Counting problems and clones of functions

Andrei A. Bulatov and Amir Hedayaty

Simon Fraser University, abulatov@cs.sfu.ca, hedayaty@gmail.com

Abstract. Counting solutions of various combinatorial problems is a well established and intensively studied area of theoretical computer science. Its applications range from networking to statistical physics. The central question in this area is, of course, the complexity of and algorithms for problems of this type. It turns out that in many important cases these questions can be answered with aid of clones of functions of multi-valued logic. In this paper we give a short review of the connections between the two areas and related results; then we extend those connections to approximate counting and present some initial results showing that clones of functions are useful in this area as well.

Counting solutions of various combinatorial problems is a task ubiquitous in computer science, combinatorics and elswhere. Applications of counting problems range across a variety of research areas from graph theory, to logic, to networking, to statistical physics, and quantum computing. The *Counting Constraint Satisfaction Problem* (#CSP for short) provides a powerful yet convenient formalism to express counting problems.

The focus of this paper is the connection between the complexity of counting CSPs, and clones of functions of multi-valued logic and co-clones of relations. This approach, sometimes referred to as the *algebraic approach*, has proved to be very efficient in the study of the decision CSPs, see, e.g., [5, 6, 1, 20]. We show how this approach can be applied to counting problems. In some cases, such as exact counting of solutions to CSPs, this approach allows one to classify completely the complexity of such problems and develop new solution algorithms. In other cases, such as approximate counting, the connection is weaker, and allows one to employ only partial functions and co-clones of the corresponding type. In this case many problems remain open. We give a short introduction into counting CSPs, and the algebraic approach, list the most notable recent results, and then show how the algebraic approach can be extended onto approximation complexity and prove some initial results in this direction, determining the approximation complexity of the counting CSPs that arise from maximal partial clones on a 3-element set.

1 Counting Problems

Definitions and examples. For a set D by D^n we denote the set of all *n-tuples* of elements of D. An *n-ary relation* is any set $R \subseteq D^n$. Tuples will be denoted in boldface, say, $\mathbf{a}$, and their entries denoted by $\mathbf{a}[1], \dots, \mathbf{a}[n]$. We will also need *predicates* corresponding to relations. To simplify the notation we use the same symbol for a relation and the corresponding predicate, for instance, for an n-ary relation R the corresponding predicate $R(x_1, \dots, x_n)$ is given by $R(\mathbf{a}[1], \dots, \mathbf{a}[n]) = 1$ if and only if $\mathbf{a} \in R$. A set of relations over the same set D is called a *constraint language* over D.

Definition 1 *An* instance *of the #CSP is a triple $\mathcal{P} = (V, D, \mathcal{C})$, where V is a finite set of* variables*, D is a set of possible* values *of the variables (also finite in this paper), and $\mathcal{C}$ is a collection of* constraints*. Each constraint $C = \langle \mathbf{s}, R \rangle$ is a pair consisting of an n_C-tuple $\mathbf{s}$ of variables, called the* constraint scope*, and an n_C-ary relation, called the* constraint relation*.*

A solution *of the instance $\mathcal{P}$ is a mapping $\varphi\colon V \to D$ such that for each constraint $C = \langle \mathbf{s}, R \rangle$ the tuple $(\varphi(\mathbf{s}[1]), \dots, \varphi(\mathbf{s}[n_C]))$ belongs to R. The objective is to find the number $\#\mathcal{P}$ of solutions to $\mathcal{P}$.*

In this paper we will mostly be concerned with problems of the form $\#\mathrm{CSP}(\Gamma)$, where Γ is a constraint language over a fixed finite set D. The restriction Γ imposes is that $\#\mathrm{CSP}(\Gamma)$ allows only those instances whose constraint relations belong to Γ. In this case we will sometimes omit the set of values when specifying an instance. If Γ consists of a single relation R, we write $\#\mathrm{CSP}(R)$, rather than $\#\mathrm{CSP}(\{R\})$.

Example 1 (3-SAT, [7, 8, 27, 26]). In the #3-SATISFIABILITY problem, given a propositional formula in the conjunctive normal form such that every clause contains exactly 3 literals, the goal is to find the number of satisfying assignments to this formula. It is not hard to see that #3-SAT is exactly the problem $\#\mathrm{CSP}(\Gamma_{3-\mathrm{SAT}})$, where $\Gamma_{3-\mathrm{SAT}}$ denotes the constraint language over $\{0,1\}$ consisting of 8 ternary relations; each of these relations contains all the triples satisfying a clause with a certain combination of positive and negative literals.

Example 2 (#H-Coloring, [13, 17, 21]). Let $H = (V, E)$ be a (di)graph. In the #H-COLORING problem, given a graph $G = (W, F)$, the goal is to find the number of *homomorphisms* from G to H, that is, mappings from W to V such that every edge of G is mapped into an edge of H. Observe that if H is a complete graph with k vertices then #H-COLORING PROBLEM is the regular #k-COLORING problem, in which the goal is find the number of proper k-colorings of a given graph G.

As is easily seen, the #H-COLORING PROBLEM is equivalent to $\#\mathrm{CSP}(\{E\})$, that is, #CSP over the constraint language containing only one relation, the edge relation of the graph. The constraints then correspond to the edges in F.

Example 3 (Systems of linear equations). The objective in the #SYSTEMS-OF-LINEAR-EQUATIONS problem is to count the number of solutions of a given system of linear equations over a finite field. This problem is equivalent to $\#\mathrm{CSP}(\Gamma_{\mathrm{Lin}})$, where Γ_{Lin} is the set of all relations representable as the solution space of a linear equation.

Example 4 (Fourier coefficients). Evaluating the quantum Fourier transform is one of the very few tasks that can be performed in polynomial time on a quantum computer, but not known to have an efficient classical algorithms. In the simplest form this problem can be viewed as a generalization of #3-SAT and can be expressed as follows. Let f be an n-ary Boolean function, $f\colon \{0,1\}^n \to \{0,1\}$. Let $S = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$, *Fourier coefficient* $\hat{f}(S)$ of f is given by

$$\hat{f}(S) = \frac{1}{2^n} \sum_{(a_1,\ldots,a_n)\in\{0,1\}^n} (-1)^{f(a_1,\ldots,a_n)} \cdot (-1)^{a_{i_1}+\ldots+a_{i_k}}$$

(here all the computations are done over the reals). It is known that

$$(-1)^{f(x_1,\ldots,x_n)} = \sum_{S\subseteq\{1,\ldots,n\}} \hat{f}(S) \cdot (-1)^{\sum_{i\in S} x_i},$$

and the expression on the right side is called the *Fourier expansion* of f.

There is a quantum algorithm that given a set $S \subseteq \{1, \ldots, n\}$ 'computes' in a certain sense Fourier coefficient $\hat{f}(S)$. On the other hand, computing a Fourier coefficient reduces to computing the number of solutions of the equation $f(x_1, \ldots, x_n) + x_{i_1} + \ldots + x_{i_k} = 0$. Indeed, if we denote this number by N then $\hat{f}(S) = \frac{1}{2^n}(2N - 2^n)$. Now, the function f is usually given by a circuit, so if this circuit is a 3-CNF, computing the Fourier coefficient is equivalent to #3-SAT.

Example 5 (Max-Cut). Recall that a *cut* of a graph $G = (V, E)$ is a partition of V into two subsets, W_1, W_2. A cut is *maximal* if the number of edges whose ends belong to different sets W_1, W_2 is maximal possible. In the #MAX-CUT problem the goal is, given a graph G, find the number of maximal cuts.

Although this problem is best representable through so called *weighted CSP*, it also admits representation as a counting CSP. However, such representation requires certain technicalities and we omit it here. More about #MAX-CUT and counting CSPs can be found [4].

Co-clones and weak co-clones. In an instance of a constraint satisfaction problem some possible combinations of values of variables are given explicitly by constraints, while other dependencies arise implicitly through interaction of constraints.

