On the complexity of the clone membership problem

Emil Jeřábek

The Czech Academy of Sciences, Institute of Mathematics

Žitná 25, 115 67 Praha 1, Czech Republic, email: jerabek@math.cas.cz

September 22, 2020

Abstract

We investigate the complexity of the Boolean clone membership problem (CMP): given a set of Boolean functions F and a Boolean function f, determine if f is in the clone generated by F, i.e., if it can be expressed by a circuit with F-gates. Here, f and elements of F are given as circuits or formulas over the usual De Morgan basis. Böhler and Schnoor [7] proved that for any fixed F, the problem is **coNP**-complete, with a few exceptions where it is in **P**. Vollmer [30] incorrectly claimed that the full problem CMP is also **coNP**-complete. We prove that CMP is in fact $\mathbf{\Theta}^{\mathbf{P}}_2$-complete, and we complement Böhler and Schnoor's results by showing that for fixed f, the problem is **NP**-complete unless f is a projection.

More generally, we study the problem B-CMP where F and f are given by circuits using gates from B. For most choices of B, we classify the complexity of B-CMP as being $\mathbf{\Theta}^{\mathbf{P}}_2$-complete (possibly under randomized reductions), **coDP**-complete, or in **P**.

1 Introduction

The fundamental concept of a *clone* originated in universal algebra as an abstraction of the notion of an algebra—a clone consists of operations term-definable in a given algebra; it reappears in logic in the form of a collection of truth functions (in a possibly multi-valued logic) definable by formulas over a given set of connectives, and in computer science as a set of functions computable by circuits over a given basis of gates. Clones have many applications in theoretical computer science in classification of the complexity of CSPs [27, 8, 33] or satisfiability and related problems [19, 24], including various nonclassical logics [2, 6, 12, 22]. The most basic computational problem associated with clones is the *clone membership problem*, asking if a given function is expressible (definable by a term, formula, or circuit) from a given set of initial functions. This problem has computer science applications in the context of formal algebraic specifications [26].

From the point of view of computational complexity, several variants of the clone membership problem were studied in the literature. The most straightforward representation of the input functions is by tables of values. In this setting, Kozen [15] proved that the membership problem for clones of *unary* functions on arbitrary finite domains is **PSPACE**-complete. The general clone membership problem for arbitrary functions on finite domains is **EXP**-complete. This result is credited in [5] to an unpublished manuscript of H. Friedman, but the first published proof is due to Bergman, Juedes, and Slutzki [4]; a mistake in their paper was corrected by Mašulović [21]. As shown by Kozik [16], there even exists a *fixed* finitely generated clone on a finite domain whose membership problem is **EXP**-complete.

The high complexity of the problem on arbitrary finite domains is related to the complicated structure of clones on domains of size ≥ 3. In contrast, the lattice of clones on the *Boolean* (two-element) domain is quite simple, and it has been explicitly described by Post [23]. As a result, the Boolean clone membership problem is computationally much easier than the general case: it was shown to be in **NL** by Bergman and Slutzki [5], while Vollmer [30] proved that it was in quasipolynomial $\mathbf{AC}^0$, which implies it is not **NL**-hard (or even $\mathbf{AC}^0[2]$-hard).

The representation of functions by tables is quite inefficient, as it always has size exponential in the number of variables. A viable alternative, especially in the Boolean case, is to represent functions by expressions (circuits or formulas) over some canonically chosen functionally complete basis, say, the De Morgan basis $\{\wedge, \vee, \neg\}$. In this setting, Böhler and Schnoor [7] studied the complexity of membership problems for *fixed* Boolean clones C: they proved that all such problems are **coNP**-complete with a few exceptions that are in **P**. More generally, they studied variants of the problem where f is not given by a circuit over a functionally complete basis, but over an arbitrary (but fixed) basis. They classified most such problems as being **coNP**-complete or in **P**.

The full Boolean clone membership problem in the circuit representation (denoted CMP in this paper) was considered by Vollmer [30], who claimed it was also **coNP**-complete. However, he did not provide much in the way of proof for the **coNP** upper bound (see Remark 3.5 below), and as we will see, this claim is wrong.

