On the Computational Complexity of Monotone Constraint Satisfaction Problems

Miki Hermann and Florian Richoux

LIX (CNRS, UMR 7161), École Polytechnique, 91128 Palaiseau, France
{hermann | richoux}@lix.polytechnique.fr

Abstract. Constraint Satisfaction Problems (CSP) constitute a convenient way to capture many combinatorial problems. The general CSP is known to be NP-complete, but its complexity depends on a parameter, usually a set of relations, upon which they are constructed. Following the parameter, there exist tractable and intractable instances of CSPs. In this paper we show a dichotomy theorem for every finite domain of CSP including also disjunctions. This dichotomy condition is based on a simple condition, allowing us to classify monotone CSPs as tractable or NP-complete. We also prove that the meta-problem, verifying the tractability condition for monotone constraint satisfaction problems, is fixed-parameter tractable. Moreover, we present a polynomial-time algorithm to answer this question for monotone CSPs over ternary domains.

1 Introduction

Constraint Satisfaction Problems (CSP) constitute a common formalism to describe many algorithmic problems originating from combinatorics and graph theory, artificial intelligence, computational molecular biology, etc. These problems require a quantitative analysis studying their computational complexity. The goal to study the complexity of constraint satisfaction problems is the recognition of conditions allowing us to distinguish between tractable and intractable instances of the considered problem, as well as the understanding of the complexity classes to which these instances belong. The study of computational complexity of constraints satisfaction problems was started by Schaefer in his landmark paper [9], where he completely characterized the complexity of Boolean CSP, distinguishing between polynomial and NP-complete instances. Feder and Vardi [4] extended this study to constraint satisfaction problems over finite domains, for which they conjectured the existence of a Dichotomy Theorem. So far, this claim was proved only for the ternary domain by Bulatov [1], exhibiting an involved Dichotomy Theorem, whereas the claim remains open for higher cardinality domains.

The main difficulty to resolve this conjecture resides in the nonexistence of a finite or countable number of function sets, called clones, under which the relations building the CSPs are closed. The closure under the functions of a given clone provides us with the required characterization of (in)tractability. There is a countably infinite number of clones for the Boolean domain and the cases discriminating between tractable and NP-complete instances appear in a finite part of the lattice. The characterization for each clone can always be expressed by a finite set of function, called a basis. These facts were

largely exploited by Schaefer in [9]. However, the corresponding lattice for domains of cardinality 3 and more is uncountable, as it was proved by Yanov and Muchnik [10].

One possibility to circumvent this difficulty is to introduce more structure into the constraint satisfaction problems, hoping to find only a finite number of cases to be analyzed. To introduce more structure, we choose to extend the constraint satisfaction problems with disjunction. Usual CSPs consider a set of constraints logically interpreted as a conjunction. Introducing a disjunction connective among constraints might seem to destroy this correspondence between the logic and set-theoretic approaches. If we consider each set of relations including the disjunction, we can easily recover the aforementioned correspondence. Our approach explores that particular part of constraint satisfaction problems that always include the disjunction relation. Monotone CSPs have been also considered from a different viewpoint by Cohen, Jeavons, Jonsson and Koubarakis in [2]. A moments reflection might consider monotone constraint satisfaction problems as too restrictive. However, this is incorrect since they represent the CSPs over weak Krasner algebras, a large and well-studied structure in universal algebra [6, 7]. The advantage of weak Krasner algebras is that they are closed only under endomorphisms. Since there is only a finite number of endomorphisms on a given finite domain, we are ensured to obtain a finite complexity characterization.

In this paper we study the complexity of monotone constraint satisfaction problems, also considered to be the CSPs over weak Krasner algebras. We derive for them a complete characterization of complexity by means of a Dichotomy Theorem for each finite domain. Once the tractability conditions derived, we study also the complexity of the meta-problem, i.e., the complexity of the decision problem whether a given monotone CSP satisfies the tractability condition. This analysis is first performed for the general case of any domain cardinality. Since the ternary domain presents a particular behavior, we perform a new complexity analysis for its meta-problem, deriving a sharper result than for the general case.

2 Basic Notions

All along this paper, a function f is an unary function $f\colon D \to D$ with D being a fixed size domain. The *range* of f, denoted by $\operatorname{ran} f$, is the set $\{f(x) \mid x \in D\}$. We write $f(A)$ for a subset $A \subseteq D$ to denote the set $\{f(a) \mid a \in A\}$.

The study of constraint satisfaction problems often uses the notion of *relations*, *clones* and *co-clones*. A n-ary **relation** R on domain D is a subset of D^n. We denote by S a set of relations $R \subseteq D^k$ where k is the arity of R. A **clone**, or a **functional clone**, is a set of functions containing the identity and closed under composition. The smallest clone containing the functions F is denoted by $[F]$. In our scope, a **co-clone**, or a **relational clone**, is a set of relations containing the equality $eq = \{(d, d) \mid d \in D\}$ and closed under conjunction, *disjunction*, variable identification and existential quantification. The smallest co-clone containing the relations S is denoted by $\langle S \rangle$. Note that we authorize the closure under disjunction in addition to the usual definition of closure.

Universal algebra often uses the concept of *kernel*. We define this notion with the help of equivalence classes. Let $f\colon D \to D$ be a function with range $\operatorname{ran} f$. A *d-equivalence class*, for $d \in \operatorname{ran} f$, is the set of elements $[d]_f = \{x \in D \mid f(x) = d\}$.