Example 6 ([5]). Let Γ be a constraint language containing a single binary relation R over the set $D = \{0, 1, 2\}$, where R is given by $R = \{(0, 0), (0, 1), (1, 0), (1, 2), (2, 1), (2, 2)\}$. Consider the following instance of $\#\mathrm{CSP}(\Gamma)$

$$\mathcal{P} = (\{v_1, v_2, v_3, v_4\}, D, \{C_1, C_2, C_3, C_4, C_5\}),$$

where $C_1 = \langle (v_1, v_2), R\rangle$, $C_2 = \langle (v_1, v_3), R\rangle$, $C_3 = \langle (v_2, v_3), R\rangle$, $C_4 = \langle (v_2, v_4), R\rangle$, $C_5 = \langle (v_3, v_4), R\rangle$. There is no explicit constraint on the pair (v_1, v_4). However, by considering all solutions to $\mathcal{P}$, it can be shown that the possible pairs of values which can be taken by this pair of variables are precisely the elements of the relation $R' = R \cup \{(1, 1)\}$.

The *co-clone* generated by a constraint language Γ includes all relations that can be implicitly expressed by an instance of $\#\mathrm{CSP}(\Gamma)$. More precisely,

Definition 2 *For a constraint language Γ over a set D, the set $\langle\langle\Gamma\rangle\rangle$ includes all relations that can be expressed (as a predicate) using (a) relations from Γ, together with the binary equality relation $=_D$ on D, (b) conjunctions, and (c) existential quantification. This set is called the* co-clone generated by Γ*, and every relation from $\langle\langle\Gamma\rangle\rangle$ is said to be* primitive positive *(*pp-*)* definable *in Γ.*

Weak co-clone $\langle\Gamma\rangle$ generated by Γ *is obtained in a similar way by disallowing existential quantification. $\langle\Gamma\rangle$ includes all relations that can be expressed using (a) relations from Γ, together with $=_D$, and (b) conjunctions,*

Example 7. (1) The relation R' in Example 6 can be expressed as follows

$$R'(x, y) = \exists z, t(R(x, z) \wedge R(x, t) \wedge R(z, t) \wedge R(z, y) \wedge R(t, y)).$$

Hence, $R' \in \langle\langle\{R\}\rangle\rangle$.

(2) Let $\neq_D$ be the binary disequality relation on the set $D = \{0, 1, 2\}$, that is, $\neq_D = \{(a, b) \in D^2 \mid a \neq b\}$ and Q the ternary disequality relation on the same set, $(a, b, c) \in R \iff \{a, b, c\} = \{0, 1, 2\}$. Then

$$Q(x, y, z) = (\neq_D (x, y) \wedge \neq_D (y, z) \wedge \neq_D (z, x)),$$

and therefore $Q \in \langle\{\neq\}\rangle$.

Galois connections. Co-clones and weak co-clones can often be conveniently and concisely represented through functions and partial functions, respectively.

Let R be a (k-ary) relation on a set D, and $f\colon D^n \to D$ an n-ary function on the same set. Function f *preserves* R, or is a *polymorphism* of R, if for any n tuples $\mathbf{a}_1, \ldots, \mathbf{a}_n \in R$ the tuple $f(\mathbf{a}_1, \ldots, \mathbf{a}_n)$ obtained by component-wise application of f also belongs to R. Relation R in this case is said to be *invariant* with respect to f. The set of all functions that preserve every relation from a constraint language Γ is denoted by $\mathsf{Pol}(\Gamma)$ and the set of all relations invariant with respect to a set of functions C is denoted by $\mathsf{Inv}(C)$.

Example 8. Let R be the solution space of a system of linear equations over a field F. Then the function $m(x, y, z) = x - y + z$ is a polymorphism of R. Indeed, let $A\mathbf{x} = \mathbf{b}$ be the system defining R, and $\mathbf{x}, \mathbf{y}, \mathbf{z} \in R$. Then

$$A\, m(\mathbf{x}, \mathbf{y}, \mathbf{z}) = A(\mathbf{x} - \mathbf{y} + \mathbf{z}) = A\mathbf{x} - A\mathbf{y} + A\mathbf{z} = \mathbf{b}.$$

In fact, the converse can also be shown: if R is invariant under m then it is the solution space of a certain system of linear equations.

Operators Inv and Pol form a *Galois connection* between sets of functions and sets of relations. Sets of the form $\mathsf{Inv}(C)$ are precisely co-clones; on the operational side there is another type of closed sets.

A set of functions is said to be a *clone* of functions if it is closed under superposition and contain all the *projection* functions, that is, functions of the form $f(x_1, \ldots, x_n) = x_i$. It is known [9, 25] that sets of functions of the form $\mathsf{Pol}(\Gamma)$ are exactly clones of functions.

As we shall see, the study of the #CSP can also make use of another Galois connection, a connection between weak co-clones and sets of *partial functions*. An n-ary partial function f on a set D is just a partial mapping $f\colon D^n \to D$. As in the case of total functions, a partial function f *preserves* relation R, if for any n tuples $\mathbf{a}_1, \ldots, \mathbf{a}_n \in R$ the tuple $f(\mathbf{a}_1, \ldots, \mathbf{a}_n)$ obtained by component-wise application of f is either undefined or belongs to R. The set of all partial functions that preserve every relation from a constraint language Γ is denoted by $\mathsf{pPol}(\Gamma)$.

The set of all tuples from D^n on which f is defined is called the *domain* of f and denoted by $\mathsf{dom}(f)$. A set of functions is said to be *down-closed* if along with a function f it contains any function f' such that $\mathsf{dom}(f') \subseteq \mathsf{dom}(f)$ and $f'(a_1, \ldots, a_n) = f(a_1, \ldots, a_n)$ for every tuple $(a_1, \ldots, a_n) \in \mathsf{dom}(f')$. A down-closed set of functions, containing all projections and closed under superposition is called a *partial clone*. Partial clones are exactly the sets of the form $\mathsf{pPol}(\Gamma)$ for a certain Γ, and the weak co-clones are precisely the sets $\mathsf{Inv}(C)$ for collections C of partial functions [15].

2 Exact Counting

We now turn to the the complexity of the #CSP.

Complexity and algorithms. Complexity classes of counting problems are somewhat similar to those for usual decision problems. We need two of them, the class FP of counting problems (actually, of problems computing a function) solvable in polynomial time, and the class #P of problems in which the goal is to find the number of solutions to a problem from NP. There are several standard types of reductions between counting problems. The most widely accepted one is *Turing* reduction. Suppose we have counting problems A and B. A Turing reduction is a polynomial time algorithm that solves A using B as an *oracle*, or a subroutine. A counting problem $A \in$#P is said to be *#P-complete* if any problem from #P is Turing reducible to A. More details about counting complexity classes and reductions can be found in [24]. For the purpose of this paper we consider problems from FP as easy (*tractable*), and #P-complete problems as hard. We assume that P$\neq$NP.

The complexity of the counting problems considered in Examples 1–5 is as follows: #3-SAT is #P-complete, the #H-Coloring problem is tractable for certain graphs and #P-complete otherwise, see Theorem 4, #System-of-Linear-Equations is tractable, computing Fourier coefficient is tractable if and only if f can be represented by a quadratic polynomial over the 2-element field, and #P-complete otherwise, finally #Max-Cut is #P-complete [26, 27, 14, 22].

Boolean #CSP. The simplest type of constraint languages are those over a 2-element set. They are usually referred to as *Boolean* constraint languages. For a Boolean language Γ the problem #CSP(Γ) can be thought of as a generalized version of the Satisfiability problem.

Theorem 3 (Creignou, Hermann [7, 8]). *Let Γ be a Boolean constraint language. Problem* #CSP(Γ) *is tractable if and only if every relation from Γ is the set of all solutions to a system of linear equations over the 2-element field. Otherwise it is #P-complete.*

Observe that the latter condition holds if and only if the *affine* operation $x - y + z$ (addition and subtraction modulo 2) is a polymorphism of Γ.