A characterization of clone membership in terms of preservation of relational invariants easily implies that CMP is computable in $\mathbf{P}^{\mathbf{NP}}$—more precisely, in the class $\mathbf{\Theta}_2^{\mathbf{P}} = \mathbf{P}^{\mathbf{NP}[\log]} = \mathbf{P}^{\|\mathbf{NP}}$. The main goal of this paper is to prove that CMP is in fact $\mathbf{\Theta}_2^{\mathbf{P}}$-complete. As a warm-up, we consider a restriction of CMP dual to Böhler and Schnoor's results: we prove that for a fixed target Boolean function f, the clone membership problem is **NP**-complete in all nontrivial cases (i.e., unless f is a projection function, or a nullary function if we allow them). This already shows that CMP cannot be **coNP**-complete unless **NP** = **coNP**. We then go on to prove that CMP is $\mathbf{\Theta}_2^{\mathbf{P}}$-complete; our main technical tool is a characterization of clones generated by threshold functions. We also discuss some variants of our results, such as using formulas instead of circuits for representation of functions, or allowing nullary functions.

In the second part of the paper, we investigate the complexity of restricted versions of CMP, denoted B-CMP, where the input functions are given by circuits or formulas over an arbitrary (but fixed) finite basis B instead of the De Morgan basis. We show that B-CMP remains $\mathbf{\Theta}_2^{\mathbf{P}}$-complete, albeit using randomized reductions, when the clone $[B]$ generated by B has infinitely many subclones, and includes some non-monotone functions; we rely on a randomized construction of formulas for threshold functions using fixed threshold functions as gates, following the method of Valiant [29]. On the other hand, if $[B]$ has only finitely many subclones, we classify the complexity of B-CMP as either **coDP**-complete or in **P**. The complexity of B-CMP remains open when $[B]$ has infinitely many subclones, but consists of monotone functions only.

2 Preliminaries

2.1 Boolean functions and clones

We will assume basic familiarity with the theory of Boolean clones; this is described in many places, for example Lau [18]. We will summarize the most important points below to fix our terminology and notation.

Let $\mathbf{2} = \{0, 1\}$. An *n-ary Boolean function* (or *operation*) is a mapping $f \colon \mathbf{2}^n \to \mathbf{2}$. We denote the set of n-ary Boolean functions by Op_n, and the set of all Boolean functions by $\mathrm{Op} = \bigcup_{n \ge 1} \mathrm{Op}_n$. (Following the tradition in literature on Boolean clones, we disallow nullary functions; we will comment later on how this affects our results.)

We will use common connectives such as $\wedge, \vee, \to, \neg$ to denote specific Boolean functions; by abuse of notation, 0 and 1 denote the constant functions of arbitrary arity. If $0 \le i < n$, the *projection function* $\pi_i^n \in \mathrm{Op}_n$ is defined by $\pi_i^n(x_0, \dots, x_{n-1}) = x_i$. For any $n > 0$ and $0 \le t \le n+1$, the *threshold function* $\theta_t^n \in \mathrm{Op}_n$ is defined by

$$\theta_t^n(x_0, \dots, x_{n-1}) = 1 \iff \bigl|\{i < n : x_i = 1\}\bigr| \ge t.$$

Notice that $\theta_0^n = 1$, $\theta_{n+1}^n = 0$, $\theta_1^2 = \vee$, $\theta_2^2 = \wedge$, and $\theta_1^1 = \pi_1^1$ is the identity function.

Given functions $f \in \mathrm{Op}_n$ and $g_0, \dots, g_{n-1} \in \mathrm{Op}_m$, their *composition* is the function $h \in \mathrm{Op}_m$ defined by

$$h(\vec{x}) = f\bigl(g_0(\vec{x}), \dots, g_{n-1}(\vec{x})\bigr).$$

A set of Boolean functions $C \subseteq \mathrm{Op}$ is a *clone* if it contains all projections and is closed under composition. The intersection of an arbitrary collection of clones is again a clone (where the empty intersection is understood as Op), thus the poset of clones under inclusion forms a complete lattice, and it yields an (algebraic) closure operator $[-] \colon \mathcal{P}(\mathrm{Op}) \to \mathcal{P}(\mathrm{Op})$; that is, for any $F \subseteq \mathrm{Op}$, $[F]$ denotes the clone generated by F.