The **kernel** $\ker f$ of a function f is the set of all equivalence classes with cardinality strictly greater than 1 for all $d \in \operatorname{ran} f$, *i.e.* $\ker f = \{[d]_f \mid d \in \operatorname{ran} f \text{ and } |[d]_f| \geq 2\}$.

We need the notion of function depth to describe the computational complexity of a clone $[F]$. The *depth* of a function f *with respect to* a set F, denoted by $\mathrm{Depth}(F, f)$, is the length of the shortest composition of functions $f_i \in F$ required to obtain f. The *depth* of a function f, denoted by $\mathrm{Depth}(f)$, is defined by the identity $\mathrm{Depth}(f) = \max(\mathrm{Depth}(F, f))$ for all sets F such that $f \in [F]$. Thus, if we want to obtain a function f from a set F, it is useless to make compositions of functions longer than $\mathrm{Depth}(f)$. If all combinations with length less or equal than $\mathrm{Depth}(f)$ do not allow us to get a function f, it means that f does not belong to the clone generated by F.

3 Monotone Constraint Satisfaction Problems

Constraint satisfaction problems are usually presented as conjunctions of constraints. The parametric problem $\textsc{csp}(S)$, with S being a set of relations, is a constraint satisfaction problem where constraints are built upon relations from S. Constraint satisfaction problems are known to be NP-complete in general, but the complexity of parametric problems $\textsc{csp}(S)$ depends on S, ranging from polynomial to NP-complete. Schaefer [9] gave a dichotomy theorem on Boolean constraints. If S verifies one of the six conditions presented by Schaefer, then $\textsc{csp}(S)$ is polynomial. Otherwise it is NP-complete.

It is difficult to obtain such a dichotomy for finite domains of higher cardinality. The only exception is a result from Bulatov [1] showing a dichotomy for ternary domains. A dichotomy theorem for other finite domains is still an open question. We will focus our interest on the complexity of parametric problems $\textsc{mcsp}(S)$ where MCSP is a generalization of CSP defined as follows: atomic constraints play the role of literals and constraints are built by means of conjunctions and disjunctions. In this paper, we present a dichotomy with simple conditions for the algorithmic problem $\textsc{mcsp}(S)$ with S being a set of relations over a finite domain D.

Let $R \subseteq D^k$ be a relation on D and V be a set of variables. A *literal* is a predicate $R(x_1, \ldots, x_k)$ formed from the relation R and variables x_1, …, x_k in V, where $\operatorname{ar}(R) = k$. A *formula* is defined inductively in the following way: (1) TRUE and FALSE, respectively denoted by $\top$ and $\bot$, are formulas; (2) a literal l is a formula; (3) if φ_1 and φ_2 are formulas, then $\varphi_1 \vee \varphi_2$ and $\varphi_1 \wedge \varphi_2$ are formulas; (4) finally if φ is a formula containing a free variable x then $\exists x\, \varphi$ is a formula. We do not consider negation in our formulas because it brings us to an obvious generalization of the problem $\textsc{csp}(S)$, which is NP-complete over every finite domain of cardinality greater than or equals to three, for each S. An *interpretation* $I \colon V \to D$ satisfies the literal $R(x_1, \ldots, x_k)$, denoted by $I \vDash R(x_1, \ldots, x_k)$, if $(I(x_1), \ldots, I(x_k)) \in R$. It is extended to formulas in the following way: (a) $I \vDash R_1(\boldsymbol{x}) \vee R_2(\boldsymbol{x})$ if $I \vDash R_1(\boldsymbol{x})$ or $I \vDash R_2(\boldsymbol{x})$, (b) $I \vDash R_1(\boldsymbol{x}) \wedge R_2(\boldsymbol{x})$ if $I \vDash R_1(\boldsymbol{x})$ and $I \vDash R_2(\boldsymbol{x})$. We note that for all formulas φ, we have $\top \vDash \varphi$ and $\bot \nvDash \varphi$.

We define the **monotone constraint satisfaction problem** as follows.

Problem: $\textsc{mcsp}(S)$
Input: A formula $\varphi(\boldsymbol{x})$.
Question: Is the formula $\varphi(\boldsymbol{x})$ satisfiable?

Our study of monotone constraint satisfaction problems is made easier by the existence of a Galois connection among clones and co-clones. Before defining what a Galois connection is, we need the following notions. Let R be a relation of arity $\mathrm{ar}(R) = k$ and f be a function of arity $\mathrm{ar}(f) = m$. We say that f is a polymorphism of R if the membership condition

$$(f(r_1[1], \ldots, r_m[1]), \ldots, f(r_1[k], \ldots, r_m[k])) \in R \tag{1}$$

is verified, with $r_i = (r_i[1], \ldots, r_i[k]) \in R$ for all $i \in \{1, \ldots, m\}$. The set of polymorphisms of a relation R is denoted by $\mathrm{Pol}\, R$. The set of polymorphisms of a relations set S is also denoted by $\mathrm{Pol}\, S$, defined by $\mathrm{Pol}\, S = \bigcap_{R \in S} \mathrm{Pol}\, R$. In a similar way, a relation R is an invariant of a function f if the condition (1) holds; we also say that R is closed under f. We denote by $\mathrm{Inv}\, f$ the invariants of f and $\mathrm{Inv}\, F$ the invariants of a set of functions F.

