ be the instance of $\#\mathsf{CSP}(\{\widehat{G} \mid G \in \mathcal{F} \cup S\})$ formed from I by replacing each G-constraint with a $\widehat{G}$-constraint. Equation (25) holds, as above. Furthermore, from [19, Section 1.3] the problem of evaluating $\#\mathsf{CSP}(\{\widehat{G} \mid G \in \mathcal{F} \cup S\})$ (even with $\widehat{G}$ as part of the input) is in $\#\mathsf{P}_{\mathbb{Q}}$, the complexity class comprising functions which are a function in $\#\mathsf{P}$ divided by a function in FP, so can be AP-reduced to $\#\mathsf{SAT}$. $\square$

Example 20. Let $F \in \mathcal{B}_2^p$ be the function defined by $F(0,0) = F(1,1) = \lambda$ and $F(0,1) = F(1,0) = 1$, where $\lambda > 1$. Then, from Theorem 19, $\#\mathsf{CSP}(F, S)$ is $\#\mathsf{BIS}$-hard, for some set S of unary weights. (In fact, this counting CSP is also $\#\mathsf{BIS}$-easy.) Note that $\#\mathsf{CSP}(F, S)$ is nothing other than the ferromagnetic Ising model with an applied field. So we recover, with no effort, the main result of Goldberg and Jerrum's investigation of this model [21].

Example 21. If F is as before, but $\lambda \in (0, 1)$, then $F \notin \mathsf{LSM}$ and Theorem 19 tells us that $\#\mathsf{CSP}(F, S)$ is $\#\mathsf{SAT}$-hard, for some set S of unary weights. This is a restatement of the known fact that the partition function of the antiferromagnetic Ising model is hard to approximate [25]. Actually, this is true without weights, but that is not directly implied by Theorem 19.

11 The classes $\langle \mathsf{LSM}_k \rangle$

In this section, we will be concerned with expressibility, and less so with efficient computability. Thus, we will use function classes without attempting to distinguish the efficiently computable functions in the class from the remainder.

Define $\mathsf{LSM}_k = \mathsf{LSM} \cap \mathcal{B}_k$. It follows from the proof of Lemma 12 that $\langle \mathsf{LSM}_2 \rangle = \langle \mathsf{IMP}, \mathcal{B}_1 \rangle$. Since LSM is central to Theorem 16, we need to consider whether $\mathsf{LSM} = \langle \mathsf{LSM}_2 \rangle_\omega$ and, if not, what internal structure it may possess. To this end, we consider here the functional clones $\langle \mathsf{LSM}_k \rangle_\omega$ for $k > 2$. As mentioned in the abstract, we will make use of two transforms: the Möbius transform to show $\langle \mathsf{LSM}_3 \rangle_\omega = \langle \mathsf{LSM}_2 \rangle_\omega$ (in fact we prove the stronger statement $\langle \mathsf{LSM}_3 \rangle = \langle \mathsf{LSM}_2 \rangle$), and the Fourier transform to show that $\langle \mathsf{LSM}_4 \rangle_\omega \neq \langle \mathsf{LSM}_2 \rangle_\omega$. See [2, 36, 33] for corresponding results in the context of optimisation.

For all $\boldsymbol{x}, \boldsymbol{y} \in \{0, 1\}^n$, we will write $\boldsymbol{x} \leq \boldsymbol{y}$ to mean that for all $1 \leq i \leq n$ we have $x_i \leq y_i$. For all $f \in \mathscr{B}_n$ define the *Möbius transform* $\widetilde{f} \in \mathscr{B}_n$ by

$$(26) \qquad \widetilde{f}(\boldsymbol{y}) = \sum_{\boldsymbol{w} \leq \boldsymbol{y}} (-1)^{|\boldsymbol{y} - \boldsymbol{w}|} f(\boldsymbol{w}) \qquad\qquad (\boldsymbol{y} \in \{0, 1\}^n),$$

and note that the Möbius transform is invertible:

$$(27) \qquad f(\boldsymbol{x}) = \sum_{\boldsymbol{y} \leq \boldsymbol{x}} \widetilde{f}(\boldsymbol{y}) \qquad\qquad (\boldsymbol{x} \in \{0, 1\}^n).$$

See also [24] for further information. We will not require anything other than (26) and (27). We next show that certain simple functions are in $\langle \mathsf{LSM}_2 \rangle$.

Lemma 22. *Let* $\boldsymbol{y} \in \{0, 1\}^n$. *Let* $t \in \mathbb{R}$, *with* $t \geq 0$ *if* $|\boldsymbol{y}| > 1$. *Let* $F \in \mathcal{B}_n$ *be the unique function satisfying* $\widetilde{\log F}(\boldsymbol{y}) = t$ *and* $\widetilde{\log F}(\boldsymbol{x}) = 0$ *for* $\boldsymbol{x} \neq \boldsymbol{y}$. *(Explicitly,* $F(\boldsymbol{x}) = e^t$ *for* $\boldsymbol{y} \leq \boldsymbol{x}$ *and* $F(\boldsymbol{x}) = 1$ *otherwise.) Then* $F \in \langle \mathsf{LSM}_2 \rangle$.

Proof. If $t \geq 0$ define $U \in \mathsf{LSM}_1$ by $U(0) = 1$ and $U(1) = e^t - 1$. We will argue that for all $\boldsymbol{x}$ we have

$$F(\boldsymbol{x}) = \sum_z U(z) \prod_{i:y_i=1} \mathrm{IMP}(z, x_i).$$

Indeed if $\boldsymbol{y} \leq \boldsymbol{x}$ we get $F(\boldsymbol{x}) = U(0) + U(1) = e^t$, and otherwise $F(\boldsymbol{x}) = U(0) = 1$.

Now we consider the case $|\boldsymbol{y}| \leq 1$. If $\boldsymbol{y} = \boldsymbol{0}$ let $i = 1$, and otherwise let i be the unique index with $y_i = 1$. Then $F(\boldsymbol{x}) = U(x_i)$ where $U(0) = F(\boldsymbol{0})$ and $U(1) = F(\boldsymbol{1})$. Hence $F \in \langle \mathsf{LSM}_1 \rangle \subset \langle \mathsf{LSM}_2 \rangle$. $\qquad\square$

If $\boldsymbol{x} \in \{0, 1\}^n$, let $\bar{\boldsymbol{x}} = \boldsymbol{1} - \boldsymbol{x}$ and, if $F \in \mathcal{B}_n$, let $\bar{F}(\boldsymbol{x}) = F(\bar{\boldsymbol{x}})$. We will use the following simple fact.

Lemma 23. $F \in \mathsf{LSM}_k$ *iff* $\bar{F} \in \mathsf{LSM}_k$.

Proof. By symmetry, it suffices to show that, for any $\boldsymbol{x}, \boldsymbol{y} \in \{0, 1\}^k$,

$$\begin{aligned} \bar{F}(\boldsymbol{x})\bar{F}(\boldsymbol{y}) &= F(\bar{\boldsymbol{x}})F(\bar{\boldsymbol{y}}) \leq F(\bar{\boldsymbol{x}} \wedge \bar{\boldsymbol{y}})F(\bar{\boldsymbol{x}} \vee \bar{\boldsymbol{y}}) \\ &= F(\overline{\boldsymbol{x} \vee \boldsymbol{y}})F(\overline{\boldsymbol{x} \wedge \boldsymbol{y}}) = \bar{F}(\boldsymbol{x} \vee \boldsymbol{y})\bar{F}(\boldsymbol{x} \wedge \boldsymbol{y}). \end{aligned} \qquad\square$$

Now we show that it is only necessary to consider permissive functions. The construction used in the lemma is adapted from one given, in a more general setting, by Topkis [31].