The *dual* of a Boolean function f is the function $f^{\mathrm{d}}(x_0, \dots, x_{n-1}) = \neg f(\neg x_0, \dots, \neg x_{n-1})$. The dual of a clone C is the clone $\{f^{\mathrm{d}} : f \in C\}$.

A *k-ary Boolean relation* is $r \subseteq \mathbf{2}^k$. The set of k-ary Boolean relations is denoted by Rel_k, and the set of all relations by $\mathrm{Rel} = \bigcup_k \mathrm{Rel}_k$. (Here, it will not make any difference if we allow nullary relations or not; the reader is welcome to make the choice.) A function $f \in \mathrm{Op}_n$ *preserves* a relation $r \in \mathrm{Rel}_k$, written as $f \triangleright r$, if f, considered as a mapping of the relational structures $\langle \mathbf{2}, r \rangle \times \cdots \times \langle \mathbf{2}, r \rangle \to \langle \mathbf{2}, r \rangle$, is a homomorphism. Explicitly, $f \triangleright r$ iff the following implication holds for every matrix $(a_i^j)_{i<n}^{j<k} \in \mathbf{2}^{k \times n}$:

$$\begin{pmatrix} a_0^0 \\ \vdots \\ a_0^{k-1} \end{pmatrix} \in r, \dots, \begin{pmatrix} a_{n-1}^0 \\ \vdots \\ a_{n-1}^{k-1} \end{pmatrix} \in r \implies \begin{pmatrix} f(a_0^0, \dots, a_{n-1}^0) \\ \vdots \\ f(a_0^{k-1}, \dots, a_{n-1}^{k-1}) \end{pmatrix} \in r.$$

For example, the relation $r = \{\langle x, y \rangle : x \vee y = 1\}$ is not preserved by the function $f(x, y, z) = x \wedge (y \vee z)$, as witnessed by the matrix $\left(\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{smallmatrix}\right)$: every column is in r, but applying f row-wise gives the vector $\left(\begin{smallmatrix} 0 \\ 0 \end{smallmatrix}\right) \notin r$. On the other hand, θ_2^3 preserves r (and all other 2-ary relations for that matter): for any $(a_i^j)_{i<3}^{j<2}$ and each $j<2$, $\theta_2^3(a_0^j,a_1^j,a_2^j)\neq a_i^j$ for at most one $i<3$, hence there exists $i<3$ such that

$$\begin{pmatrix}\theta_2^3(a_0^0,a_1^0,a_2^0)\\ \theta_2^3(a_0^1,a_1^1,a_2^1)\end{pmatrix}=\begin{pmatrix}a_i^0\\ a_i^1\end{pmatrix}.$$

If $F\subseteq\mathrm{Op}$ and $R\subseteq\mathrm{Rel}$, we write $F\vartriangleright R$ if $f\vartriangleright r$ for all $f\in F$ and $r\in R$. The set of *invariants* of $F\subseteq\mathrm{Op}$ and the set of *polymorphisms* of $R\subseteq\mathrm{Rel}$ are defined by

$$\begin{aligned}\mathrm{Inv}(F)&=\{r\in\mathrm{Rel}:F\vartriangleright r\},\\ \mathrm{Pol}(R)&=\{f\in\mathrm{Op}:f\vartriangleright R\}.\end{aligned}$$

We will use the following fundamental properties of the Inv and Pol operators (see [18, Thms. 2.6.1, 2.9.1]):

Fact 2.1 *The mappings* $\mathrm{Inv}\colon\mathcal{P}(\mathrm{Op})\to\mathcal{P}(\mathrm{Rel})$ *and* $\mathrm{Pol}\colon\mathcal{P}(\mathrm{Rel})\to\mathcal{P}(\mathrm{Op})$ *form an antitone Galois connection:* $R\subseteq\mathrm{Inv}(F)\iff F\subseteq\mathrm{Pol}(R)$.

The Galois-closed subsets of Op *are exactly the clones: i.e.,* $\mathrm{Pol}(R)$ *is a clone for every* $R\subseteq\mathrm{Rel}$*, every clone is of the form* $\mathrm{Pol}(R)$ *for some* $R\subseteq\mathrm{Rel}$*, and* $[F]=\mathrm{Pol}(\mathrm{Inv}(F))$ *for all* $F\subseteq\mathrm{Op}$.