Pöschel proved in [7] that applying disjunction on constraints implies that relations satisfying an instance of a MCSP problem are invariant only under unary functions. Polymorphisms of these relations are thus endomorphisms and we denote them by End instead of Pol. Let us for a moment denote respectively by A and B sets of relations and functions. Pöschel [7] specifies that the mappings $\mathrm{End}\colon A \to B$ and $\mathrm{Inv}\colon B \to A$ present a Galois connection between the ordered structures $(A, \subseteq)$ and $(B, \subseteq)$. Moreover, for all sets of relations S and functions F, the identities $\mathrm{Inv}\,\mathrm{End}\, S = \langle S \rangle$ and $\mathrm{End}\,\mathrm{Inv}\, F = [F]$ hold. Pöschel points out that the sets $\langle S \rangle$ and $[F]$ are respectively a **weak Krasner algebra** and an **endomorphisms monoid**. In this scope we can consider the monotone constraint satisfaction problems as CSP defined upon weak Krasner algebras. The existence of the aforementioned Galois connection allows us to easily decide the complexity of monotone constraint satisfaction problems. The complexity analysis is based on the following result.

Proposition 1. *Let S_1 and S_2 be two sets of relations over the domain D, such that $eq \in S_1$. The inclusion $\mathrm{End}\, S_1 \subseteq \mathrm{End}\, S_2$ implies $\textsc{mcsp}(S_2) \leq_m^{\mathrm{P}} \textsc{mcsp}(S_1)$.*

Proof. From $\mathrm{End}\, S_1 \subseteq \mathrm{End}\, S_2$ follows $\langle S_1 \rangle = \mathrm{Inv}\,\mathrm{End}\, S_1 \supseteq \mathrm{Inv}\,\mathrm{End}\, S_2 = \langle S_2 \rangle$. Since every co-clone (equivalent to a weak Krasner algebra) contains the equality relation eq and is closed under conjunction, disjunction, and existential quantification, we immediately derive the reduction $\textsc{mcsp}(S_2) \leq_m^{\mathrm{P}} \textsc{mcsp}(S_2 \cup \{eq\}) \leq_m^{\mathrm{P}} \textsc{mcsp}(S_1 \cup \{eq\})$. Since $eq \in S_1$, we have $\textsc{mcsp}(S_1 \cup \{eq\}) = \textsc{mcsp}(S_1)$ and the result follows. □

According to Proposition 1, a complexity result proved for a set of relations S immediately extends to all relations in the co-clone $\langle S \rangle$, provided that S contains the equality relation eq. Therefore from now on we always assume that the **equality relation eq is included in every set S**. Moreover, the presence of the quaternary relation $J = \{(x, y, z, w) \in D^4 \mid (x = y) \vee (z = w)\}$ in a classical constraint satisfaction problem reduces it to a monotone constraint satisfaction problem.

Proposition 2. *Let S be a set of relations on the domain D and consider the relation $J = \{(x, y, z, w) \in D^4 \mid (x = y) \vee (z = w)\}$. If $J \in S$ then $\mathrm{Pol}\, S = \mathrm{End}\, S$.*

Proof. Suppose that there exists a binary function $g \in \operatorname{Pol} S$ depending on both arguments, i.e., there exist values $a_0, a_1, a_2, b_0, b_1, b_2 \in D$ such that $a_0 \neq a_1$, $b_1 \neq b_2$, $g(a_0, a_2) \neq g(a_1, a_2)$ and $g(b_0, b_1) \neq g(b_0, b_2)$. Clearly, the vectors $j_1 = (a_0, a_1, b_0, b_0)$ and $j_2 = (a_2, a_2, b_1, b_2)$ belong to J, but the vector $(g(a_0, a_2),\ g(a_1, a_2),\ g(b_0, b_1),\ g(b_0, b_2))$ is absent from J. Therefore the function g cannot be a polymorphism of a set of relations containing J. From any function f of arity $\operatorname{ar}(f) > 2$ we can always produce a binary function by variable identification. □

Since the number of endomorphisms over a finite domain is always finite, we are ensured to obtain a finite complexity characterization for MCSP.

4 Complexity of MCSP(S)

We will exhibit a dichotomy of MCSP(S) complexity characterized by a simple criterion. It is easier to prove this dichotomy for MCSP($f(S)$) where f is a permutation keeping invariant the set S. We need the following proposition adapted from Jeavons [5] showing that the problems MCSP(S) and MCSP($f(S)$) are logspace-equivalent.

Proposition 3 (Jeavons [5]). *Let S be a set of relations on D and f be an unary function on D. Let $f(S) = \{f(R) \mid R \in S\}$. If each relation $R \in S$ is closed under f then* MCSP($f(S)$) *is logspace-equivalent to* MCSP(S).

Proof. Suppose that each relation $R \in S$ is closed under f. Let $\varphi(\boldsymbol{x})$ be an instance of MCSP($f(S)$). We have a formula $\varphi(\boldsymbol{x})$ where literals $R(x_1, \ldots, x_k)$ are constructed from relations $R \in f(S)$. According to the hypothesis, all relations $R \in S$ are closed under f, *i.e.* the inclusion $f(R) \subseteq R$ holds for all $R \in S$. We deduce that $\varphi(\boldsymbol{x})$ is also an instance of MCSP(S).