Lemma 24. *For every $F \in \mathsf{LSM}_k$ there exists $G \in \mathsf{LSM}_k^{>0}$ such that $F = R_F G$. Furthermore $R_F \in \langle \mathsf{LSM}_2 \rangle$, so $\langle \mathsf{LSM}_k \rangle = \langle \mathsf{LSM}_k^{>0}, \mathsf{LSM}_2 \rangle$.*

Proof. First, assume $F(\boldsymbol{0}) \neq 0$. Let $\mu = \min_{\boldsymbol{x}} \{ F(\boldsymbol{x}) \mid F(\boldsymbol{x}) \neq 0 \} / \max_{\boldsymbol{x}} F(\boldsymbol{x})$. Set

$$G(\boldsymbol{x}) = \max \{ F(\boldsymbol{y}) \mu^{|\boldsymbol{x}| - |\boldsymbol{y}|} \mid \boldsymbol{y} \leq \boldsymbol{x} \}.$$

Then G is strictly positive and $G(\boldsymbol{x}) = F(\boldsymbol{x})$ wherever $F(\boldsymbol{x}) \neq 0$. It remains to show that $G \in \mathsf{LSM}$. For all $\boldsymbol{x}, \boldsymbol{x}'$ there exist $\boldsymbol{y} \leq \boldsymbol{x}$ and $\boldsymbol{y}' \leq \boldsymbol{x}'$ such that

$$\begin{aligned}
G(\boldsymbol{x})G(\boldsymbol{x}') &= F(\boldsymbol{y})F(\boldsymbol{y}')\mu^{|\boldsymbol{x}|-|\boldsymbol{y}|+|\boldsymbol{x}'|-|\boldsymbol{y}'|} \\
&\leq F(\boldsymbol{y} \wedge \boldsymbol{y}')F(\boldsymbol{y} \vee \boldsymbol{y}')\mu^{|\boldsymbol{x}|-|\boldsymbol{y}|+|\boldsymbol{x}'|-|\boldsymbol{y}'|} \\
&= F(\boldsymbol{y} \wedge \boldsymbol{y}')F(\boldsymbol{y} \vee \boldsymbol{y}')\mu^{|\boldsymbol{x} \wedge \boldsymbol{x}'|-|\boldsymbol{y} \wedge \boldsymbol{y}'|+|\boldsymbol{x} \vee \boldsymbol{x}'|-|\boldsymbol{y} \vee \boldsymbol{y}'|} \\
&\leq G(\boldsymbol{x} \wedge \boldsymbol{x}')G(\boldsymbol{x} \vee \boldsymbol{x}').
\end{aligned}$$

Now we deal with the case $F(\boldsymbol{0}) = 0$. Let $F'(\boldsymbol{x}) = F(\boldsymbol{x})$ for all $\boldsymbol{x} \neq \boldsymbol{0}$ and let $F'(\boldsymbol{0}) = 1$. Then $F' \in \mathsf{LSM}$ and $F'(\boldsymbol{0}) \neq 0$, and we have shown that there exists $G \in \mathsf{LSM}^{>0}$ such that $F' = R_{F'}G$. But then $F = R_F G$.

By [20, Corollary 18], R_F is a conjunction of implications and constants, and hence $R_F \in \langle \delta_0, \delta_1, \mathrm{IMP} \rangle \subset \langle \mathsf{LSM}_2 \rangle$. Thus $\langle \mathsf{LSM}_k \rangle \subseteq \langle \mathsf{LSM}_k^{>0}, \mathsf{LSM}_2 \rangle$. The reverse inclusion is trivial. $\qquad\square$

Lemma 25. $\langle \mathsf{LSM}_3 \rangle = \langle \mathsf{LSM}_2 \rangle$.

Proof. From Lemma 24, we need only prove $\mathsf{LSM}_3^{>0} \subseteq \langle \mathsf{LSM}_2 \rangle$. Thus let $F \in \mathsf{LSM}_3^{>0}$, $f = \log F$, and first assume $\tilde{f}(1,1,1) \geq 0$. We will show that $F \in \langle \mathsf{LSM}_2 \rangle$. Note that, by log-supermodularity of F,

$$\tilde{f}(1,1,0) = f(0,0,0) - f(1,0,0) - f(0,1,0) + f(1,1,0) \geq 0.$$

and similarly $\tilde{f}(1,0,1), \tilde{f}(0,1,1) \geq 0$. Hence $\tilde{f}(\boldsymbol{y}) \geq 0$ for all $|\boldsymbol{y}| > 1$. For all $\boldsymbol{y} \in \{0,1\}^3$ let $F_{\boldsymbol{y}}$ be the unique function satisfying $\widetilde{\log F_{\boldsymbol{y}}}(\boldsymbol{y}) = \tilde{f}(\boldsymbol{y})$ and $\widetilde{\log F_{\boldsymbol{y}}}(\boldsymbol{z}) = 0$ for $\boldsymbol{z} \neq \boldsymbol{y}$. Then $\widetilde{\log F} = \sum_{\boldsymbol{y}} \widetilde{\log F_{\boldsymbol{y}}}$, which implies $\log F = \sum_{\boldsymbol{y}} \log F_{\boldsymbol{y}}$, which implies $F = \prod_{\boldsymbol{y}} F_{\boldsymbol{y}}$. By Lemma 22 we have $F_{\boldsymbol{y}} \in \langle \mathsf{LSM}_2 \rangle$ for all $\boldsymbol{y}$, and therefore $F \in \langle \mathsf{LSM}_2 \rangle$. If $\tilde{f}(1,1,1) < 0$, let $H = \overline{F}$. Note that $H \in \mathsf{LSM}_3^{>0}$ and $\widetilde{\log H}(1,1,1) > 0$, so by the previous paragraph, $H \in \langle \mathsf{LSM}_2 \rangle$. By Lemma 23 this implies $F \in \langle \mathsf{LSM}_2 \rangle$. $\qquad\square$

In view of Lemma 25, it might be conjectured that $\mathsf{LSM} = \langle \mathsf{LSM}_2 \rangle_\omega$. In fact, this is not the case, as we will now show. First we consider the class $\mathcal{P}$ of functions $F \in \mathcal{B}$ for which the *Fourier transform* $\widehat{F}$ has nonnegative coefficients, where

$$(28) \qquad \widehat{F}(\boldsymbol{y}) = \frac{1}{2^n} \sum_{\boldsymbol{w} \in \{0,1\}^n} (-1)^{|\boldsymbol{w} \wedge \boldsymbol{y}|} F(\boldsymbol{w}) \qquad (\boldsymbol{y} \in \{0,1\}^n).$$

Thus $F \in \mathcal{P}$ if and only if $\widehat{F} \in \mathcal{B}$. See [17] for further information. We will use (28) and the *convolution theorem*: for all $F, G \in \mathcal{B}_n$ we have

$$(29) \qquad \widehat{FG}(\boldsymbol{x}) \;=\; \sum_{\boldsymbol{y} \in \{0,1\}^n} \widehat{F}(\boldsymbol{y}) \widehat{G}(\boldsymbol{x} \oplus \boldsymbol{y}) \qquad\qquad (\boldsymbol{x}, \boldsymbol{y} \in \{0,1\}^n).$$

where $\oplus$ denotes componentwise addition modulo 2. See for example [17, Section 2.3] for a proof of the dual statement.

To show that $\mathcal{P}$ is closed under pps-formula evaluation, it is useful to restrict to atomic formulas where variables are not repeated within a scope.