If we allowed nullary functions, then Galois-closed subsets of Rel would be exactly the *coclones*: subsets $R\subseteq\mathrm{Rel}$ that are closed under definitions by *primitive positive* formulas, i.e., first-order formulas $\varphi(\vec{x})$ of the form $\exists\vec{y}\bigwedge_{i<k}\psi_i(\vec{x},\vec{y})$, where each ψ_i is an atomic formula (an instance of a relation $r\in R$, or of equality). Under our restriction to non-nullary functions, Galois-closed subsets of Rel are only the coclones that contain the empty relation $\varnothing\in\mathrm{Rel}_1$. (See Behrisch [3] for a detailed discussion.)

The lattice of Boolean clones was completely described by Post [23]. In particular, we will make use of the following characterization (which follows from [28, pp. 36ff] or [18, Thm. 3.1.1]), fixing our naming of basic clones and their invariants along the way. Here, for any $f\in\mathrm{Op}_n$, $\mathrm{gr}(f)=\{\langle\vec{x},y\rangle\in\mathbf{2}^{n+1}:y=f(\vec{x})\}\in\mathrm{Rel}_{n+1}$ denotes the graph of f. Recall that an element a of a lattice L is *meet-irreducible* if $a=\bigcap S$ implies $a\in S$ for any finite $S\subseteq L$, where $\bigcap$ denotes the lattice meet operation; if this holds also for infinite $S\subseteq L$, it is *completely meet-irreducible.*

Fact 2.2 *Every Boolean clone is an intersection of a family of completely meet-irreducible clones, which are:*

- *The clone* $\mathrm{M}=[\wedge,\vee,0,1]=\mathrm{Pol}(\leq)$ *of* monotone *functions, where* $\leq$ *denotes the relation* $\{\langle 0,0\rangle,\langle 0,1\rangle,\langle 1,1\rangle\}\in\mathrm{Rel}_2$.
- *The clone* $\mathrm{A}=[+,1]=\mathrm{Pol}(r_A)$ *of* affine *functions, where* $+$ *denotes addition in the two-element field* $\mathbb{F}_2$ *(i.e., XOR), and* $r_A=\{\langle x,y,z,w\rangle:x+y+z+w=0\}\in\mathrm{Rel}_4$.
- *The clone* $\mathrm{D}=[\theta_2^3,\neg]=\mathrm{Pol}\big(\mathrm{gr}(\neg)\big)$ *of* self-dual *functions (i.e., functions* f *such that* $f=f^{\mathrm{d}}$*).*
- *The clone* $\bigwedge=[\wedge,0,1]=\mathrm{Pol}\big(\mathrm{gr}(\wedge)\big)$ *of* conjunctive *functions.*
- *The clone* $\bigvee=[\vee,0,1]=\mathrm{Pol}\big(\mathrm{gr}(\vee)\big)$ *of* disjunctive *functions.*