Corollary 1. *For a Boolean constraint language Γ the problem* #CSP(Γ) *is tractable if and only if* $\mathsf{Pol}(\Gamma)$ *contains the affine operation.*

Non-Boolean #CSP. For non-Boolean #CSPs researchers have mostly focused on the #H-Coloring problem.

Theorem 4 (Dyer, Greenhill [13]). *The #H-Coloring problem for an undirected graph H is tractable if and only if every connected component of H is an single vertex, or a complete bipartite graph without loops, or a complete graph with all loops present. Otherwise it is #P-complete. Moreover, the #P-complete problems remain #P-complete even if restricted to graphs of degree at most 3.*

In [3] it is shown that an undirected graph satisfies the conditions of Theorem 4 if and only if its edge relation has a *Mal'tsev* operation as a polymorphism, that is, an function m satisfying equations $m(x, y, y) = m(y, y, x) = x$ for any x, y.

A *directed acyclic graph* (DAG) is a digraph without directed cycles.

Theorem 5 (Dyer, Goldberg, Paterson [12]). *For any DAG H the #H-*Coloring *problem is either tractable or #P-complete. The DAGs that give rise to polynomial time solvable problems are those satisfying the* Lovasz-goodness *condition, see, [12] for details.*

Observe that in all the cases considered above every counting CSP is either tractable or #P-complete. This suggests a natural question: Is this true for any problem of the form #CSP(Γ)? This question (similar questions also arise for other types of constraint satisfaction problems), or rather a positive resolution of it, is known as the *dichotomy conjecture* for counting problems. In the rest of this section we survey a sequence of results that eventually lead to a proof of the dichotomy conjecture.

Although all the results listed above are very non-trivial, the methods they use strongly depend on the nature of problems and the type of digraphs considered. To attack the dichotomy conjecture a more universal approach is needed. To develop such approach we make use of clones and co-clones.

Clones and complexity. The following theorem shows a link between the complexity of counting CSPs and clones.

Theorem 6 (Bulatov, Dalmau [3]). *Let Γ_1, Γ_2 be constraint languages over the same finite set such that Γ_2 is finite and every relation from Γ_2 is pp-definable in Γ_1. Then* #CSP(Γ_2) *is Turing reducible to* #CSP(Γ_1)*. In particular, if* #CSP(Γ_1) *is tractable then so is* #CSP(Γ_2)*; similarly, if* #CSP(Γ_2) *is #P-complete then so is* #CSP(Γ_1)*.*

Due to the Galois connection between clones and co-clones Theorem 6 can be restated as follows.

Corollary 2. *Let Γ_1, Γ_2 be constraint languages over the same finite set and Γ_2 is finite. If* $\mathsf{Pol}(\Gamma_1) \subseteq \mathsf{Pol}(\Gamma_2)$ *then* #CSP(Γ_2) *is Turing reducible to* #CSP(Γ_1)*.*

Adding several other algebraic results we obtain the first general necessary condition for tractability of counting CSPs.

Theorem 7 (Bulatov, Dalmau [3]). *If Γ does not have a Mal'tsev polymorphism then* #CSP(Γ) *is #P-complete.*

Congruences and a dichotomy theorem. Theorem 7 implies almost straightforwardly Theorems 3 and 4. However, the existence of a Mal'tsev polymorphism is not a sufficient condition for tractability as the following example shows.

Example 9. Consider the graphs shown in Fig. 1. They look very similar and they both have a Mal'tsev polymorphism. Intuitively, if the vertices of the middle layer are treated as elements of the direct sum of two copies of a 2-element group (the two copies of the leftmost vertex of H_2 should be identified), then such polymorphism acts as $x - y + z$ on the middle layer. It then can be extended straightforwardly on the vertices of other layers. (Clearly for the second graph one should be careful with the two copies of the leftmost vertex, but this obstacle can easily be handled.) However, the #H_1-Coloring problem is tractable, while #Max-Cut is reducible to #H_2-Coloring, and therefore it is #P-complete.

Let Γ be a constraint language over a set D and $R \in \langle\langle\Gamma\rangle\rangle$ a k-ary relation pp-definable in Γ. A *congruence* of R is a $2k$-ary relation Q pp-definable in Γ and satisfying the following conditions: (a) for any $\mathbf{a} \in Q$ the tuples $(\mathbf{a}[1], \dots, \mathbf{a}[k])$ and $(\mathbf{a}[k+1], \dots, \mathbf{a}[2k])$ belong to R, thus, Q can be viewed as a binary relation on R; (b) Q viewed as a binary relation on R is an equivalence relation. Now, let Q, Q_1, Q_2 be congruences of R such that $Q \subseteq Q_1, Q_2$, and let $A_1, \dots, A_m$ and $B_1, \dots, B_n$ be equivalence classes of Q_1

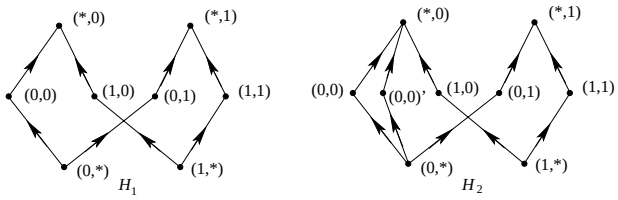

Fig. 1.

and Q_2, respectively. By M_{ij} we denote the number of Q-classes in $A_i \cap B_j$, and $M(R; Q_1, Q_2; Q)$ denotes the matrix (M_{ij}). The constraint language Γ is said to be *congruence singular* if, for any relation $R \in \langle\langle \Gamma \rangle\rangle$, and any congruences Q, Q_1, Q_2 of R with $Q \subseteq Q_1, Q_2$, the *row rank* of the matrix $M(R; Q_1, Q_2; Q)$ equals the number of classes of the smallest equivalence relation containing both Q_1 and Q_2.

Theorem 8 (Bulatov [2]). *The problem* $\#\mathrm{CSP}(\Gamma)$ *is tractable if and only if* Γ *is congruence singular; otherwise it is #P-complete.*

Example 10 (Example 9 continued). Let $H_2 = (V, E)$. The unary relation $C = \{(0,0), (0,0)', (1,0), (0,1), (1,1)\}$ is pp-definable in $\{E\}$; indeed, $C(x) = \exists y, z(E(y,x) \wedge E(x,z))$. The equivalence relations on this set Q_1, Q_2 with classes $A_1 = \{(0,0), (0,0)', (0,1)\}$, $A_2 = \{(1,0), (1,1)\}$, and $B_1 = \{(0,0), (0,0)', (1,0)\}$, $B_2 = \{(0,1), (1,1)\}$, respectively, are congruences of C; say,

$$Q_1(x,y) = \exists z(C(x) \wedge C(y) \wedge E(z,x) \wedge E(z,y)).$$

Then $M(C; Q_1, Q_2; =_C) = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}$. Clearly its rank equals 2, while the smallest equivalence relation on C containing both Q_1 and Q_2 is the total relation and so has only one class. Therefore $\#H_2$-Coloring problem is #P-complete.

3 Approximation

As it is seen from the results above, tractable counting CSPs are rare. Other efficiently solvable counting problems are also scarce. Of those that do not fall under one of the CSP type considered above two can be mentioned: the #Spanning Tree problem and the #Planar Graph Matching [19]. (Observe that both are reducible to solving systems of linear equations.) This is why approximation algorithms look so appealing for solving counting problems. However, the study of approximation of counting problems is in its very early stage. Here we list several initial results in this direction.

Relative error and classes of problems. Let A be a counting problem. An algorithm Alg is said to be an *approximation algorithm* for A with relative error ε (which may depend on the size of the input) if it is polynomial time and for any instance $\mathcal{P}$ of A it outputs a certain number $\mathsf{Alg}(\mathcal{P})$ such that

$$\frac{|\#\mathcal{P} - \mathsf{Alg}(\mathcal{P})|}{\#\mathcal{P}} < \varepsilon,$$

where $\#\mathcal{P}$ denotes the exact number of solutions to $\mathcal{P}$. Algorithm Alg is *randomized* approximation algorithm, if it uses some source of randomness and for any instance $\mathcal{P}$ the above inequality is true with probability at least $\frac{2}{3}$.