Lemma 26. *Let $\mathcal{F} \subseteq \mathcal{B}$. For all pps-formulas ψ over $\mathcal{F}$ there is another pps-formula ψ' over $\mathcal{F} \cup \{\mathrm{EQ}\}$ such that $F_\psi = F_{\psi'}$ and no atomic formula of ψ' contains a repeated variable.*

Proof. Given ψ obtain ψ' as follows. For each variable v_i that is used $d_i \geq 2$ times in total in ψ, replace the uses of v_i by new distinct variables $v_i^1, \cdots, v_i^{d_i}$, multiply by atomic formulas $\mathrm{EQ}(v_i, v_i^j)$ for $1 \leq j \leq d_i$, then sum over these new variables v_i^j. $\qquad\square$

Lemma 27. *$\mathcal{P}$ is closed under addition, summation, products and limits. Moreover, $\mathcal{P}$ is a pps_ω-definable functional clone.*

Proof. If $F, G \in \mathcal{P}$, then $\widehat{F + G} = \widehat{F} + \widehat{G}$ is clearly non-negative, and $\widehat{FG}$ is nonnegative by the convolution theorem (29). For summation, as in Lemma 6, we consider summing over the last variable. So, let $H(\boldsymbol{x}) = \sum_t F(\boldsymbol{x}, t)$. Then it follows easily from (28) that $\widehat{H}(\boldsymbol{y}) = 2\widehat{F}(\boldsymbol{y}, 0) \geq 0$ for all $\boldsymbol{y}$. For limits note that if $F_n \to F$ then $\widehat{F_n} \to \widehat{F}$, and a limit of non-negative functions is non-negative.

Let ψ be a pps-formula over $\mathcal{P} \cup \{\mathrm{EQ}\}$. We will argue that that $F_\psi \in \mathcal{P}$. By Lemma 26 there is a pps-formula ψ' over $\mathcal{P} \cup \{\mathrm{EQ}\}$ such that $F_\psi = F_{\psi'}$ and such that no atomic formula of ψ contains a repeated variable. The functions F_φ defined by atomic formulas $\varphi = G(v_{i_1}, \cdots, v_{i_k})$ of ψ' are therefore "expansions": permutations of the function $G' \in \mathcal{B}_n$, $n \geq k$, defined by

$$(30) \qquad G'(\boldsymbol{x}, \boldsymbol{x}') \;=\; G(\boldsymbol{x}) \qquad\qquad (\boldsymbol{x} \in \{0,1\}^k \text{ and } \boldsymbol{x}' \in \{0,1\}^{n-k}).$$

It therefore suffices to check that $\mathcal{P}$ is closed under expansions. Let G' be the expansion defined by (30). Then, for all $\boldsymbol{y} \in \{0,1\}^k$ and $\boldsymbol{y}' \in \{0,1\}^{n-k}$, we have $\widehat{G'}(\boldsymbol{y}, \boldsymbol{y}') = \widehat{G}(\boldsymbol{y})$ if $\boldsymbol{y}' = 0^k$, and $\widehat{G'}(\boldsymbol{y}, \boldsymbol{y}') = 0$ otherwise, and hence $G' \in \mathcal{P}$. Note that $\widehat{\mathrm{EQ}} = \frac{1}{2}\mathrm{EQ}$, so $\mathrm{EQ} \in \mathcal{P}$. Thus $\mathcal{P}$ is a pps-definable functional clone, but it is also closed under limits. $\qquad\square$

For $F \in \mathcal{B}$, let $F^\star$ denote $F\bar{F}$. Now let $\mathcal{C}$ be the class of functions $F \in \mathcal{B}$ such that $G^\star \in \mathcal{P}$ for every pinning $G(\boldsymbol{x}) = F(\boldsymbol{x}; \boldsymbol{c})$. Note, in particular, that if $U \in \mathcal{B}_1$, $U^\star(z) = U(0)U(1)$, a nonnegative constant. Therefore we have $\mathcal{B}_1 \subseteq \mathcal{C}$ and, to establish that $F \in \mathcal{C}$, we need only check pinnings of F of arity at least 2.

Lemma 28. *$\mathcal{C}$ is a pps_ω-definable functional clone.*

Proof. As in Lemma 27 we will check that $\mathcal{C}$ is closed under "expansions", products, summations, and limits. But a pinning of an expansion (or product, summation, or limit) of functions in $\mathcal{C}$ is an expansion (or product, summation, or limit) of pinnings of functions in $\mathcal{C}$, which are necessarily in $\mathcal{C}$ because $\mathcal{C}$ is closed under pinnings. So it suffices to check the $\mathcal{C}$ condition for trivial pinnings, for example to check closure under products it suffices to show that $F, G \in \mathcal{C}$ implies $(FG)^\star \in \mathcal{P}$.

Let $G \in \mathcal{C}$ have arity k, let $n \geq k$, let G' be the function defined by (30). Note that $G'\overline{G'}$ is an expansion of $G\overline{G}$, so $G'\overline{G'} \in \mathcal{P}$ and $G' \in \mathcal{C}$. We have $\mathrm{EQ} \in \mathcal{C}$, since $\mathrm{EQ}\,\overline{\mathrm{EQ}} = \mathrm{EQ} \in \mathcal{P}$. Closure under product follows from Lemma 27 and the observation that $(FG)^\star = F^\star G^\star$. For summation, again as in Lemma 6, we consider summing over the last variable. Then, if $H(\boldsymbol{x}) = \sum_t F(\boldsymbol{x}, t)$, where F has arity $k + 1$, then

$$H^\star(\boldsymbol{x}) = \sum_t F(\boldsymbol{x}, t) \sum_t \bar{F}(\boldsymbol{x}, t) = (F_0)^\star(\boldsymbol{x}) + (F_1)^\star(\boldsymbol{x}) + \sum_t F^\star(\boldsymbol{x}, t).$$

where F_0 and F_1 are the pinnings $F_i(\boldsymbol{x}) = F(\boldsymbol{x}; i)$. We have $(F_0)^\star, (F_1)^\star \in \mathcal{P}$ by the pinning assumption, and the arity k function $\sum_t F^\star(\boldsymbol{x}, t)$ is in $\mathcal{P}$ by Lemma 27. Thus $H^\star$ is the sum of three functions in $\mathcal{P}$, and so, using Lemma 27 again, $H^\star \in \mathcal{P}$. Finally note that $\mathcal{C}$ is closed under limits: if $F_n \to F$ as $n \to \infty$ then $F_n^\star \to F^\star$, but $\mathcal{P}$ is closed under limits. □

Lemma 29. $\langle \mathsf{LSM}_2 \rangle_\omega \subseteq \mathcal{C}.$

Proof. Let $F \in \mathsf{LSM}_2$. Note that $\widehat{F^\star}(0,0) = (F(0,0)F(1,1) + F(0,1)F(1,0))/2$, and $\widehat{F^\star}(0,1) = \widehat{F^\star}(1,0) = 0$, and $\widehat{F^\star}(1,1) = (F(0,0)F(1,1) - F(0,1)F(1,0))/2 \geq 0$. So $F^\star \in \mathcal{P}$, and hence $F \in \mathcal{C}$. Thus $\mathsf{LSM}_2 \subseteq \mathcal{C}$ and, since $\mathcal{C}$ is a pps_ω-definable functional clone, $\langle \mathsf{LSM}_2 \rangle_\omega \subseteq \mathcal{C}$. □

Lemma 30. $\langle \mathsf{LSM}_2 \rangle_\omega \subset \langle \mathsf{LSM}_4 \rangle_\omega.$

Proof. Since $\mathsf{LSM}_2 \subseteq \mathcal{C}$ by Lemma 29, we need only exhibit a function $F \in \mathsf{LSM}_4$ which is not in $\mathcal{C}$. Define $F : \{0,1\}^4 \to \mathbb{R}^{>0}$ by

$$F(x_1, x_2, x_3, x_4) = \begin{cases} 4, & \text{if } x_1 + x_2 + x_3 + x_4 = 4; \\ 2, & \text{if } x_1 + x_2 + x_3 + x_4 = 3; \\ 1, & \text{otherwise.} \end{cases}$$

To show $F \in \mathsf{LSM}_4$, by the symmetry of F and Lemma 8, it suffices to show that the three 2-pinnings $F(0,0,x_3,x_4)$, $F(0,1,x_3,x_4)$ and $F(1,1,x_3,x_4)$ are lsm. This is equivalent to the inequalities $1 \times 1 \geq 1 \times 1$, $2 \times 1 \geq 1 \times 1$, and $4 \times 1 \geq 2 \times 2$ respectively, which clearly hold.