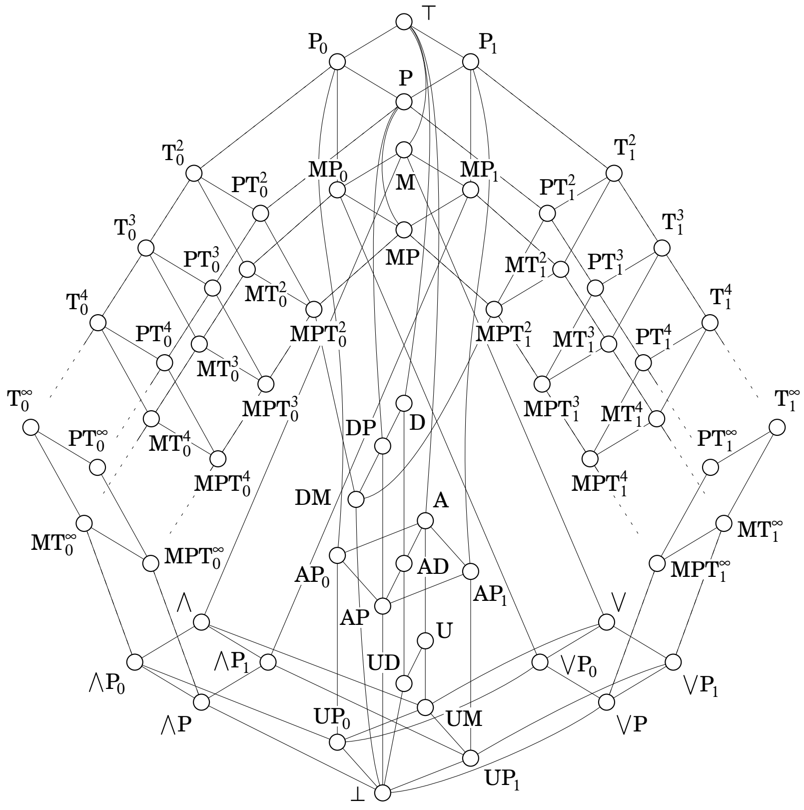

Figure 1: The lattice of Boolean clones (Post's lattice).

- *The clone* $\mathrm{U} = [\neg, 0] = \mathrm{Pol}(r_U)$ *of* essentially unary *functions* (*i.e., functions that depend on at most one variable*), *where* $r_U = \big\{\langle x, y, z\rangle : z \in \{x, y\}\big\} \in \mathrm{Rel}_3$.
- *For each* $m \geq 1$*, the clones* $\mathrm{T}_1^m = [\theta_2^{m+1}, \to] = \mathrm{Pol}(r_1^m)$ *and* $\mathrm{T}_0^m = [\theta_m^{m+1}, \not\to] = \mathrm{Pol}(r_0^m)$, *where* $x \not\to y = \neg(x \to y) = x \wedge \neg y$*, and the relations* $r_\alpha^m \in \mathrm{Rel}_m$ *are defined by*

$$\begin{aligned} r_1^m &= \{\vec{x} \in \mathbf{2}^m : x_0 \vee \cdots \vee x_{m-1} = 1\}, \\ r_0^m &= \{\vec{x} \in \mathbf{2}^m : x_0 \wedge \cdots \wedge x_{m-1} = 0\}. \end{aligned}$$

  Since $r_\alpha^1 = \{\alpha\}$*, this includes as a special case the clones* $\mathrm{P}_1 = [\wedge, \to] = \mathrm{T}_1^1$ *and* $\mathrm{P}_0 = [\vee, \not\to] = \mathrm{T}_0^1$ *of* 1-preserving *and* 0-preserving *functions* (*respectively*).

We will denote intersections of named clones by juxtaposition, so that, e.g., $\mathrm{AD} = \mathrm{A} \cap \mathrm{D}$. For convenience, we also put $\mathrm{P} = \mathrm{P}_0\mathrm{P}_1$ and $\mathrm{T}_\alpha^\infty = \bigcap_m \mathrm{T}_\alpha^m$. We have $\mathrm{T}_1^\infty = [\to]$ and $\mathrm{T}_0^\infty = [\not\to]$. We will denote the top and bottom of the lattice of clones by $\top$ and $\bot$, i.e., $\top = \mathrm{Op}$, and $\bot = \{\pi_i^n : i < n\} = \mathrm{UP}$. We define

$$R_n = \{\leq, r_A, \mathrm{gr}(\neg), \mathrm{gr}(\wedge), \mathrm{gr}(\vee), r_U\} \cup \{r_\alpha^m : 0 < m \leq n, \alpha \in \mathbf{2}\} \subseteq \mathrm{Rel}$$

for each $n \geq 0$, and $R_\infty = \bigcup_n R_n$.

The Hasse diagram of the lattice of Boolean clones (called *Post's lattice*) is depicted in Fig. 1. (In fact, Post [23] did not work with the modern definition of clones, but with a slightly weaker concept of *iterative classes*, which do not necessarily contain all projections. Thus, his original lattice has four more classes.)

2.2 Computational complexity

We assume the reader is familiar with basic notions of complexity theory, including the classes **P**, **NP**, and **coNP**, and logspace functions.

If C is a complexity class, we say that a language L is *C-hard* if for every $L' \in C$, there exists a logspace many-one reduction from L' to L; if, moreover, $L \in C$, then L is *C-complete*. A reader familiar with uniform $\mathbf{TC}^0$ functions (see e.g. [1]) may easily find that all our results below actually hold with $\mathbf{TC}^0$ many-one reductions in place of logspace reductions. (We believe that the bulk of our results actually hold under even more restricted notions of reductions such as dlogtime or $\mathbf{AC}^0$, but this might require extra effort, or possibly a relaxed syntactic representation of circuits and formulas allowing for insertion of padding.)