The following framework is viewed as one of the most realistic models of efficient computations. A *fully polynomial approximation scheme* (FPAS, for short) for a problem A is an algorithm Alg such that: It takes as input an instance $\mathcal{P}$ of A and a real number $\varepsilon > 0$, the relative error of Alg on the input $\mathcal{P}, \varepsilon$ is less than ε,

and Alg is polynomial in the size of $\mathcal{P}$ and $\frac{1}{\varepsilon}$. A FPAS is called a *fully polynomial randomized approximation scheme* (FPRAS) if Alg is randomized and the above inequality is satisfied with probability at least $\frac{3}{4}$.

To determine the approximation complexity of problems another type of reductions is used. Suppose A and B are two counting problems whose complexity (of approximation) we want to compare. An *approximation preserving reduction* or *AP-reduction* from A to B is a probabilistic algorithm Alg, using B as an oracle, that takes as input a pair $(\mathcal{P}, \varepsilon)$ where $\mathcal{P}$ is an instance of A and $0 < \varepsilon < 1$, and satisfies the following three conditions: (i) every oracle call made by Alg is of the form $(\mathcal{P}', \delta)$, where $\mathcal{P}'$ is an instance of B, and $0 < \delta < 1$ is an error bound such that $\frac{1}{\delta}$ is bounded by a polynomial in the size of $\mathcal{P}$ and $\frac{1}{\varepsilon}$; (ii) the algorithm Alg meets the specifications for being randomized approximation scheme for A whenever the oracle meets the specification for being randomized approximation scheme for B; and (iii) the running time of Alg is polynomial in the size of I and $\frac{1}{\varepsilon}$. If an approximation preserving reduction from A to B exists we write $A \leq_{\mathrm{AP}} B$, and say that A is *AP-reducible to* B. If $A \leq_{\mathrm{AP}} B$ and $B \leq_{\mathrm{AP}} A$ then we say that A and B are *AP-interreducible*, and write $A \equiv_{\mathrm{AP}} B$.

Approximation complexity of #CSP. Up to AP-reductions all #CSPs whose complexity is known cluster into 5 groups (it mostly follows from [12, 10, 11]).

The first group consists of those problems solvable in polynomial time, and the members of this group are completely described in the previous section.

The second group is formed by problems solved by an FPRAS, and it contains problems #Dual SAT, in which the goal is to count satisfying assignments of a propositional formula in DNF, and #H-Coloring on graphs with bounded vertex degree [18, 23].

A representative of the third group is #Bipartite-Independent-Set, or #BIS, for short, in which the goal is to find the number of *independent sets* in a given bipartite graph. The #BIS problem can be represented as the #H-Coloring problem, where H is a path of length 3. Every independent set of a connected bipartite graph G corresponds to two homomorphisms from G to H, while a non-bipartite graph does not have homomorphisms to H.

The fourth group is formed by problems AP-interreducible with the #Bipartite-3-Coloring problem: Given a bipartite graph, find the number of its 3-colorings.

Finally, the fifth group contains most difficult problems in #P, they are interreducible with #SAT and called *#P-complete*.

In the case of Boolean constraint languages Γ the clustering above can be refined to a complete characterization of approximation complexity of #CSP(Γ). By C_0, C_1 we denote unary relations over $\{0, 1\}$ that have only one element 0 or 1, respectively. By O we denote the binary relation which is the natural order on $\{0, 1\}$.

Theorem 9 (Dyer, Goldberg, Jerrum [11]). *Let Γ be a constraint language over $\{0, 1\}$. If every relation in Γ is the set of solutions of a system of linear equations, #CSP(Γ) is tractable. Otherwise if $\Gamma \subseteq \langle\langle C_0, C_1, O\rangle\rangle$ then #CSP(Γ) is interreducible with #BIS. Otherwise #CSP(Γ) is interreducible with #SAT, that is, #P-complete.*

Theorem 9 hints that the approximation complexity of counting CSPs may be again determined by the co-clone generated by a constraint language (or its polymorphisms). However, it remains an open question whether or not an analog of Theorem 6 is true. We can, however, prove a weaker result.

Theorem 10. *Let Γ_1, Γ_2 be constraint languages over the same finite set such that Γ_2 is finite and every relation from Γ_2 belongs to $\langle\Gamma_1\rangle$. Then #CSP(Γ_2) is AP-reducible to #CSP(Γ_1).*

Proof: Suppose $\Gamma_2 = \{R_1, \ldots, R_k\}$. Then every ($n_i$-ary) R_i is defined by a formula $\Phi_i(x_1, \ldots, x_{n_i})$ that uses relations from Γ_1, the equality relation and conjunctions. It will be convenient to view Φ_i as an instance $\mathcal{P}_i = (\{x_1, \ldots, x_{n_i}\}, \mathcal{C}_i)$ of the CSP, where each predicate from Φ_i represents a constraint, and the relation R_i as the set of all its solutions.

Let $=$ denote the equality relation. We first show that #CSP(Γ_2) $\leq_{AP}$ #CSP($\Gamma_1 \cup \{=\}$). Take an instance $\mathcal{P} = (V, \mathcal{C})$ of #CSP(Γ_2). The corresponding instance of #CSP(Γ_1) is constructed as follows: The

set of variables remains V; for each constraint $C = \langle \mathbf{s}, R_i \rangle \in \mathcal{C}$ we include all the constraints from $\mathcal{C}_i$ replacing every variable x_j in their scopes with $\mathbf{s}[j]$. Denote the resulting instance by $\mathcal{P}'$. It is straightforward that instances $\mathcal{P}$ and $\mathcal{P}'$ have the same solutions. Therefore, the algorithm that on input $(\mathcal{P}, \varepsilon)$ calls the oracle $\#\mathrm{CSP}(\Gamma_1)$ with input $(\mathcal{P}', \varepsilon)$, and then passes the oracle's answer to output, satisfies all the requirements for AP-reductions.

Now, to reduce $\#\mathrm{CSP}(\Gamma_1 \cup \{=\})$ to $\#\mathrm{CSP}(\Gamma_1)$ it suffices to, first, find sets $W_1, \ldots, W_\ell$ of variables that are forced to be assigned the same values by the equality constraints, and, second, replace every occurrence of variable $x \in W_i$ with a variable y_i. Again, it is straightforward that the original and the resulting instances have the same number of solutions, and that all the requirements for AP-reductions are respected. □

Due to the Galois connection between sets of relations and sets of partial clones [15], partial functions are in control of the approximation complexity of counting problems.

4 Approximation complexity of maximal partial clones on a 3-element set

In this section we prove some initial results linking partial clones of functions and the approximation complexity of counting problems. The description of maximal partial clones [16] is one of the most celebrated results in the study of partial clones, and one of not so many that give us information about the structure of particular partial clones. It is also remarkable that the family of counting problems that arise from maximal partial clones contains the whole spectrum of approximation complexities known so far, except for FPRAS. In particular, the two problems, #BIS and #Bipartite-3-Coloring, whose approximation complexity is not quite known, can be represented as $\#\mathrm{CSP}(R)$ where $\mathsf{pPol}(R)$ is a maximal partial clone. In the first case R is the relation on a 4-element set given by a 3-path; in the second case, R is the relation given by a 6-cycle. Below we shall see some maximal partial clones that give rise to polynomial time solvable and #P-complete problems. This motivates the study of the approximation complexity of maximal partial clones. In this paper we obtain a partial classification of such clones on a 3-element set.

Table 1 lists all maximal partial clones on $\{0, 1, 2\}$ up to permutations of this set. Binary relations from this table are often convenient to treat as digraphs. The digraphs of binary relations from Table 1 are shown in Fig. 2. The main result of this section is the following theorem.