Take a formula $\varphi(\boldsymbol{x})$ being an instance of MCSP(S). It can be transformed by a logspace reduction to an instance $\varphi'(\boldsymbol{x})$ of MCSP($f(S)$) by replacing each literal constructed from a relation R by a literal constructed from $f(R)$. Moreover, because R is closed under f, we have $f(R) \subseteq R$ and we derive that all solutions of $\varphi'(\boldsymbol{x})$ are also solutions of $\varphi(\boldsymbol{x})$. Conversely, if h is a solution of $\varphi(\boldsymbol{x})$ then $f(h)$ is a solution of $\varphi'(\boldsymbol{x})$. Thus we have a logspace-equivalence among MCSP($f(S)$) and MCSP(S). □

We will study monotone constraint satisfaction problems MCSP(S) through sets of functions F satisfying $\operatorname{Inv} F = S$. The following propositions prove that MCSP(S) is polynomial if $[F]$ contains a constant function, and it is NP-complete otherwise.

Proposition 4. *Let F be a set of unary functions such that $\operatorname{End} S = [F]$ for a set of relations S. If $[F]$ contains a constant function then* MCSP(S) *is polynomial.*

Proof. If the endomorphisms of S contain a constant function $f_d(x) = d$ for all $x \in D$, then for the set of relations $\langle S \rangle$ invariant under $[F]$, each relation $R \in \langle S \rangle$ contains a d-vector, *i.e.* a mapping of each variable to the value d. Therefore each instance of MCSP(S) is satisfiable by a d-vector. □

Lemma 5. *Let $[F]$ contain no constant functions. Let $f \in [F]$ be a function with the smallest cardinality of $\operatorname{ran} f$. Then $\operatorname{End} f(S)$ contains only permutations.*

Proof. Suppose there is a function $g \in \operatorname{End} f(S)$ not being a permutation. Necessarily g is not injective, *i.e.* the inclusion $\operatorname{ran} g \subsetneq \operatorname{ran} f$ holds. This is a contradiction with the fact that the cardinality of $\operatorname{ran} f$ is the smallest among all functions in $[F]$. □

Proposition 6. *Let F be a set of unary functions such that the clone $[F]$, equivalent to $\operatorname{End} S$ for a set of relations S, does not contain constant functions. Then* MCSP(S) *is* NP*-complete.*

Proof. We adapt the proof of Proposition 5.6 from [5]. Let $[F]$ be without constant functions. Let $f \in [F]$ be a function with minimal cardinality of its range $\operatorname{ran} f$. By Lemma 5, we know that $\operatorname{End} f(S)$ contains only permutations. Since $[F]$ does not contain constant functions, we also know that cardinality of $\operatorname{ran} f$ satisfies the condition $|\operatorname{ran} f| \geq 2$. We have to separate two cases.

If $|\operatorname{ran} f| = 2$, then we assume without loss of generality that $\operatorname{ran} f = \{0, 1\}$. The set $\operatorname{End} f(S)$ contains only permutations on $\{0, 1\}$. Let R_{NAE} be the relation $\{0, 1\}^3 \setminus \{000, 111\}$. It is clear that relation R_{NAE} is closed under every permutations on $\{0, 1\}$. Hence we have the inclusion $\operatorname{End} f(S) \subseteq \operatorname{End} R_{NAE}$. The relation R_{NAE} gives rise to the NOT-ALL-EQUAL-3SAT problem, known to be NP-complete. We conclude that MCSP$(f(S))$ is NP-complete.

Let now $|\operatorname{ran} f| \geq 3$. The set of relations $\operatorname{Inv} \operatorname{End}(f(S))$ is closed under permutations on $\operatorname{ran} f$. In particular, it contains the set of relations $Q \subseteq D^2$ where $Q = \{a_1, a_2, \ldots, a_k\}^2 \setminus \{(a_1, a_1), (a_2, a_2), \ldots, (a_k, a_k)\}$, such that the elements a_1, $\ldots$, a_k present in the relation satisfy the identity $|\{a_1, a_2, \ldots, a_k\}| = |\operatorname{ran} f|$. Relations in Q are the valid valuations for all instances of the $|\operatorname{ran} f|$-coloring problem. This problem is known to be NP-complete since $|\operatorname{ran} f| \geq 3$. We conclude that MCSP$(f(S))$ is also NP-complete in this case.

We have seen in Proposition 3 that the problem MCSP$(f(S))$ is logspace-equivalent to MCSP(S). We conclude that MCSP(S) is NP-complete. □

From Propositions 4 and 6 we derive the main theorem of this section.

Theorem 7. *The monotone constraint satisfaction problem* MCSP(S) *is polynomial if the set $\operatorname{End} S$ contains a constant function. Otherwise, it is* NP*-complete.*

We have presented a very simple dichotomy condition for the problem MCSP(S) on a finite domain D. To decide this condition, it is sufficient to check if the endomorphisms set $\operatorname{End} S$ contains a constant function. The endomorphism set $\operatorname{End} S$ is always equal to the clone $[F]$ for some set of functions F.

We can also consider the monotone CSPs starting from a set of functions F. Given a set of unary functions, we consider the problem MCSP$(\operatorname{Inv} F)$. An interesting question from a complexity point of view is to ask now, given a set of unary functions F, if the clone $[F]$ contains a constant function. This meta-problem is treated in the next section.