To show that $F \notin \mathcal{C}$, we need only use (28) to calculate

$$\widehat{F^\star}(1,1,1,1) = \frac{4 \times 1 - 4 \times 2 + 6 \times 1 - 4 \times 2 + 4 \times 1}{2^4} = -\frac{1}{8} < 0.$$

Indeed $F \in \langle \mathsf{LSM}_4 \rangle_\omega$ but $F \notin \mathcal{C}$. By Lemma 29, $\langle \mathsf{LSM}_2 \rangle_\omega \subseteq \langle \mathsf{LSM}_4 \rangle_\omega \cap \mathcal{C} \subset \langle \mathsf{LSM}_4 \rangle_\omega$. □

Unfortunately, this approach does not seem to extend to showing $\langle \mathsf{LSM}_4 \rangle_\omega \subset \mathsf{LSM}$ or even $\langle \mathsf{LSM}_4 \rangle \subset \mathsf{LSM}$. Neither can we extend the result of Lemma 25 to show that $\langle \mathsf{LSM}_4 \rangle_\omega = \langle \mathsf{LSM}_5 \rangle_\omega$. However, we will venture the following, which is true for $k = 1$.

Conjecture 31. For all $k \geq 1$, $\langle \mathsf{LSM}_{2k} \rangle_\omega = \langle \mathsf{LSM}_{2k+1} \rangle_\omega \subset \langle \mathsf{LSM}_{2k+2} \rangle_\omega$.

A consequence of a proof of Conjecture 31 would be that $\mathsf{LSM} \neq \langle \mathcal{F} \rangle_\omega$ for any finite set of functions $\mathcal{F}$.

12   Restricted unary weights

In this section and the next, we depart from the conservative case, and consider allowing only restricted classes of unary weights.

We have already noted that restricting to nonnegative (as opposed to arbitrary real) unary weights produces a richer lattice of functional clones, and an apparently richer complexity landscape. Thus, by further restricting the unary functions available, we might expect to further refine the lattice of functional clones.

In this section, we will consider the unary functions that favour 1 over 0, or vice versa. With a view to studying this setting, let $\mathcal{B}_1^{\text{down}}$ (respectively $\mathcal{B}_1^{\text{up}}$) be the class of unary functions U from $\mathcal{B}_1$ such that $U(1) \leq U(0)$ (respectively, $U(0) \leq U(1)$). By $\mathcal{B}_1^{\text{down},p}$ and $\mathcal{B}_1^{\text{up},p}$ we denote the efficient versions of these sets. Also denote by EQ' the *permissive equality* function defined by $\text{EQ}'(x,y) = 2$ if $x = y$, and $\text{EQ}'(x,y) = 1$ otherwise.

The first hint that partitioning $\mathcal{B}_1$ into $\mathcal{B}_1^{\text{up}}$ and $\mathcal{B}_1^{\text{down}}$ may yield new phenomena comes from the observation that for any finite subset S of $\mathcal{B}_1^{\text{down},p}$, the problem $\#\mathsf{CSP}(\text{EQ}', S)$ reduces to the ferromagnetic Ising model with a consistent external field, for which Jerrum and Sinclair have given an FPRAS [25]. (A consistent field is one that favours one of the two spins consistently over every site.) The point here is that $\#\mathsf{CSP}(\text{EQ}', S)$ is tractable, even though $\text{EQ}' \notin \langle \text{NEQ}, \mathcal{B}_1 \rangle_\omega$. In contrast, by Theorem 19 or from the arguments in [21], there is a finite subset S of $\mathcal{B}_1^p$ such that $\#\mathsf{CSP}(\text{EQ}', S)$ — the ferromagnetic Ising model with local fields, with different spins favoured at different sites — is $\#\mathsf{BIS}$-hard. Of course similar remarks apply to $\mathcal{B}_1^{\text{up},p}$.

In terms of functional clones, the clone $\langle \text{EQ}', \mathcal{B}_1^{\text{down}} \rangle_\omega$ is not amongst those we met in §9: it is incomparable with $\langle \text{NEQ}, \mathcal{B}_1 \rangle_\omega$, and strictly contained in $\langle \text{IMP}, \mathcal{B}_1 \rangle_\omega$. Specifically, we have

Lemma 32.   *(i)* $\text{NEQ} \notin \langle \text{EQ}', \mathcal{B}_1^{\text{down}} \rangle_\omega$.

(ii) $\text{EQ}' \notin \langle \text{NEQ}, \mathcal{B}_1 \rangle_\omega$.

(iii) $\langle \text{EQ}', \mathcal{B}_1^{\text{down}} \rangle_\omega \subset \langle \text{EQ}', \mathcal{B}_1 \rangle_\omega = \langle \text{IMP}, \mathcal{B}_1 \rangle_\omega$,
and $\langle \text{EQ}', \mathcal{B}_1^{\text{down},p} \rangle_{\omega,p} \subset \langle \text{EQ}', \mathcal{B}_1^p \rangle_{\omega,p} = \langle \text{IMP}, \mathcal{B}_1^p \rangle_{\omega,p}$.

Proof. Recall the class $\mathcal{P}$ of functions with non-negative Fourier coefficients defined in §11. Note that $\mathcal{P}$ is a pps_ω-definable functional clone by Lemma 27. Note that $\widehat{\text{EQ}'}(0,0) = 6/4$ and $\widehat{\text{EQ}'}(1,1) = 2/4$ and $\widehat{\text{EQ}'}(0,1) = \widehat{\text{EQ}'}(1,0) = 0$, so $\text{EQ}' \in \mathcal{P}$. Also, for any $U \in \mathcal{B}_1$ we always have $\widehat{U}(0) \geq 0$, but $\widehat{U}(1) = (U(0) - U(1))/2 \geq 0$ if and only if $U \in \mathcal{B}_1^{\text{down}}$. So $\langle \text{EQ}', \mathcal{B}_1^{\text{down}} \rangle_\omega \subseteq \mathcal{P}$.

For (i) note that $\widehat{\text{NEQ}}(1,1) = -2/4 < 0$ so $\text{NEQ} \notin \mathcal{P}$.

For (ii), Remark 14 showed that all functions in $\langle \text{NEQ}, \mathcal{B}_1 \rangle_\omega$ are products of atomic formulas. Therefore, if $\text{EQ}' \in \langle \text{NEQ}, \mathcal{B}_1 \rangle_\omega$, it must have one of the three forms $U_1(x)U_2(y)$, $U_1(x)\text{EQ}(x,y)$ or $U_1(x)\text{NEQ}(x,y)$, where $U_1, U_2 \in \mathcal{B}_1$. Now note that $\text{EQ}'(x,y)$ is not of any of these.