Theorem 11. *The approximation complexity of maximal partial clones 1–17 is as shown in Table 1. There FP stands for problems solvable in polynomial time, #BIS stands for problems AP-reducible with the* #Bipartite-Independent-Set *problem, and #P-Complete for #P-complete problems.*

The rest of this section is dedicated to proving Theorem 11.

4.1 Constructions and reductions

We will need several auxiliary statements. The first of them generalizes the Pinning Lemma from [11]. This lemma allows one to add an extra unary relation to the constraint language. The proof of Lemma 1 also follows closely the proof in [11]. This is why we omit a portion of computation in our proof. It is convenient to consider a subset of the domain as a unary relation.

Lemma 1 (Extended Pinning). *Let Γ be a constraint language over the set $D = \{0, 1, \cdots, k-1\}$, and let for a certain subset $S \subset D$ there be an ℓ-ary relation $R \in \Gamma$ and a coordinate position j, $1 \le j \le \ell$, such that for any $a \in S$ the relation R has more tuples $\mathbf{a}$ with $\mathbf{a}[j] = a$ than tuples $\mathbf{b}$ with $\mathbf{b}[j] \notin S$. Then $\#\mathrm{CSP}(\Gamma \cup \{S\}) \le_{AP} \#\mathrm{CSP}(\Gamma)$.*

Proof: Fix an ℓ-ary relation $R \in \Gamma$, and a coordinate position j such that R and j satisfy the conditions of the lemma. Let also w be the minimal (over elements $a \in S$) number of tuples $\mathbf{a}$ such that $\mathbf{a}[j] = a$, and let w' be the number of tuples $\mathbf{b}$ with $\mathbf{a}[j] \notin S$. By the conditions of the lemma $w' < w$.

Table 1. Maximal partial clones on $\{0,1,2\}$

#	R_i	Complexity	#	R_i	Complexity
1	$\{0\}$	FP	15	$\begin{pmatrix}0&1&2\\1&2&0\\2&0&1\end{pmatrix}$	FP
2	$\{0,1\}$	FP	16	$\begin{pmatrix}0&0&1&1&2&2\\1&2&0&2&0&1\\2&1&2&0&1&0\end{pmatrix}$	#P-Complete
3	$\begin{pmatrix}0\\1\end{pmatrix}$	FP	17	$\begin{pmatrix}0&1&2&0\\0&1&2&1\\0&1&2&2\end{pmatrix}$	$\geq_{AP} \#BIS$
4	$\begin{pmatrix}0&1\\1&0\end{pmatrix}$	FP	18	$\begin{pmatrix}0&1&2&0&0\\0&1&2&1&2\\0&1&2&2&1\end{pmatrix}$	$\geq_{AP} \#BIS$
5	$\begin{pmatrix}0&0\\1&2\end{pmatrix}$	FP	19	$\begin{pmatrix}0&1&2&0&1&2\\0&1&2&1&2&0\\0&1&2&2&0&1\end{pmatrix}$	
6	$\begin{pmatrix}0&1&0&2\\1&0&2&0\end{pmatrix}$	FP	20	$\begin{pmatrix}0&1&2&0&0&1&1&2&2\\0&1&2&1&2&0&2&0&1\\0&1&2&2&1&2&0&1&0\end{pmatrix}$	
7	$\begin{pmatrix}0&1&2&0\\0&1&2&1\end{pmatrix}$	#BIS	21	$\{(0,1,2),(a,a,b)\|a,b\in E_3\}$	
8	$\begin{pmatrix}0&1&2&0&1\\0&1&2&1&0\end{pmatrix}$	FP	22	$\{(0,1,2),(1,0,2),(a,a,b)\|a,b\in E_3\}$	
9	$\begin{pmatrix}0&1&2&0&0\\0&1&2&1&2\end{pmatrix}$	#BIS	23	$\{(0,1,2),(2,1,0),(a,b,a)\|a,b\in E_3\}$	
10	$\begin{pmatrix}0&1&2&0&2\\0&1&2&1&0\end{pmatrix}$	#BIS	24	$\{(0,1,2),(0,2,1),(b,a,a)\|a,b\in E_3\}$	
11	$\begin{pmatrix}0&1&2&0&0&1\\0&1&2&1&2&2\end{pmatrix}$	#BIS	25	$\{(a,a,b,b),(a,b,a,b)\|a,b\in E_3\}$	
12	$\begin{pmatrix}0&1&2&0&1&0&2\\0&1&2&1&0&2&0\end{pmatrix}$	#BIS	26	$\{(a,a,b,b),(a,b,a,b),(a,b,b,a)\|a,b\in E_3\}$	
13	$\begin{pmatrix}0\\1\\2\end{pmatrix}$	FP	27	$\{(a,b,c)\in E_3^3 \|\|\{a,b,c\}\|\leq 2\}$	
14	$\begin{pmatrix}2&2\\1&0\\0&1\end{pmatrix}$	FP	28	$P_2\cup[c_\infty]$	

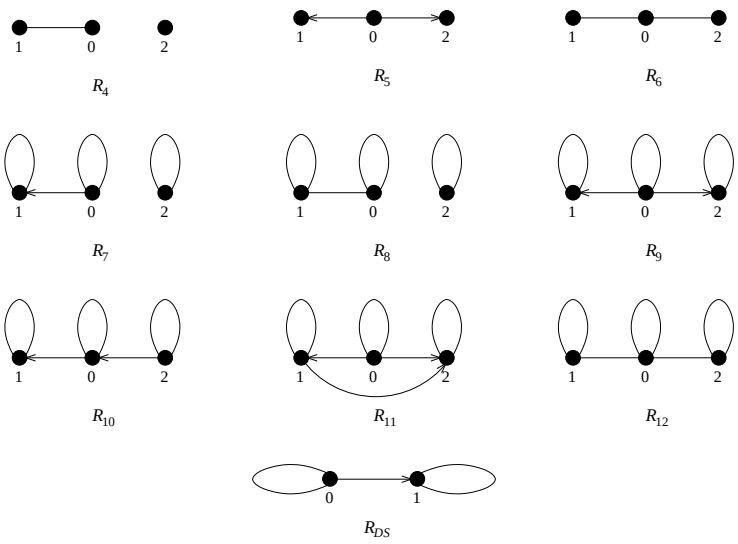

Fig. 2. Binary relations from Table 1

Consider an instance $\mathcal{P}$ of #CSP($\Gamma \cup \{S\}$) with n variables. Let N_S be the set of variables which occur in the scope of constraints of $\mathcal{P}$ with relation S. Set $n_S = |N_S|$ and $m = \lceil (n+2)/\lg(w/w') \rceil$. Construct an instance $\mathcal{P}'$ of #CSP(Γ) as follows:

- The set of variables of $\mathcal{P}'$ includes all variables from $\mathcal{P}$, and also, for each variable $x \in N_S$, any $u \in \{1, 2, \cdots, m\}$, and any $v \in \{0, 1, \cdots, k\} - \{j\}$ a fresh variable $x_{u,v}$.
- Include all constraints from $\mathcal{P}$ other than those involving S.
- For each constraint $C = \langle (x), S \rangle$ from $\mathcal{P}$ include m constraints whose relation is R, variable x occupies the jth position in the scope, and the variable $x_{u,v}$ is in the vth position of the uth constraint.

Now any solution of $\mathcal{P}$ can be extended in at least w^{mn_S} ways to a solutions of $\mathcal{P}'$, provided all variables from N_S take values from S. On the other hand, every assignment that does not satisfy this condition can be extended in at most $w^{m(n_S-1)}w'^m$ ways. There exist no more than k^n such solutions. Therefore if N and N' denote the number of solutions to $\mathcal{P}$ and $\mathcal{P}'$, respectively, then

$$Nw^{mn_S} \leq N' \leq Nw^{mn_S} + k^n w^{m(n_s-1)} w'^m.$$

So, for a properly chosen m,

$$N \leq \frac{N'}{w^{mn_S}} \leq N + \frac{1}{4},$$

which implies

$$N = \left\lfloor \frac{N'}{w^{mn_S}} \right\rfloor.$$

Now we can complete the proof in exactly the same way as in [11]. □

The next lemma shows that when reducing problem #CSP(Γ) to #CSP(Γ') we can restrict ourselves to *connected* instances $\mathcal{P}$ of #CSP(Γ). The instance $\mathcal{P}$ is connected if the hypergraph formed by the scopes of constraints in $\mathcal{P}$ is connected.