5 Complexity of Clones

To determine if a composition of functions from the set F constructs a constant function, one has to compute at least a part of the clone $[F]$. Salomaa [8] calls *class membership problem* the question to decide, given a set of functions F and a function class C,

whether the clone $[F]$ contains a function belonging to C. Salomaa proved this problem to be NP-hard. We will show that it is even NP-complete if C is the class of constant functions. However, it is interesting to note that we are working on constraint satisfaction problem where the domain size is fixed. We will see that this allows us to prove the *class membership problem* to be fixed-parameter tractable (FPT).

We need first a result by Salomaa [8] allowing us to bind the combination depth of functions belonging to a set F to obtain a constant function. This bound will be useful for the NP-membership proof in the sequel.

Proposition 8 (Salomaa [8]). *Let F be a set of functions without constant functions. Let D be the functions domain and n the domain size. Each constant function f_c on D verifies the condition* $\mathrm{Depth}(f_c) \leq n^3/2 - 3n^2/2 + 2n$.

Following Salomaa [8], it is not necessary to go beyond this polynomial bound in order to find a constant function in $[F]$. Once arrived at this bound without finding a constant function, we know that there are no constant functions in $[F]$.

Proposition 9. *The class membership problem is* NP*-complete if C is the class of constant functions.*

Proof. We know from Salomaa [8] that this problem is NP-hard. We just have to show that it is also in NP. By Proposition 8, we know that the depth of every constant function is limited by $n^3/2 - 3n^2/2 + 2n$, with n being the domain size. A certificate f_c cannot be longer than this bound. We have to check two conditions. First, that each function composing the sequence f_c is in F, and second that f_c is a constant function. The first step is obviously in $O(n^3.|F|)$, and the second step asking to compute f_c is in $O(n^4)$ because we have to compute n values n^3 times. Notice that $|F|$ is the number k of functions in F times the size of a function, which is n. Hence, the certificate f_c has a size depending of n, and it can be decided in $O(k.n^4)$ whether f_c is a constant function. So the class membership problem is in NP if C is the class of constant function, and thus this problem is NP-complete. □

We now focus on the complexity of the problem to determine if a constant function can be obtained from a function set F, *i.e.* the *class membership problem* where C is the class of constant functions. If we assume that D has a fixed size, we can consider the following *parametric class membership problem* version:

Problem: CLASS MEMBERSHIP PROBLEM
Input: A set of functions F.
Parameter: The domain size n.
Question: Does the clone $[F]$ contain a constant function?

We will show that the complexity of this problem is *fixed-parameter tractable*. We begin to introduce this complexity class.

Definition 10 (Downey & Fellows [3]). *A parametric problem P is* fixed-parameter tractable*, or* FPT*, if there exists an algorithm taking inputs (I, k), where I is the principal part of input and k the parameter, and deciding if the membership $(I, k) \in P$ holds in time $f(k).|I|^c$, with f being an arbitrary function and c a constant.*

Algorithm 1 Class Membership Problem

```
1: Q ← {f_i | f_i ∈ F}
2: S ← ∅
3: while Q ≠ ∅ do
4:     f ← dequeue(Q)
5:     S ← S ∪ {f}
6:     for all f_i ∈ F do
7:         g = f_i ∘ f
8:         if g is a constant function then
9:             return "YES"
10:        end if
11:        if g ∉ S then
12:            enqueue(Q, g)
13:        end if
14:    end for
15: end while
16: return "NO"
```

Theorem 11. *The parametric class membership problem for a set of unary functions F is fixed-parameter tractable and it can be decide in time $O(n^n.|F|)$, where n is the domain size parameter.*

Proof. It is sufficient to exhibit an algorithm deciding the *class membership problem*, taking as parameter the domain size n, and terminating in $O(f(n).|F|^c)$ for some constant c. Thus the proof is relative to complexity of Algorithm 1.

First, we prove that Algorithm 1 is sound and terminating. Notice that Q is the set of functions we have to treat (represented by a queue) and S the set of already produced functions. At the beginning Q is instantiated to F. For each element in Q, the algorithm composes them with each function in F, and puts these combinations into Q if they are new (that is, not in S). The following property is the loop invariant: Q never contains twice the same function. Every possible function originating from a combination of F will be explored by the algorithm and tested whether it is a constant function, showing the algorithm soundness.

Since the number of unary functions on a finite domain of size n cannot be greater that n^n, the size of Q cannot pass this limit. From Line 11 follows that the queue Q cannot contain twice the same function and Line 4 shows that an element is removed from Q at each pass through the while loop. Hence Algorithm 1 terminates.

Let us analyze the complexity of Algorithm 1. Lines 1 and 2 are just instantiations. Line 3 activates a loop executed while Q is not empty. We have seen that $|Q| \leq n^n$. Lines 4 and 5 are executed in constant time. Line 6 does not depend on the size of F and complexity of Lines 7 to 13 depends on the choice of data structures. If we choose to use a hash table to represent the set S, where the length of collisions lists is proportional to n, these lines are executed in $O(n)$. Clearly Algorithm 1 runs in time $O(n^n.|F|)$ and allows us to conclude that the *class membership problem* is fixed-parameter tractable, where n is the parameter. □

6 Complexity of Clones on Ternary Domains

The meta-problem for MCSP on ternary domains can be treated more efficiently than the general meta-problem. Actually, we do not have to compute $[F]$, even a part of it, to know if there exists a combination of functions in F that leads to a constant function. To know if it is possible to produce such a constant function, it is sufficient to check if the functions in F verify the subsequent conditions. We first note the following fact.