Let $B \subseteq \mathrm{Op}$ be finite. A *(Boolean) circuit over a basis B* (or *B-circuit* for short) in n variables is a labelled directed acyclic graph C (possibly with multiple edges) with exactly one node of out-degree 0 (the output node), such that each node of C is either labelled with one of the variables x_i, $i < n$, in which case it has in-degree 0, or it is labelled with a k-ary function $f \in B$, in which case it has in-degree k, and it comes with an explicit numbering of the incoming edges (such a node is called an *f-gate*). A circuit computes a function $\mathbf{2}^n \to \mathbf{2}$ in the usual way. A function $f \in \mathrm{Op}$ is computable by a B-circuit iff $f \in [B]$. Given a B-circuit $C\colon \mathbf{2}^n \to \mathbf{2}$ and an assignment $a \in \mathbf{2}^n$, we can evaluate $C(a)$ in polynomial time; moreover, evaluation of circuits over the *De Morgan basis* $\{\wedge, \vee, \neg\}$ is **P**-complete (Ladner [17]). The *depth* of a circuit is the maximal length of a path from a leaf to the output node.

A *B-formula* is a B-circuit such that all nodes have out-degree at most 1. Formulas are usually represented as strings in a more economical way than directed graphs: the *prefix* (*Polish*) notation can be defined inductively such that each variable x_i is a B-formula, and if $f \in B$ is k-ary, and $\varphi_0, \dots, \varphi_{k-1}$ are B-formulas, then $f\varphi_0 \dots \varphi_{k-1}$ is a B-formula. If all $f \in B$ have arity at most 2, we may also use the common *infix* notation where $(\varphi_0\, f\, \varphi_1)$ is employed instead of $f\varphi_0\varphi_1$ for f of arity 2. Evaluation of formulas over any finite basis is in $\mathbf{NC}^1$, and over the De Morgan basis it is $\mathbf{NC}^1$-complete under dlogtime reductions (Buss [9]). Here, $\mathbf{NC}^1$ is the class of languages computable in time $O(\log n)$ on an alternating Turing machine (with random-access input); equivalently, it consists of languages computable by a dlogtime-uniform (more precisely, U_E-uniform) sequence of De Morgan circuits of depth $O(\log n)$, see Ruzzo [25] for details. We have $\mathbf{NC}^1 \subseteq \mathbf{L}$.

By duplicating nodes as needed, we can unravel any B-circuit of depth d into a B-formula of depth d, and therefore of size $2^{O(d)}$; in particular, any circuit of depth $O(\log n)$ can be transformed into a polynomial-size formula. One consequence of this is that we can efficiently convert formulas of depth $O(\log n)$ to a different basis (the corresponding result for circuits being trivial), which will be useful in Section 4:

Lemma 2.3 *Let $B, B' \subseteq \mathrm{Op}$ be finite sets such that $B \subseteq [B']$.*

(i) *Given a B-circuit, we can construct in logspace an equivalent B'-circuit.*

(ii) *Given a B-formula of depth $O(\log n)$, we can construct in logspace an equivalent B'-formula.*

Proof:

(i): For each $f \in B$, we fix an expression of f in terms of B', and replace with it all f-gates in the circuit.

(ii): Starting from a B-formula of depth $O(\log n)$, the construction above produces a B'-circuit of depth $O(\log n)$, which can be unravelled into a B'-formula of depth $O(\log n)$, and consequently of size $n^{O(1)}$. $\square$

The *Boolean clone membership problem* CMP is the following decision problem:

> **Input:** A finite set of functions $F \subseteq \mathrm{Op}$ and a function $f \in \mathrm{Op}$, all given by Boolean circuits over the De Morgan basis.
>
> **Output:** YES if $f \in [F]$, otherwise NO.

For any clone $C = [C]$, the membership problem CMP_C is the special case of CMP where F is fixed:

> **Input:** A function $f \in \mathrm{Op}$, given by a Boolean circuit over the De Morgan basis.
>
> **Output:** YES if $f \in C$, otherwise NO.

Dually, for a fixed function $f \in \mathrm{Op}$, CMP^f denotes the following special case of CMP:

> **Input:** A finite set of functions $F \subseteq \mathrm{Op}$, given by Boolean circuits over the De Morgan basis.
>
> **Output:** YES if $f \in [F]$, otherwise NO.