Let $\mathcal{P} = (V, \mathcal{C})$ be an instance of #CSP(Γ). Let Hyp($\mathcal{P}$) be the hypergraph with the vertex set V and such that it contains a hyperedge $\{v_1, \ldots, v_\ell\}$ if and only if there is a constraint in $\mathcal{C}$ with the scope $(v_1, \ldots, v_\ell)$. Suppose that $V_1, \ldots, V_r$ are the connected components of Hyp($\mathcal{P}$). Then these components induce instances

$\mathcal{P}_1, \ldots, \mathcal{P}_r$, where $\mathcal{P}_t = (V_t, \mathcal{C}_t)$ and $\mathcal{C}_t$ contains all constraints from $\mathcal{C}$ with scopes containing variables from V_t. Instances $\mathcal{P}_t$ will be called *connected components* of $\mathcal{P}$.

Let also $\#\mathrm{CSP}_c(\Gamma)$ denote the problem $\#\mathrm{CSP}(\Gamma)$ restricted to connected instances.

Lemma 2. *For any constraint language Γ the problem $\#\mathrm{CSP}(\Gamma)$ is AP-interreducible with $\#\mathrm{CSP}_c(\Gamma)$.*

Proof: The reduction of $\#\mathrm{CSP}_c(\Gamma)$ to $\#\mathrm{CSP}(\Gamma)$ is trivial. Now, let $\mathcal{P}$ be an instance of $\#\mathrm{CSP}(\Gamma)$, and let $\mathcal{P}_1, \ldots, \mathcal{P}_r$ be its connected components. Take $\varepsilon > 0$ and set $\delta = \frac{\varepsilon}{2r}$. Our reduction, given an instance $(\mathcal{P}, \varepsilon)$ calls the algorithm for $\#\mathrm{CSP}_c(\Gamma)$ on instances $(\mathcal{P}_1, \delta), , \ldots, (\mathcal{P}_r, \delta)$ and outputs $N = N_1 \cdot, \ldots, \cdot N_r$, where N_i is the answer given by the oracle on $(\mathcal{P}_i, \delta)$.

We claim that the above reduction is an AP-reduction. First of all, observe that it is polynomial time, and the instances it produces satisfy the conditions of AP-reductions. It remains to show that if the oracle approximate the solution within relative error δ then the reduction provides approximation within ε.

Since we can assume ε small, we have $(1-\delta)^r \geq 1 - 2r\delta = 1 - \varepsilon$ and $(1+\delta)^r \leq 1 + 2r\delta = 1 + \varepsilon$. Now if the actual solutions to $\mathcal{P}$, $\mathcal{P}_1, \ldots, \mathcal{P}_r$ are N', $N'_1, \ldots, N'_r$, then we obviously have $N' = N'_1 \cdot \ldots \cdot N'_r$. Also

$$1 - \delta < \frac{N_1}{N'_1} < 1 + \delta,$$

Therefore

$$\frac{N}{N'} < \frac{N_1 \cdot \ldots \cdot N_r}{N'_1 \cdot \ldots \cdot N'_r} < (1+\delta)^r \leq 1 + \varepsilon.$$

The lower bound is proved similarly. □

Lemma 3. *Let φ be a polynomial time computable function that maps every instance of a counting problem A to an instance of a counting problem B in such a way that there are constants $d > 0$ and c (not necessarily positive) such that for any A-instance $\mathcal{P}$ we have $\#\mathcal{P} = d \cdot \#\varphi(\mathcal{P}) + c$. Then if any instance of A has at least one solution (in the case $c < 0$), or every instance of B has at least one solution (in the case $c > 0$), then there is an AP-reduction from A to B.*

Proof: The AP-reduction works as follows: On an instance $\mathcal{P}$ of A and $\varepsilon > 0$ it makes an oracle call $(\varphi(\mathcal{P}), \delta)$ to B, where $\delta = \frac{\varepsilon}{p}$ and $p = 1 + |c|/d$ if $|c|/d$ is integer, and $p = 1 + \frac{|c|}{d\lceil |c|/d \rceil - c/d}$ otherwise, and if the oracle's reply is N, it returns $dN + c$. Clearly the algorithm makes polynomially many steps and oracle calls, and the oracle request is of the correct form. Thus, it suffices to show that if the oracle's solution is within relative error of δ then the algorithm gives an ε-approximation of $\#\mathcal{P}$.

Let the exact and approximation solutions for $\varphi(\mathcal{P})$ be N' and N, respectively; then the exact and approxmation solutions for $\mathcal{P}$ are $dN' + c$ and $dN + c$, respectively. Since $\#\mathcal{P} > 0$, we can assume $dN + c, dN' + c, N, N' > 0$. Observe that $\frac{dN'}{dN'+c} \leq p$. Indeed, if $c > 0$ this fraction does not exceed 1. If $c < 0$ then it follows from the assumption $dN' + c > 0$ which implies $N' > |c|/d$. Thus the relative error can be bounded by

$$\frac{dN + c}{dN' + c} \leq \frac{(1+\delta)dN' + c}{dN' + c} = 1 + \frac{\delta dN'}{dN' + c} \leq 1 + \frac{\varepsilon}{p} p = 1 + \varepsilon.$$

The inequality $1 - \varepsilon \leq \frac{dN+c}{dN'+c}$ is similar. □

Let R be a k-ary relation and $S = \{i_1, \ldots, i_\ell\} \subseteq \{1, \ldots, k\}$. By $\mathrm{pr}_S R$ we denote the *projection* of R onto the set S of its coordinate positions, that is, the relation $\{(a_{i_1}, \ldots, a_{i_\ell}) \mid (a_1, \ldots, a_k) \in R\}$. Observe that $\mathrm{pr}_S R$ can be pp-defined in R by quantifying away all coordinate positions of R except for those in S. Although existential quantification is not known to give rise to AP-reducible problems, in certaint cases it does.

Lemma 4. *If Γ is a constraint language and for some k-ary relation $R \in \Gamma$ there is a set $S \subset \{0, 1, \cdots, k-1\}$ such that $|\mathrm{pr}_S R| = |R|$ then $\#CSP(\Gamma \cup \{\mathrm{pr}_S R\}) \leq_{AP} \#CSP(\Gamma)$.*

Proof: The AP-reduction is constructed as follows: Given an instance $\mathcal{P}$ of $\#CSP(\Gamma \cup \{\mathrm{pr}_S R\})$ with variable set V we define an instance $\mathcal{P}'$ of $\#CSP(\Gamma)$ such that it has the same number of solutions. Let $m = k = |S|$. The instance $\mathcal{P}'$ includes all the variables of $\mathcal{P}$ and all its constraints except for those having $\mathrm{pr}_S R$ as the constraint relation. For each constraint in $\mathcal{P}$ of the form $\langle (v_1, \dots, v_\ell), \mathrm{pr}_S R \rangle$ we introduce a constraint $\langle (w_1, \dots, w_k), R \rangle$, where $w_i = v_{i_j}$ if $i \in S$ and $i = i_j$, or a new variable that does not appear in any other constraint scope.

Clearly, the restriction of any solution of $\mathcal{P}'$ onto V is a solution of $\mathcal{P}$. Furthermore, the condition $|\mathrm{pr}_S R| = |R|$ implies that any solution of $\mathcal{P}$ can be extended to a solution of $\mathcal{P}'$ in a unique way. The lemma is proved. □

4.2 Polynomial time cases

Most of the polynomial time cases in our classification follow from the next simple observation.