For (iii), the inclusion $\langle \text{EQ}', \mathcal{B}_1^{\text{down}} \rangle_\omega \subseteq \langle \text{EQ}', \mathcal{B}_1 \rangle_\omega$ is trivial. It is strict since, as we showed above, $\langle \text{EQ}', \mathcal{B}_1^{\text{down}} \rangle_\omega \cap \mathcal{B}_1 = \mathcal{B}_1^{\text{down}}$. The equality follows from Lemma 12(iv). $\square$

It is interesting to note that the strict inclusion between $\langle \mathrm{EQ}', \mathcal{B}_1^{\mathrm{down}} \rangle_{\omega,p}$ and $\langle \mathrm{EQ}', \mathcal{B}_1 \rangle_{\omega,p}$ is provable, even though the gap in the computational complexity of the related counting problems is only suspected. The other side of the coin is that two functional clones may differ, without there being a corresponding gap in complexity between the two counting CSPs. The main result of the section exhibits this phenomenon in a natural context: the two functional clones are incomparable, but there is an approximation-preserving reduction from one of the corresponding counting CSPs to the other. This is interesting, as it demonstrates that it is sometimes necessary, when constructing approximation-preserving reductions, to go beyond the gadgetry implied by the clone construction (even with the liberal notion employed here, including limits).

Recall that $\oplus_3$ is the relation $\{(0,0,0), (0,1,1), (1,0,1), (1,1,0)\}$.

Lemma 33. $\oplus_3 \notin \langle \mathrm{IMP}, \mathcal{B}_1^{\mathrm{down},p} \rangle_{\omega,p}$ *and* $\mathrm{IMP} \notin \langle \oplus_3, \mathcal{B}_1^{\mathrm{down},p} \rangle_{\omega,p}$

Proof. First we show that $\oplus_3 \notin \langle \mathrm{IMP}, \mathcal{B}_1^{\mathrm{down},p} \rangle_{\omega,p}$. By Lemma 7, $\langle \mathrm{IMP}, \mathcal{B}_1^{\mathrm{down},p} \rangle_{\omega,p} \subseteq \mathsf{LSM}$. However, $\oplus_3 \notin \mathsf{LSM}$, since for $\boldsymbol{x} = (1,1,0)$ and $\boldsymbol{y} = (0,1,1)$,

$$0 = \oplus_3(\boldsymbol{x} \vee \boldsymbol{y}) \oplus_3(\boldsymbol{x} \wedge \boldsymbol{y}) < \oplus_3(\boldsymbol{x}) \oplus_3(\boldsymbol{y}) = 1.$$

Now we show that $\mathrm{IMP} \notin \langle \oplus_3, \mathcal{B}_1^{\mathrm{down},p} \rangle_{\omega,p}$. Recall the class $\mathcal{P}$ of functions with non-negative Fourier coefficients defined in §11. Note that $\mathcal{P}$ is a pps_ω-definable functional clone by Lemma 27. For all $U \in \mathcal{B}_1^{\mathrm{down}}$ we have $\widehat{U}(0) = (U(0) + U(1))/2 \geq 0$ and $\widehat{U}(1) = (U(0) - U(1))/2 \geq 0$, so $U \in \mathcal{P}$. Also, $\widehat{\oplus_3} = \frac{1}{2}\mathrm{EQ}_3$ where EQ_3 is the arity 3 equality relation $\{(0,0,0),(1,1,1)\}$. So $\langle \oplus_3, \mathcal{B}_1^{\mathrm{down},p} \rangle_{\omega,p} \subseteq \mathcal{P}$. But $\widehat{\mathrm{IMP}}(0,1) = (1 - 1 - 1)/4 < 0$, so $\mathrm{IMP} \notin \mathcal{P}$. $\qquad\square$

We know now that IMP is not pps_ω-definable in terms of $\oplus_3$ and $\mathcal{B}_1^{\mathrm{down},p}$. In contrast, we see in the next result that IMP is nevertheless efficiently reducible to $\oplus_3$ and $\mathcal{B}_1^{\mathrm{down},p}$.

Lemma 34. *There is a finite subset S of $\mathcal{B}_1^{\mathrm{down},p}$ such that*

$$\#\mathsf{CSP}(\mathrm{IMP}) \leq_{\mathrm{AP}} \#\mathsf{CSP}(\oplus_3, S)$$

Proof. First, we need some definitions. Suppose that M is a matrix over $\mathrm{GF}(2)$ with rows V and columns E with $|V| = n$. For a column e and a "configuration" $\sigma : V \to \{0,1\}$, define $\delta_e(\sigma)$ to be $\bigoplus_{i \in V} M_{i,e}\sigma(i)$, where the addition is over $\mathrm{GF}(2)$. $\delta_e(\sigma)$ is the parity of the number of 1s in column e of M that are assigned to 1 by σ. Given a parameter $y > 0$, the *Ising partition function* of the binary matroid $\mathcal{M}$ represented by M is given by

$$Z_{\mathrm{Ising}}(\mathcal{M}; y) = \sum_{\sigma : V \to \{0,1\}} \prod_{e \in E} y^{1 \oplus \delta_e(\sigma)}.$$

Now, from [20, Theorem 3], $\#\mathsf{CSP}(\mathrm{IMP}) =_{\mathrm{AP}} \#\mathsf{BIS}$. Also, for every efficiently approximable real number $y > 1$, from [23, Theorem 1] there is an AP-reduction from $\#\mathsf{BIS}$ to the problem of computing $Z_{\mathrm{Ising}}(\mathcal{M}; y)$, for given M.

The set of subsets $A \subseteq E$ such that the submatrix corresponding to A has an even number of 1s in every row is called the *cycle space* of $\mathcal{M}$ and is denoted $\mathcal{C}(\mathcal{M})$. A standard calculation expresses $Z_{\mathrm{Ising}}(\mathcal{M}; y)$ in terms of $\mathcal{C}(\mathcal{M})$. Let $w = (y-1)/(y+1)$.

$$\sum_{\sigma:V\to\{0,1\}} \prod_{e\in E} y^{1\oplus\delta_e(\sigma)} = \sum_{\sigma}\prod_{e}\left(\frac{y+1}{2}+\frac{y-1}{2}(-1)^{\delta_e(\sigma)}\right)$$

$$= \left(\frac{y+1}{2}\right)^{|E|}\sum_{\sigma}\prod_{e}\left(1+w(-1)^{\delta_e(\sigma)}\right)$$

$$= \left(\frac{y+1}{2}\right)^{|E|}\sum_{\sigma}\sum_{A\subseteq E}\prod_{e\in A}w(-1)^{\delta_e(\sigma)}$$

$$= \left(\frac{y+1}{2}\right)^{|E|}\sum_{A\subseteq E}w^{|A|}\sum_{\sigma}\prod_{e\in A}(-1)^{\delta_e(\sigma)}$$

$$= \left(\frac{y+1}{2}\right)^{|E|}\sum_{A\in\mathcal{C}(\mathcal{M})}w^{|A|}2^n,$$

$$= \left(\frac{y+1}{2}\right)^{|E|}2^n\sum_{A\in\mathcal{C}(\mathcal{M})}w^{|A|}.$$

Here is the justification of the penultimate line (why only $A \in \mathcal{C}(\mathcal{M})$ contribute to the sum, and why the factor 2^n): Suppose, for a set $A \subseteq E$, that some row i has has an odd number of 1's in columns in A. Then for any configuration $\sigma' : V \setminus \{i\} \to \{0,1\}$, one of the contributions extending σ' to domain V contributes -1 and the other contributes $+1$. On the other hand, if i has an even number of 1's in A, then the two contributions are the same, so we just get a factor of 2 times the contribution from the smaller problem, without this row.