Remark 12. Kernels of functions on a ternary domain are limited to only one equivalence class. This can be easily verified by the pigeonhole principle. Moreover, the size of these kernels can only be equal to 0, 2 or 3. Therefore we identify in the sequel the kernel of a unary function over a ternary domain with its singleton equivalence class.

Lemma 13. *Let F be a set of functions without constants. The clone $[F]$ contains a constant function f_c if and only if there exists two functions $f_a, f_b \in [F]$ such that $f_c = f_a \circ f_b$ and $\operatorname{ran} f_b \subseteq \ker f_a$.*

Proof. The only-if implication is obvious, therefore we focus on the if-implication. Let $f_c \in [F]$ be a constant function. Then f_c must be a composition of two functions f_a and f_b — possibly obtained by composition — since F does not contain any constant function. Without loss of generality, we can assume that f_a and f_b are non-constant functions such that $f_c = f_a \circ f_b$.

Suppose that for every equivalence class $[d]_{f_a} \in \ker f_a$, we have $\operatorname{ran} f_b \nsubseteq [d]_{f_a}$. Let $x, y \in \operatorname{ran} f_b$ such that, for every $[d]_{f_a}$ we have $\{x, y\} \nsubseteq [d]_{f_a}$. Then $f_a(x) \neq f_a(y)$, but since $x, y \in \operatorname{ran} f_b$, there must exist $x', y' \in D$ such that $f_b(x') = x$ and $f_b(y') = y$. Thus $(f_a \circ f_b)(x') \neq (f_a \circ f_b)(y')$, *i.e.* $f_c(x') \neq f_c(y')$. This is a contradiction with the fact that f_c is a constant function. □

From the aforementioned lemma we immediately derive the following corollary.

Corollary 14. *If there exist two functions f_a and f_b such that $\operatorname{ran} f_b = \ker f_a$, then the composition $f_a \circ f_b$ is a constant function.*

In addition to Lemma 13, we show some useful results on the range and kernel set of functions in the sequel.

Lemma 15. *Let $f, g \in F$. The following conditions hold:*
(i) $\operatorname{ran}(f \circ g) \subseteq \operatorname{ran} f$ *and* $\ker g \subseteq \ker(f \circ g)$*;*
(ii) if $\operatorname{ran} g \nsubseteq \ker f$ *and* $|\ker f| = 2$ *then* $\operatorname{ran}(f \circ g) = \operatorname{ran} f$*;*
(iii) if $\operatorname{ran} g \nsubseteq \ker f$ *and* $|\ker g| = 2$ *then* $\ker g = \ker(f \circ g)$.

Proof. Every unary function f is monotone, *i.e.*, if $A \subseteq B$ then $f(A) \subseteq f(B)$ holds for all subsets A, B of D. Since $\operatorname{ran} g \subseteq D$ and f is monotone, we have $f(\operatorname{ran} g) \subseteq f(D)$. Moreover, $f(\operatorname{ran} g)$ is $\operatorname{ran}(f \circ g)$ and $f(D)$ is $\operatorname{ran} f$. Therefore the inclusion $\operatorname{ran}(f \circ g) \subseteq \operatorname{ran} f$ holds. Let $x, y \in \ker g$. We have $g(x) = g(y)$, therefore $(f \circ g)(x) = (f \circ g)(y)$, *i.e.* $x, y \in \ker(f \circ g)$.

Let $\operatorname{ran} g \nsubseteq \ker f$ and $|\ker f| = 2$. We have to show that $\operatorname{ran} f \subseteq \operatorname{ran}(f \circ g)$. Let $y \in \operatorname{ran} f$. Suppose that for all $x \in \operatorname{ran} g$, the inequality $f(x) \neq y$ holds. Let $z \in \operatorname{ran} f$,

with $z \neq y$. So for all $x \in \operatorname{ran} g$, we have $f(x) = z$ since $|\ker f| = 2$, which implies $\operatorname{ran} g \subseteq \ker f$: contradiction with the assumption.

Let $\operatorname{ran} g \nsubseteq \ker f$. We have to show that $\ker(f \circ g) \subseteq \ker g$. Suppose that there exist $x, y \in \ker(f \circ g)$ such that $g(x) \neq g(y)$. Since $|\ker g| = 2$, we have $\{g(x), g(y)\} = \operatorname{ran} g$. However the equality $(f \circ g)(x) = (f \circ g)(y)$ holds, so we have the inclusion $\operatorname{ran} g \subseteq \ker f$, which is a contradiction. Thus, for all $x, y \in \ker(f \circ g)$, we have $x, y \in \ker g$. □

Corollary 16. *Let* $\{f, g\} = F$ *such that* $\operatorname{ran} g \nsubseteq \ker f$, $\operatorname{ran} f \nsubseteq \ker g$ *and* $|\ker f| = |\ker g| = 2$. *If* $\operatorname{ran} f = \operatorname{ran} g$ *then for every function* $h \in [F]$ *we have* $\operatorname{ran} h = \operatorname{ran} f$. *If* $\ker f = \ker g$ *then for every function* $h \in [F]$, *we have* $\ker h = \ker f$.