Lemma 5. *If Γ is in a weak co-clone generated by a set of unary relations, then $\#\mathrm{CSP}(\Gamma)$ is polynomial time solvable.*

Proof: By Theorem 10 it suffices to consider the case when Γ contains only unary relations. Then given an instance $\mathcal{P} = (V, \mathcal{C})$, for each variable x we find the intersection of all constraint relations imposed on x; let n_x be the number of elements in this intersection. Then clearly

$$\#\mathcal{P} = \prod_{x \in V} n_x.$$

□

Lemma 5 implies that $\#\mathrm{CSP}(R)$ is polynomial time if R is one of the relations R_1, R_2, R_3, and R_{13}. In the case of $R = R_4$ the problem is equivalent to a problem on a 2-element set, and its polynomial time solvability follows from Theorem 3.

The polynomial time solvability of two more problems $\#\mathrm{CSP}(R_6)$ and $\#\mathrm{CSP}(R_8)$ follows from Theorem 4.

When dealing with a binary relation R we will treat $\#\mathrm{CSP}(R)$ as the $\#R$-Coloring problem. In particular, instances of a such problem are digraphs.

Lemma 6. $\#\mathrm{CSP}(R_5)$ *is polynomial time solvable.*

Proof: We consider $\#\mathrm{CSP}(R_5)$ as the problem of counting homomorphisms of a given digraph $G = (V, E)$ to R_5. As is easily seen, if a homomorphism from G to R_5 exists then V can be partitioned into $V_0 \cup V_1 \cup V_2$ where the vertices from $V_0 \cup V_1$ are not isolated, the vertices from V_2 are isolated, every vertex from V_0 has indegree 0, and every vertex from V_1 has outdegree 0. Any homomorphism of G to R_5 maps the set V_0 to $\{0\}$, the set V_1 to $\{1, 2\}$ arbitrarily, and the set V_2 to $\{0, 1, 2\}$ arbitrarily. Thus, the number of such homomorphisms is $2^{|V_1|}3^{|V_2|}$. □

Observe that Lemma 6 can also be deduced from Theorem 5, however, the condition of polynomial time solvability given in [12] is fairly sophisticated.

Lemma 7. $\#\mathrm{CSP}(R_{14})$ *is polynomial time solvable.*

Proof: The solution algorithm is similar to the one from the previous lemma. If an instance $\mathcal{P}$ of $\#\mathrm{CSP}(R_{14})$ with variable set V has a solution, then V can be partitioned into $V_1 \cup V_2$ so that in any solution of the problem the variables from V_1 take value from $\{0, 1\}$ while the variables from V_2 take value 2. The variables from V_2 are those that appear in the first position of constraint scopes, and do not appear in any other position. Now let $R' = \mathrm{pr}_{2,3} R$. Consider the problem $\mathcal{P}'$ constructed as follows: The set of variables

of $\mathcal{P}'$ is V_1, and every constraint $\langle (u,v,w), R\rangle$ of $\mathcal{P}$ is replaced with $\langle (v,w), R'\rangle$. As is easily seen, $\mathcal{P}$ and $\mathcal{P}'$ have the same number of solutions, but $\mathcal{P}'$ is an instance of #CSP(R'), where $R' = \{(0,1),(1,0)\}$, which is solvable in polynomial time by Theorem 3. □

Lemma 8. #CSP(R_{15}) *is polynomial time solvable.*

Proof: Observe that the relation R_{15} has a polymorphism that is the affine operation $x-y+z$ of $GF(3)$. As noted in Example 8, this means that #CSP(R_{15}) is equivalent to counting solutions of systems of linear equations over the 3-element field, which can be done in polynomial time. □

4.3 #P-complete cases

So far we have identified only one maximal partial clone that gives rise to a #P-complete counting problem.

Lemma 9. *$\#CSP(R_{16})$ is #P-complete.*

Proof: We use Lemma 4 with $S = \{2,3\}$ to show that the #3-Coloring problem, equivalent to #CSP($\neq_3$), where $\neq_3$ is the disequality relation on a 3-element set, or, alternatively, a complete graph with 3 vertices, AP-reduces to $\#CSP(R_{16})$. The problem #CSP($\neq_3$) is #P-complete by [26]. □

4.4 #BIS-complete cases

In this subsection we consider maximal partial clones that give rise to counting CSPs AP-interreducible with #BIS.

In many cases it is more convenient to use another problem instead of #BIS. Recall that a *downset* of a partial order P is a subset $T \subseteq P$ such that if $a \in T$ and $b \leq a$ then $b \in T$. The #Downset problem is, given a partial order, find the number of downsets in it. It is shown in [10] that #BIS and #Downset are AP-interreducible. The #Downset problem can also be viewed as the $\#R_{DS}$-Coloring problem, where R_{DS} is the digraph shown in Fig. 2.

Lemma 10. *$\#CSP(R_7) \equiv_{AP}$ #Downset.*

Proof: We need to present reductions between #Downset and #CSP(R_7) in both directions. Observe that R_7 consists of two connected components, one of which is isomorphic to R_{DS}, and the other one is an isolated vertex with a loop. Therefore, for any connected digraph G the number of homomorphisms into R_7 is one more than that into R_{DS}. Now the result follows from Lemmas 3 and 2. □

Recall that a *strongly connected component* of a directed graph G is a maximal set of vertices S such that for any $v, w \in S$ there is a directed path from v to w and from w to v. Let $\tilde{G}$ denote the (acyclic) digraph obtained from G by contracting each strongly connected component into a single vertex. Clearly, if a digrah H is acyclic (possibly with loops) then any homomorphism from G to H maps an entire strongly connected component into a single vertex of H.

Lemma 11. *If H is an acyclic digraph then for any G the number of homomorphisms from G to H equals that from $\tilde{G}$ to H.*

Lemma 12. *$\#CSP(R_9) \equiv_{AP}$#Downset.*

Proof: First we show that #Downset AP-reduces to #CSP(R_9). Let $G = (V, C)$ be an instance of #Downset, by Lemma 11 it can be assumed acyclic; let also N be the number of homomorphisms from G to R_{DS}. Add a new vertex u to G and, for each $v \in V$ the edge (v, u). Denote the resulting graph by G'. We claim that the number of homomorphisms from G' to R_9 equals $2N + 1$. Indeed, for every homomorphism $\varphi : G' \to R_9$ the image $\varphi(V \cup \{u\}) \subseteq \{0, 1\}$ or $\varphi(V \cup \{u\}) \subseteq \{0, 2\}$, depending whether $\varphi(u) = 1$ or $\varphi(u) = 2$. In the case $\varphi(u) = 0$ we have $\varphi(V \cup \{u\}) \subseteq \{0\}$. In both cases $\varphi(u) = 1$ and $\varphi(u) = 2$ homomorphisms from G' to R_9 one-to-one correspond to homomorphisms from G to R_{DS}. We complete the proof by applying Lemma 3.

Next we show that #CSP(R_9) $\leq_{AP}$ #Downset. Let $G = (V, E)$ be a digraph, an instance of #CSP(R_9). By Lemma 11 we can assume G is acyclic. Let $G' = (V', E')$ be a disjoint union of $G_1 = (V_1, E_2)$ and $G_2 = (V_2, E_2)$, where G_1 is an isomorphic copy of G with vertex v_1 corresponding to each $v \in V$, and G_2 is an isomorphic copy of G^R obtained by reversing the direction of all edges in G with vertex v_2 corresponding to each$v \in V$. We also add edge (v_1, v_2) for each vertex $v \in V$.

We show that the number of homomorphisms of G to R_9 equals the number of homomorphisms from G' to R_{DS}. Observe, first, that for every homomorphism $\varphi : G' \to R_{DS}$ and any $v \in V$ we have $(\varphi(v), \varphi(v')) \in \{(0, 0), (0, 1), (1, 1)\}$. Thus for every homomorphism $\varphi : G' \to R_{DS}$ a mapping $\varphi' : G \to R_9$ can be defined by $\varphi'(v) = 0, 1$ or 2 if $(\varphi(v), \varphi(v')) = (0, 1)$, $(0, 0)$, or $(1, 1)$, respectively. It is straightforward that φ is a homomorphism. Indeed, if $(v, w) \in E$ and, say, $\varphi(v) = 1$, then $(w', v'), (w, w') \in E'$ and $\varphi'(v) = 0, \varphi(v') = 0$, which implies $\varphi(w') = 0$ and $\varphi(w) = 0$. Other cases are similar.