Note that, since $y > 1$, we have $0 < w < 1$. Now the point is that it is easy to express the sum $\sum_{A\in\mathcal{C}(\mathcal{M})} w^{|A|}$ as the solution to an instance of $\#\mathsf{CSP}(\oplus_3, U_w)$, where U_w is the unary function defined by $U_w(0) = 1$ and $U_w(1) = w$. A vector $\boldsymbol{x}$ represents the choice of $A \subseteq E$ — the j'th column is in A iff $x_j = 1$. Then the constraint that the submatrix corresponding to A has an even number of 1s in some row, say row i, is given by the linear equation $\bigoplus_{j:M_{i,j}=1} x_j = 0$. If this linear equation has just two terms then it is an equality, and it can be represented in the CSP instance by substituting one variable for the other. Otherwise, it can be expressed using conjunctions of atomic formulas $\oplus_3$. Thus, we have an AP-reduction from $\#\mathsf{CSP}(\mathsf{IMP})$ to $\#\mathsf{CSP}(\oplus_3, U_w)$. $\qquad\square$

Remark 35. Lemma 33 and 34 show that, as far as counting CSPs are concerned, the expressibility provided by efficient pps$_\omega$-definable functional clones is more limited than AP-reductions. Here we show that the problem $\#\mathsf{CSP}(\mathsf{IMP})$ is AP-reducible to a $\#\mathsf{CSP}$ problem whose constraint language consists of functional constraints from $\oplus_3 \cup \mathcal{B}_1^{\mathrm{down},p}$, but we aren't able to express IMP using $\oplus_3 \cup \mathcal{B}_1^{\mathrm{down},p}$. On the other hand, if the definition of efficient pps$_\omega$-definable functional clones were somehow extended to remedy this deficiency, then Lemma 18 would probably have to be weakened. While we do know (from Lemma 34) that there is a finite subset S of $\mathcal{B}_1^{\mathrm{down},p}$ for which $\#\mathsf{CSP}(\mathsf{IMP}) \leq_{\mathrm{AP}} \#\mathsf{CSP}(\oplus_3, S)$, the corresponding stronger statement from Lemma 18, $\#\mathsf{CSP}(\mathsf{IMP}, \oplus_3, S) \leq_{\mathrm{AP}} \#\mathsf{CSP}(\oplus_3, S)$, is unlikely to be true since $\#\mathsf{CSP}(\mathsf{IMP}, \oplus_3, S) =_{\mathrm{AP}} \#\mathsf{SAT}$ [20, Theorem 3].

Remark 36. Lemma 33, as with the earlier Lemma 32, shows that there may be a rich structure of efficient functional clones $\langle \mathcal{F} \rangle_{\omega,p}$ with $\mathcal{B}_1^{\mathrm{down},p} \subseteq \mathcal{F}$. By contrast, Corollary 17 and Lemma 7 guarantee that if $\mathcal{B}_1^p \subseteq \mathcal{F}$, then, the only possibilities are

- $\langle \mathcal{F} \rangle_{\omega,p} \subseteq \langle \mathrm{NEQ}, \mathcal{B}_1^p \rangle_{\omega,p}$, or

- $\langle \mathrm{IMP}, \mathcal{B}_1^p \rangle_{\omega,p} \subseteq \langle \mathcal{F} \rangle_{\omega,p} \subseteq \mathsf{LSM}$,

- or $\langle \mathcal{F} \rangle_{\omega,p} = \mathcal{B}^p$.

13   Using fewer weights

In this section, we show that one or two unary weights often suffice to generate the functional clones that we encountered in previous sections.

In the following proposition, $\frac{1}{2}$ denotes the constant nullary function that takes value $\frac{1}{2}$.

Lemma 37.   *(i)* $\mathcal{B}_1 \subseteq \langle \mathrm{IMP}, \frac{1}{2} \rangle_{\omega}$ $(\mathcal{B}_1^p \subseteq \langle \mathrm{IMP}, \frac{1}{2} \rangle_{\omega,p})$.
        (ii) $\mathcal{B}_1^{\mathrm{up}} \subseteq \langle \mathrm{OR}, \frac{1}{2} \rangle_{\omega}$ $(\mathcal{B}_1^{\mathrm{up},p} \subseteq \langle \mathrm{OR}, \frac{1}{2} \rangle_{\omega,p})$.
        (iii) $\mathcal{B}_1^{\mathrm{down}} \subseteq \langle \mathrm{NAND}, \frac{1}{2} \rangle_{\omega}$ $(\mathcal{B}_1^{\mathrm{down},p} \subseteq \langle \mathrm{NAND}, \frac{1}{2} \rangle_{\omega,p})$.

Proof. Note that (iii) is the same as (ii) with the roles of 0 and 1 reversed, so we will just prove (i) and (ii). We start with a general construction that works for both parts (i) and (ii) of the proposition. Let F be a binary function and I, J instances of $\#\mathsf{CSP}(F)$. We assume the sets of variables of I and J are disjoint. The disjoint sum $I \uplus J$ of I and J is the instance whose set of variables is the union of those of I and J, and $F(x,y)$ is in $I \uplus J$ if and only if $F(x,y)$ occurs in I or J. The ordinal sum $I +_{\leq} J$ of I and J is their disjoint sum along with every atomic formula $F(x,y)$ such that x is a variable of I and y is a variable of J.

Claim.   (i) For any $F \in \mathcal{B}_2$, $Z(I \uplus J) = Z(I) \cdot Z(J)$.
        (ii) If $F \in \{\mathrm{IMP}, \mathrm{OR}\}$ then $Z(I +_{\leq} J) = Z(I) + Z(J) - 1$.

The first part is trivial. To show the second part consider an assignment $(\boldsymbol{x}, \boldsymbol{y})$ such that $\boldsymbol{x}, \boldsymbol{y}$ map the variables of I, J, respectively, to $\{0,1\}$, and $F_{I+_{\leq}J}(\boldsymbol{x}, \boldsymbol{y}) \neq 0$. If $F = \mathrm{IMP}$ and any of the components of $\boldsymbol{x}$ equals 1, then $\boldsymbol{y} = \boldsymbol{1}$. However, if $\boldsymbol{x} = \boldsymbol{0}$ (or $\boldsymbol{y} = \boldsymbol{1}$) then $\boldsymbol{y}$ (resp. $\boldsymbol{x}$) can be any legitimate assignment of J (resp. I). If $F = \mathrm{OR}$ then one of the $\boldsymbol{x}, \boldsymbol{y}$ must be $\boldsymbol{1}$, while the remaining one can be any assignment with $F_I(\boldsymbol{x}) \neq 0$ or $F_J(\boldsymbol{y}) \neq 0$. This completes the proof of the claim.

Denote the instance consisting of a single variable without constraints by $\boldsymbol{2}$, the disjoint sum of k instances $\boldsymbol{2}$ by $\boldsymbol{2}^k$, and the ordinal sum $I +_{\leq} \ldots +_{\leq} I$ of k copies of instance I by $k \cdot I$. (Note that the operator $+_{\leq}$ is associative, so this makes sense.) Let also $\frac{\boldsymbol{1}}{\boldsymbol{2}}$ denote the instance consisting of a single nullary $\frac{1}{2}$ function, and let $\frac{\boldsymbol{1}}{\boldsymbol{2}}^k$ denote the sum of k copies of $\frac{1}{2}$. Note that $Z(\boldsymbol{2}) = 2$ and $Z(\frac{\boldsymbol{1}}{\boldsymbol{2}}) = \frac{1}{2}$, justifying the notation. Note that for every natural number a and every positive integer ℓ, $Z(a \cdot \boldsymbol{2}^{\ell}) = a2^{\ell} - a + 1$. (This can be proved by induction on a with base case $a = 0$, using Part (ii) of the claim for the inductive step.) Furthermore, for every positive integer k, if $a_1, \ldots, a_k$ are natural numbers

then $Z(a_1 \cdot \mathbf{2}^1 +_\leq \cdots +_\leq a_k \cdot \mathbf{2}^k) = (a_1 2^1 + \cdots + a_k 2^k) - (a_1 + \cdots + a_k) + 1$. (This can be proved by induction on k using base case $k = 1$ using the previous observation.)