We need the notions of *circular* and *swap permutation* on a ternary domain to present our main result.

Definition 17. *A* ***circular permutation*** c *on* $D = \{0, 1, 2\}$ *is a permutation satisfying the condition* $c(x) = (x+k) \bmod |D|$ *with* $k \in D$, *for all* $x \in D$. *A* ***swap permutation*** s *on* $D = \{0, 1, 2\}$ *is a permutation satisfying the conditions* $s(x) = y$, $s(y) = x$ *and* $s(z) = z$, *for distinct* $x, y, z \in D$. *The set* $\{x, y\}$ *is called* $\operatorname{swap} s$.

We can now introduce the main result of this section, dividing it into two parts.

Proposition 18. *Let* F *a set of functions. If* F *satisfies one of the following conditions, then there exists a constant function in* $[F]$. *The conditions are:*

(i) there exists a constant function $f \in F$;
(ii) there exist $f, g \in F$ *(not necessarily distinct) such that* $\operatorname{ran} f = \ker g$;
(iii) there exist $f, c \in F$ *such that* $|\ker f| = 2$ *and* c *is a circular permutation;*
(iv) there exist $f, s \in F$ *such that* $|\ker f| = 2$ *and* s *is a swap permutation where* $\operatorname{swap} s \neq \operatorname{ran} f$ *and* $\operatorname{swap} s \neq \ker f$;
(v) there exist $f, s_1, s_2 \in F$ *such that* $|\ker f| = 2$ *and* s_1, s_2 *are swap permutations satisfying the condition* $\operatorname{swap} s_1 \neq \operatorname{swap} s_2$.

Proof. Case (i) is obvious. Case (ii) follows from Corollary 14. We notice here that the existence of f and g satisfying $\operatorname{ran} f \subseteq \ker g$ implies either $\operatorname{ran} f = \ker g$, or $|\ker g| = 3$, *i.e* g is a constant.

Case (iii) is obvious by (ii) if $\operatorname{ran} f = \ker f$. Otherwise, let $\operatorname{ran} f = \{x, y\}$ and $\ker f = \{y, z\}$ with $x, y, z \in D$ all different. Notice that $\ker(c^2 \circ f) = \ker(c \circ f) = \ker f$. There are two possibles cases: (1) $\operatorname{ran}(c \circ f) = \{y, z\}$, hence by (ii) $c \circ f$ is a constant function; (2) $\operatorname{ran}(c \circ f) = \{x, z\}$. It is easy to see that $\operatorname{ran}(c^2 \circ f) = \{y, z\}$. By (ii), $c^2 \circ f$ is a constant function.

Case (iv) is also obvious by (ii) if $\operatorname{ran} f = \ker f$. Otherwise, we have $s(\operatorname{ran} f) = \ker f$. Thus, for all $x \in D$ we have $(s \circ f)(x) = \ker f$, *i.e.* $\operatorname{ran}(s \circ f) = \ker f$, and we conclude by (ii) that $(f \circ s \circ f)$ is a constant function.

Case (v) is obvious by (ii) if $\operatorname{ran} f = \ker f$. Otherwise, since we have $\operatorname{swap} s_1 \neq \operatorname{swap} s_2$, the composition $s_1 \circ s_2$ necessarily produces a circular permutation. We can conclude by (iii). □

We will see now that the conditions listed in Proposition 18 are necessary to get a constant function in $[F]$.

Proposition 19. *Let F be a set of functions satisfying no condition from Proposition 18. Then there is no constant function in $[F]$.*

Proof. If F does not verify any condition of Proposition 18, then we are in one of these cases:

(1) F contains only permutations;
(2) for each $f \in F$ we have $|\ker f_i| = 2$ and for all $f_i, f_j \in F$ (eventually $f_i = f_j$), we have $\operatorname{ran} f_i \neq \ker f_j$ and $\operatorname{ran} f_j \neq \ker f_i$.;
(3) F satisfies Condition 2 and contains also swap permutations with the same swap set, such that for all $f_i \in F$ with $|\ker f_i| = 2$, for all swap permutations $s_k \in F$, we have $\operatorname{swap} s_k = \ker f_i$ or $\operatorname{swap} s_k = \operatorname{ran} f_i$. Notice that both are impossible because we have $\ker f_i \neq \operatorname{ran} f_i$.

Case (1) is obvious. If F is a set of permutations, then $[F]$ is a set of permutations, too. By the pigeonhole principle, case (2) implies $\ker f_i = \ker f_j$, or $\operatorname{ran} f_i = \operatorname{ran} f_j$, or both, for all $f_i, f_j \in F$. From Corollary 16, we know that $\operatorname{ran} f_i = \operatorname{ran} f_j$ for all $f_i, f_j \in F$ implies that every $f \in [F]$ verifies the equality $\operatorname{ran} f = \operatorname{ran} f_i$. Since $|\ker f_i| = 2$ implies $|\ker f| = 2$, so f cannot be a constant function. We can apply the same argument for the case where $\ker f_i = \ker f_j$ for all $f_i, f_j \in F$.