Conversely, if φ' is a homomorphism of G to R_9 then the mapping $\varphi : G' \to R_{DS}$ given by $(\varphi(v), \varphi(v')) = (0, 1)$, $(0, 0)$, or $(1, 1)$, whenever $\varphi'(v) = 0, 1$ or 2, respectively, is a homomorphism of G' to R_{DS}. Thus the solutions of the two instances are in one-to-one correspondence that implies the result. □

Lemma 13. *$\#CSP(R_{10}) \equiv_{AP}$#Downset.*

Proof: To show that #Downset$\leq_{AP}$ #CSP(R_{10}), we apply Lemma 1 to the relation R_{10}, the second coordinate position, and the set $S = \{0, 1\}$. As is easily seen, $R_{DS} = R_{10}(x, y) \wedge S(x) \wedge S(y)$ that implies the result.

To prove #CSP(R_{10}) $\leq_{AP}$#Downset we use a construction similar to that in the previous proof. For a given graph $G = (V, E)$ we construct a graph $G' = (V', E)$ which is a union of two disjoint copies $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ of G, the two copies of a vertex $v \in V$ are denoted v_1, v_2. Furthermore, for every $v \in V$ we introduce an edge (v_1, v_2), and for each edge $(v, w) \in E$ we introduce edges (v_1, w_2) and (w_1, v_2) in G'.

Then, as in the proof of Lemma 12, there is a one-to-one correspondence between homomorphisms of G to R_{10} and those from G' to R_{DS}, defined by the following rule: If $\varphi : G' \to R_{DS}$ is a homomorphism, then $\varphi' : G \to R_{10}$ is such that $\varphi'(v) = 0, 1$, or 2, for $v \in V$, whenever $(\varphi(v_1), \varphi(v_2)) = (0, 1)$, $(1, 1)$, or $(0, 0)$, respectively. The result follows. □

Lemma 14. *$\#CSP(R_{11}) \equiv_{AP}$#Downset.*

Proof: To show that #Downset$\leq_{AP}$ #CSP(R_{11}), we apply Lemma 1 to the relation R_{11}, the first coordinate position, and the set $S = \{0, 1\}$. As is easily seen, $R_{DS} = R_{11}(x, y) \wedge S(x) \wedge S(y)$ proving the reduction.

Again, to prove #CSP(R_{11}) $\leq_{AP}$#Downset we use a construction similar to that in the proof of Lemma 12. For a given graph $G = (V, E)$ we construct a graph $G' = (V', E')$ which is a union of two disjoint copies $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ of G, the two copies of a vertex $v \in V$ are denoted v_1, v_2. Furthermore, for every $v \in V$ we introduce an edge (v_1, v_2).

Then, as in the proof of Lemma 12, there is a one-to-one correspondence between homomorphisms of G to R_{11} and those from G' to R_{DS}, defined by the following rule: If $\varphi : G' \to R_{DS}$ is a homomrphism, then $\varphi' : G \to R_{11}$ is such that $\varphi'(v) = 0, 1$, or 2, for $v \in V$, whenever $(\varphi(v_1), \varphi(v_2)) = (0, 1)$, $(1, 1)$, or $(0, 0)$,

respectively. The result follows. □

The problem $\#CSP(R_{12})$ was considered in [10], where it is dubbed as the #2-Particle-WR-Configs, and shown to be interreducible with #BIS.

Finally, we identify two #BIS-hard problems, meaning #BIS is AP-reducible to those problems. Their precise approximation complexity remains open.

Lemma 15. *The #BIS problem is AP-reducible to $\#CSP(R_{17})$ and $\#CSP(R_{18})$*

Proof: In both cases we apply Lemma 4. For $\#CSP(R_{17})$ we use this lemma setting $S = \{1, 2\}$. The resulting relation is then R_7, so $\#CSP(R_7) \leq_{AP} \#CSP(R_{17})$, and by Lemma 10, $\#CSP(R_7)$ is AP-interreducible with #BIS.

For $\#CSP(R_{18})$ we again use Lemma 4 setting $S = \{1, 2\}$. The resulting relation is then R_9, so $\#CSP(R_9) \leq_{AP} \#CSP(R_{18})$, and by Lemma 12, $\#CSP(R_9)$ is AP-interreducible with #BIS. □

References

1. A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *J. ACM*, 53(1):66–120, 2006.
2. A. Bulatov. The complexity of the counting constraint satisfaction problem. In *ICALP (1)*, pages 646–661, 2008.
3. A. Bulatov and V. Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. *Inf. and Comp.*, 205(5):651–678, 2007.
4. A. Bulatov and M. Grohe. The complexity of partition functions. *Theor. Comput. Sci.*, 348(2-3):148–186, 2005.
5. A. Bulatov, P. Jeavons, and A. Krokhin. Functions of multiple-valued logic and the complexity of constraint satisfaction: A short survey. In *ISMVL*, pages 343–351, 2003.
6. A. Bulatov, P. Jeavons, and A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM J. Comput.*, 34(3):720–742, 2005.
7. N. Creignou and M. Hermann. Complexity of generalized satisfiability counting problems. *Information and Computation*, 125(1):1–12, 1996.
8. N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, 2001.
9. K. Denecke and S.L. Wismath. *Universal algebra and applications in Theoretical Computer Science*. Chapman and Hall/CRC Press, 2002.
10. M. Dyer, L.A. Goldberg, C. Greenhill, and M. Jerrum. On the relative complexity of approximate counting problems. In *APPROX*, volume 1913 of *LNCS*, pages 108–119. Springer-Verlag, 2000.
11. M. Dyer, L.A. Goldberg, and M. Jerrum. An approximation trichotomy for Boolean #CSP. *CoRR*, abs/0710.4272, 2007.
12. M. Dyer, L. Goldberg, and M. Paterson. On counting homomorphisms to directed acyclic graphs. In *ICALP*, pages 38–49, 2006.
13. M. Dyer and C. Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17:260–289, 2000.
14. A. Ehrenfeucht and M. Karpinski. The computational complexity of (xor, and)-counting problems. Technical Report 8543-CS, 1990. Available at `http://citeseer.ist.psu.edu/ehrenfeucht90computational.html`
15. I. Fleischer and I.G. Rosenberg. The Galois connection between partial operations and relations. *Pacific J. Math.*, 79:93–97, 1978.
16. L. Haddad and I.G. Rosenberg. Completeness theory for finite partial algebras. *Algebra Universalis*, 29:378–401, 1992.
17. P. Hell and J. Nešetřil. On the complexity of H-coloring. *J. of Comb. Theor., Ser.B*, 48:92–110, 1990.
18. M. Jerrum. A very simple algorithm for estimating the number of k-colorings of a low-degree graph. *Random Struct. Algorithms*, 7(2):157–166, 1995.
19. P.W. Kasteleyn. Graph theory and crystal physics. In *Graph Theory and Statistical Physics* (F. Harari, ed.), Academic Press, 43–110, 1967.
20. B. Larose, C. Loten, and C. Tardif. A characterisation of first-order constraint satisfaction problems. In *LICS*, pages 201–210, 2006.

21. L.A. Levin. Universal enumeration problems. *Problems on Information Transmission*, 9:265–266, 1973.
22. R. Lidl and H. Niederreiter. *Finite fields.* volume 20 of Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge, 2nd edition, 1997.
23. M. Luby and B. Velickovic. On deterministic approximation of DNF. *Algorithmica*, 16(4/5):415–433, 1996.
24. C.H. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.
25. R. Pöschel and L.A. Kalužnin. *Funktionen- und Relationenalgebren.* DVW, Berlin, 1979.
26. L. Valiant. The complexity of computing the permanent. *Theoretical Computing Science*, 8:189–201, 1979.
27. L. Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.