Suppose $G \in \mathcal{B}_1$ and let $G^{(n)}$ be a rational valued approximation to G such that $G^{(n)}(z) \neq 0$ and $|G^{(n)}(z) - G(z)| \leq 2^{-n}$, for $z \in \{0, 1\}$. (This is possible even if $G(z) = 0$.) Assume that this rational approximation is given as a finite binary expansion, so that $G^{(n)}(0) = \frac{1}{2^m}(a_0 + a_1 2^1 + \cdots + a_k 2^k) \neq 0$ and $G^{(n)}(1) = \frac{1}{2^m}(b_0 + b_1 2^1 + \cdots + b_\ell 2^\ell) \neq 0$. Let I, J be instances of #CSP(F) given by

$$I = a_1 \cdot \mathbf{2}^1 +_\leq \cdots +_\leq a_k \cdot \mathbf{2}^k +_\leq (a_0 + a_1 + \cdots + a_k - 1) \cdot \mathbf{2},$$
$$J = b_1 \cdot \mathbf{2}^1 +_\leq \cdots +_\leq b_\ell \cdot \mathbf{2}^\ell +_\leq (b_0 + b_1 + \cdots + b_\ell - 1) \cdot \mathbf{2}.$$

From the observations above, $Z((a_0 + \cdots + a_k - 1) \cdot \mathbf{2}) = a_0 + \cdots + a_k$ so $Z(I) = 2^m G^{(n)}(0)$. Similarly, $Z(J) = 2^m G^{(n)}(1)$. Let V_0, V_1 be the variables of I, J, respectively, and let C_0 (respectively, C_1) be the set of pairs (x, y) such that $F(x, y)$ is an atomic formula of I (respectively, J).

First suppose $F = \text{IMP}$. Consider the formula

$$\psi_n = \sum (\tfrac{1}{2})^m \left(\prod_{(a,b)\in C_0\cup C_1} \text{IMP}(a, b) \right) \left(\prod_{a\in V_0} \text{IMP}(c, a) \right) \left(\prod_{b\in V_1} \text{IMP}(b, c) \right)$$

where c is a new variable and the sum is over all variables in $V_0 \cup V_1$. An assignment $\boldsymbol{x} : V_0 \cup V_1 \cup \{c\} \to \{0, 1\}$ can only contribute to F_{ψ_n} if either: $\boldsymbol{x}(c) = 0$ and $\boldsymbol{x}(b) = 0$ for all $b \in V_1$, or $\boldsymbol{x}(c) = 1$ and $\boldsymbol{x}(a) = 1$ for all $a \in V_0$. Hence $F_{\psi_n}(0) = 2^{-m} Z(I) = G^{(n)}(0)$ and $F_{\psi_n}(1) = 2^{-m} Z(J) = G^{(n)}(1)$.

Now suppose $F = \text{OR}$, and let $G \in \mathcal{B}_1^{\text{up}}$ and $G^{(n)}$ be as before, but with the restriction $G(1) > G(0)$. This time, let $G^{(n)}(0) = \frac{1}{2^m}(a_0 + a_1 2^1 + \cdots + a_k 2^k) \neq 0$ and $G^{(n)}(1) - G^{(n)}(0) = \frac{1}{2^m}(b_0 + b_1 2^1 + \ldots + b_\ell 2^\ell) \neq 0$. Here we are using the fact that $G^{(n)}(1) - G^{(n)}(0) > 0$, which follows from $G(1) > G(0)$ for sufficiently large n. (In the degenerate case $G(1) = G(0)$, one can still find suitable approximations by taking $b_0, b_1, \ldots, b_{\ell-1} = 0$ and $b_\ell = 1$.) Let instances I, J be defined for the values $a_0 + a_1 2^1 + \cdots + a_k 2^k$, $1 + b_0 + b_1 2^1 + \cdots + b_\ell 2^\ell$ in the similar way to before (but note the extra 1); and let V_0, V_1, C_0, C_1 again denote the set of variables and constraints of I, J. As before $Z(I) = 2^m G^{(n)}(0)$ and $Z(J) = 2^m(G^{(n)}(1) - G^{(n)}(0)) + 1$. Let $K = I +_\leq J$ and let C be the set of scopes of the constraints in K. Consider the formula

$$\psi_n = \sum (\tfrac{1}{2})^m \left(\prod_{(a,b)\in C} \text{OR}(a, b) \right) \left(\prod_{b\in V_1} \text{OR}(b, c) \right),$$

where c is a new free variable and the sum is over all variables in $V_0 \cup V_1$. An assignment $\boldsymbol{x} : V_0 \cup V_1 \cup \{c\} \to \{0, 1\}$ can only contribute to F_{ψ_n} if either: $\boldsymbol{x}(c) = 0$ and $\boldsymbol{x}(b) = 1$ for all $b \in V_0$, or $\boldsymbol{x}(c) = 1$. Therefore $F_{\psi_n}(0) = 2^{-m} Z(I) = G^{(n)}(0)$ and $F_{\psi_n}(1) = 2^{-m} Z(I +_\leq J) = 2^{-m}(Z(I) + Z(J) - 1) = G^{(n)}(1)$.

To obtain the efficient version of the proposition let M_0 and M_1 be TMs that, given n, compute the first n bits of $G(0)$ and $G(1)$ respectively, in time polynomial in n. Then a TM M' that constructs $G_\varepsilon \in \langle F, \frac{1}{2} \rangle$ such that $\|G_\varepsilon - G\|_\infty < \varepsilon$ works as follows. First, it finds

the smallest n such that $n > \log \varepsilon^{-1}$. Then it runs M_0 and M_1 on input n to find $G^{(n)}(0)$ and $G^{(n)}(1)$. Finally, M' outputs the formula ψ_n. The running time of M' is the sum of: the time to run M_0, the time to run M_1, and time $O(\log^2 \varepsilon^{-1})$ to construct ψ_n. $\qquad\square$

Corollary 38. *Let $G \in \mathcal{B}_1$. (For the results on efficient pps-definability assume $G \in \mathcal{B}_1^p$.)*

(i) *If $G(0) > G(1)$ and $G(1) \neq 0$ then $\mathcal{B}_1 \subseteq \langle \mathrm{OR}, G, \frac{1}{2}\rangle_\omega$,*
   and $\mathcal{B}_1^p \subseteq \langle \mathrm{OR}, G, \frac{1}{2}\rangle_{\omega,p}$.

(ii) *If $G(0) < G(1)$ and $G(0) \neq 0$ then $\mathcal{B}_1 \subseteq \langle \mathrm{NAND}, G, \frac{1}{2}\rangle_\omega$,*
   and $\mathcal{B}_1^p \subseteq \langle \mathrm{NAND}, G, \frac{1}{2}\rangle_{\omega,p}$.

Proof. We prove (i), as (ii) is quite similar. Let H be a function in $\mathcal{B}_1$. If $H(0) \leq H(1)$ then, by Proposition 37, $H \in \langle \mathrm{OR}, \frac{1}{2}\rangle_\omega$ (or $H \in \langle \mathrm{OR}, \frac{1}{2}\rangle_{\omega,p}$). Assume $H(0) > H(1)$. There is k such that $\frac{G(0)^k}{G(1)^k} > \frac{H(0)}{H(1)}$. Let $H' = H/G^k$. Then $H' \in \mathcal{B}_1^{\mathrm{up}}$ so, by Proposition 37, $H' \in \langle \mathrm{OR}, \frac{1}{2}\rangle_\omega$. Hence $H \in \langle \mathrm{OR}, G, \frac{1}{2}\rangle_\omega$. Also, if $G, H \in \mathcal{B}_1^p$ then $H' \in \mathcal{B}_1^{\mathrm{up},p}$ so $H' \in \langle \mathrm{OR}, \frac{1}{2}\rangle_{\omega,p}$. Hence $H \in \langle \mathrm{OR}, G, \frac{1}{2}\rangle_{\omega,p}$. $\qquad\square$

Corollary 39. $\mathcal{B} \subseteq \langle \mathrm{OR}, \mathrm{NAND}, \frac{1}{2}\rangle_\omega$ *and* $\mathcal{B}^p \subseteq \langle \mathrm{OR}, \mathrm{NAND}, \frac{1}{2}\rangle_{\omega,p}$.