Like in case (2), we have for all $f_i, f_j \in F$ the equations $\ker f_i = \ker f_j$, or $\operatorname{ran} f_i = \operatorname{ran} f_j$, or both. We distinguish four cases. Let $\operatorname{ran} f_i = \operatorname{ran} f_j$, for all $f_i, f_j \in F$, and $\operatorname{swap} s_k = \operatorname{ran} f_i$. It is clear that $\operatorname{ran}(s_k \circ f_i) = \operatorname{ran}(f_i \circ s_k) = \operatorname{ran} f_i$. Since $|\ker f_i| = 2$, we also have $|\ker(s_k \circ f_i)| = |\ker(f_i \circ s_k)| = 2$, hence $s_k \circ f_i$ and $f_i \circ s_k$ cannot be constant functions. Now let $\operatorname{swap} s_k = \ker f_i$, for a $f_i \in F$. We must have $\ker f_i = \ker f_j$ for all $f_i, f_j \in F$ because otherwise there exists $f_k \in F$ such that $\operatorname{swap} s_k = \operatorname{ran} f_k$, and thus $\ker f_k = \operatorname{ran} f_i$ which constitutes a contradiction. Therefore we have $\ker(s_k \circ f_i) = \ker(f_i \circ s_k) = \ker f_i$. Since $|\ker f_i| = 2$, we conclude that there is no composition producing a constant function. With the same arguments, we see that we cannot get a constant function if $\ker f_i = \ker f_j$ for all $f_i, f_j \in F$, whenever $\operatorname{swap} s_k = \operatorname{ran} f_i$ or $\operatorname{swap} s_k = \ker f_i$. □

Theorem 20. *Given a set of functions F on a ternary domain, the problem to know whether the clone $[F]$ contains a constant function can be decided in polynomial time.*

Proof. By Propositions 18 and 19, we know that $[F]$ contains a constant function if and only if F satisfies one of the conditions in Proposition 18. The satisfiability test of each condition can be done in polynomial time. For case (i), we have to test for all $f \in F$ whether f is a constant function. This condition can be verified in time $O(|F|)$. For case (ii), we are looking for $f, g \in F$, for which we compute $\ker f$ and $\operatorname{ran} g$, such that $\operatorname{ran} g \subseteq \ker f$. The verification of condition (ii) is done in time $O(|F|^2)$. For case (iii), we are looking for $f \in F$ such that $|\ker f| = 2$ and for a circular permutation $c \in F$. This can be verified in time $O(|F|)$. For case (iv), we have to check for all $f, s \in F$ if $|\ker f| = 2$, if s is a swap permutation, followed by a check if $\operatorname{swap} s \neq \ker f$ and $\operatorname{swap} s \neq \operatorname{ran} f$. This can be verified in time $O(|F|^2)$. Finally for case (v), we are looking for $f \in F$ such that $|\ker f| = 2$ and for $s_1, s_2 \in F$ such that s_1 and s_2 are swap permutations, such that $\operatorname{swap} s_1 \neq \operatorname{swap} s_2$. This is verified in time $O(|F|^2)$. □

We have shown a polynomial-time algorithm concerning the meta-problem on a ternary domain. Despite its complexity in $O(|F|^2)$ which is less efficient than the general meta-problem algorithm in $O(n^n.\,|F|)$, with the constant $n = 3$ for the ternary case, this algorithm use a method allowing to skip the computation, even partially, of the clone $[F]$. Thus, this method constitute a serious approach for obtaining polynomial-time algorithms more efficient than the general meta-problem algorithm.

7 Concluding Remarks

We analyzed the computational complexity of monotone constraint satisfaction problems, allowing also the disjunction connective to be applied. We obtained a complete characterization expressed by a Dichotomy Theorem, distinguishing between tractable and NP-complete instances. The tractability condition turned out to be the closure of the constraints under a constant function. Since the endomorphism set $\mathrm{End}\,S$ is equal to the clone $[F]$ generated from a set of unary functions F, it is also interesting to study the meta-problem of the tractability condition. This means, given a set of unary functions F, whether the clone $[F]$ contains a constant function. We showed that the meta-problem is NP-complete if the domain is part of the input, but it is fixed-parameter tractable, with an algorithm running in time $O(n^n\,|F|)$, if we consider the domain as a parameter. We performed a special complexity analysis for the meta-problem of the ternary domain, for which we deduced conditions ensuring the presence of a constant function in the clone $[F]$ without the necessity of computing (at least a part of) the functions in $[F]$.

References

1. A. A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *Journal of the Association for Computing Machinery*, 53(1):66–120, 2006.
2. D. Cohen, P. Jeavons, P. Jonsson, and M. Koubarakis. Building tractable disjunctive constraints. *Journal of the Association for Computing Machinery*, 47(5):826–853, 2000.
3. R. G. Downey and M. R. Fellows. *Parametrized Complexity*. Springer-Verlag, 1999.
4. T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.
5. P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1-2):185–204, 1998.
6. M. Krasner. Une généralisation de la notion de corps. *Journal de Mathématiques pures et appliquées*, 17:367–385, 1938.
7. R. Pöschel. Galois connections for operations and relations. In K. Denecke *et al*, editors, *Galois Connections and Applications*, pages 231–258. Kluwer, 2004.
8. A. Salomaa. Composition sequences for functions over a finite domain. *Theoretical Computer Science*, 292(1):263–281, 2003.
9. T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th Symposium on Theory of Computing (STOC'78), San Diego (California, USA)*, pages 216–226, 1978.
10. Yu. I. Yanov and A. A. Muchnik. On the existence of k-valued closed classes that have no bases. *Doklady Akademii Nauk SSSR*, 127:44–46, 1959. In Russian.