Proof. Let $F \in \mathcal{B}^p$. Let $U(x) = \sum_y \mathrm{NAND}(x,y)$. Then $U(0) = 2$ and $U(1) = 1$ and $U \in \langle \mathrm{OR}, \mathrm{NAND}, \frac{1}{2}\rangle_{\omega,p}$. By Corollary 38 we have $\mathcal{B}_1^p \subseteq \langle \mathrm{OR}, U, \frac{1}{2}\rangle_{\omega,p}$, and by Lemma 15, $F \in \langle \mathrm{OR}, \mathcal{B}_1^p\rangle_{\omega,p}$. So $F \in \langle \mathrm{OR}, \mathrm{NAND}, \frac{1}{2}\rangle_{\omega,p}$ by Lemma 4. $\qquad\square$

References

[1] R. Ahlswede and D. E. Daykin, *An inequality for the weights of two families of sets, their unions and intersections*, Z. Wahrsch. Verw. Gebiete, 43 (1978), pp. 183–185.

[2] A. Billionnet and M. Minoux, *Maximizing a supermodular pseudoboolean function: A polynomial algorithm for supermodular cubic functions*, Discrete Applied Mathematics, 12 (1985), pp. 1 – 11.

[3] E. Böhler, N. Creignou, S. Reith, and H. Vollmer, *Playing with Boolean blocks, part II: Constraint satisfaction problems*, ACM SIGACT Newsletter, 35 (2004), pp. 22–35.

[4] E. Böhler, S. Reith, H. Schnoor, and H. Vollmer, *Bases for Boolean co-clones*, Inf. Process. Lett., 96 (2005), pp. 59–66.

[5] E. Boros and P. L. Hammer, *Pseudo-Boolean optimization*, Discrete Applied Mathematics, 123 (2002), pp. 155–225.

[6] A. A. Bulatov, *Complexity of conservative constraint satisfaction problems*, ACM Trans. Comput. Log., 12 (2011), pp. 24:1–24:66.

[7] A. A. Bulatov, M. Dyer, L. A. Goldberg, M. Jalsenius, M. Jerrum, and D. Richerby, *The complexity of weighted and unweighted #CSP*, J. Comput. Syst. Sci., 78 (2012), pp. 681–688.

[8] A. A. BULATOV, M. DYER, L. A. GOLDBERG, AND M. JERRUM, *Log-supermodular functions, functional clones and counting CSPs*, in 29th International Symposium on Theoretical Aspects of Computer Science, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012, pp. 302–313.

[9] A. A. BULATOV AND M. GROHE, *The complexity of partition functions*, Theor. Comput. Sci., 348 (2005), pp. 148–186.

[10] J.-Y. CAI, X. CHEN, AND P. LU, *Non-negative weighted #CSPs: An effective complexity dichotomy*, CoRR, abs/1012.5659 (2010).

[11] J.-Y. CAI, P. LU, AND M. XIA, *Dichotomy for Holant* problems of Boolean domain*, in Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, 2011, pp. 1714–1728.

[12] P. CHEBOLU, L. A. GOLDBERG, AND R. MARTIN, *The complexity of approximately counting stable matchings*, Theoretical Computer Science, 437 (2012), pp. 35 – 68.

[13] X. CHEN, *Guest column: Complexity dichotomies of counting problems*, SIGACT News, 42 (2011), pp. 54–76.

[14] D. COHEN AND P. JEAVONS, *Chapter 8: The complexity of constraint languages*, in Handbook of Constraint Programming, vol. 2 of Foundations of Artificial Intelligence, Elsevier, 2006, pp. 245–280.

[15] N. CREIGNOU, S. KHANNA, AND M. SUDAN, *Complexity Classifications of Boolean Constraint Satisfaction Problems*, SIAM, Philadelphia, PA, USA, 2001.

[16] N. CREIGNOU, P. KOLAITIS, AND B. ZANUTTINI, *Structure identification of Boolean relations and plain bases for co-clones*, Journal of Computer and System Sciences, 74 (2008), pp. 1103–1115.

[17] R. DE WOLF, *A Brief Introduction to Fourier Analysis on the Boolean Cube*, no. 1 in Graduate Surveys, Theory of Computing Library, 2008.

[18] M. DYER, L. A. GOLDBERG, C. GREENHILL, AND M. JERRUM, *The relative complexity of approximate counting problems*, Algorithmica, 38 (2004), pp. 471–500.

[19] M. DYER, L. A. GOLDBERG, AND M. JERRUM, *The complexity of weighted Boolean #CSP*, SIAM Journal on Computing, 38 (2009), pp. 1970–1986.

[20] ——, *An approximation trichotomy for Boolean #CSP*, Journal of Computer and System Sciences, 76 (2010), pp. 267–277.

[21] L. A. GOLDBERG AND M. JERRUM, *The complexity of ferromagnetic Ising with local fields*, Combin. Probab. Comput., 16 (2007), pp. 43–61.

[22] L. A. Goldberg and M. Jerrum, *Approximating the partition function of the ferro-magnetic Potts model*, in Automata, Languages and Programming, 37th International Colloquium, Proceedings, Part I, vol. 6198 of Lecture Notes in Computer Science, Springer, 2010, pp. 396–407.

[23] ——, *Approximating the Tutte polynomial of a binary matroid and other related combinatorial polynomials*, CoRR, abs/1006.5234 (2010).

[24] M. Grabisch, J.-L. Marichal, and M. Roubens, *Equivalent representations of set functions*, Math. Oper. Res., 25 (2000), pp. 157–178.

[25] M. Jerrum and A. Sinclair, *Polynomial-time approximation algorithms for the Ising model*, SIAM J. Comput., 22 (1993), pp. 1087–1116.

[26] M. Jerrum, L. G. Valiant, and V. Vazirani, *Random generation of combinatorial structures from a uniform distribution*, Theoret. Comput. Sci., 43 (1986), pp. 169–188.

[27] K.-I. Ko and H. Friedman, *Computational complexity of real functions*, Theoretical Computer Science, 20 (1982), pp. 323–352.

[28] V. Kolmogorov and S. Živný, *The complexity of conservative valued CSPs*, in Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2012, pp. 750–759.

[29] C. McQuillan, *LSM is not generated by binary functions*, CoRR, abs/1110.0461 (2011).

[30] M. Mitzenmacher and E. Upfal, *Probability and Computing*, Cambridge University Press, Cambridge, 2005.

[31] D. M. Topkis, *Minimizing a submodular function on a lattice*, Operations Research, 26 (1978), pp. 305–321.

[32] J. Torán, *On the complexity of computable real sequences*, RAIRO Inform. Théor. Appl., 21 (1987), pp. 175–180.

[33] S. Živný and P. Jeavons, *Classes of submodular constraints expressible by graph cuts*, Constraints, 15 (2010), pp. 430–452.

[34] I. Wegener, *Complexity Theory*, Springer-Verlag, Berlin, 2005.

[35] T. Yamakami, *Approximate counting for complex-weighted Boolean constraint satisfaction problems*, CoRR, abs/1007.0391 (2010).

[36] S. Živný, D. Cohen, and P. Jeavons, *The expressive power of binary submodular functions*, Discrete Appl. Math., 157 (2009), pp. 3347–3358.