Notice that $\mathrm{CMP}^f = \mathrm{CMP}^g$ whenever $[f] = [g]$, as the output condition can be stated as $[f] \subseteq [F]$.

For any class C and $q \colon \mathbb{N} \to \mathbb{N}$, $\mathbf{P}^{C[q]}$ denotes the class of languages computable in polynomial time using at most $q(n)$ queries to an oracle $A \in C$; similarly for a set of such functions q. The class $\mathbf{P}^{\parallel C}$ consists of languages computable in polynomial time with non-adaptive (parallel) access to an oracle $A \in C$; we also write $\mathbf{P}^{\parallel C[q]}$ if the number of oracle queries is bounded by $q(n)$.

The class $\mathbf{\Theta}_2^{\mathbf{P}}$, introduced by Wagner [31], is defined as $\mathbf{P}^{\mathbf{NP}[O(\log n)]}$. It has several other equivalent characterizations, see Buss and Hay [10]: in particular, $\mathbf{\Theta}_2^{\mathbf{P}} = \mathbf{P}^{\parallel \mathbf{NP}}$.

The *Boolean hierarchy* $\mathbf{BH}$ is defined as the smallest class that includes $\mathbf{NP}$, and is closed under (finitary) intersections, unions, and complements; alternatively, it can be characterized as $\mathbf{BH} = \mathbf{P}^{\mathbf{NP}[O(1)]}$. The Boolean hierarchy is stratified into levels: the bottom level consists of $\mathbf{NP}$ and $\mathbf{coNP}$, and the next level of the classes

$$\mathbf{DP} = \{L_0 \cap L_1 : L_0 \in \mathbf{NP}, L_1 \in \mathbf{coNP}\},$$
$$\mathbf{coDP} = \{L_0 \cup L_1 : L_0 \in \mathbf{NP}, L_1 \in \mathbf{coNP}\}.$$

We will use the observation that for any **NP**-complete language L_0 and a **coNP**-complete language L_1, the language

$$L = \{\langle x, y\rangle : x \in L_0 \text{ or } y \in L_1\}$$

is **coDP**-complete: given a **coDP** language $L' = L'_0 \cup L'_1$, where $L'_0 \in \mathbf{NP}$ and $L'_1 \in \mathbf{coNP}$, we may fix reductions f_i of L'_i to L_i for $i = 0, 1$; then $w \mapsto \langle f_0(w), f_1(w)\rangle$ is a reduction of L' to L.

3 The complexity of CMP

In this section, we will investigate the complexity of CMP and its subproblems CMP_C and CMP^f. We will start with upper bounds ($\mathrm{CMP} \in \mathbf{\Theta}^{\mathbf{P}}_2$, $\mathrm{CMP}_C \in \mathbf{coNP}$, $\mathrm{CMP}^f \in \mathbf{NP}$) in the first subsection. The corresponding lower bounds will follow in Section 3.2 for CMP_C and CMP^f, and in Section 3.3 for CMP (the $\mathbf{\Theta}^{\mathbf{P}}_2$-completeness of CMP will be the main result of this section). We discuss how these results are affected if we vary the definition of CMP and its subproblems in Section 3.4.

3.1 Upper bounds

Similarly to [5, 7, 30], we will extract an algorithm for CMP from Fact 2.1, which implies

$$f \in [F] \iff \mathrm{Inv}(F) \subseteq \mathrm{Inv}(f) \iff \forall r \in \mathrm{Rel}\,(F \not\triangleright r \text{ or } f \triangleright r). \tag{1}$$

Using Fact 2.2, we may restrict attention to $r \in R_\infty$, but this is still an infinite number of invariants due to the two infinite families r_0^m and r_1^m, $m \in \mathbb{N}$, which correspond to the two infinite arms of Post's lattice. In order to turn (1) into an algorithmic criterion, we need to place an efficient bound on m in terms of F and f, which is the content of the next lemma.

Lemma 3.1 *Let $n \geq 1$, $f \in \mathrm{Op}_n$, and $\alpha \in \mathbf{2}$. The following are equivalent:*

(i) $f \in \mathrm{T}^\infty_\alpha$.

(ii) $f \in \mathrm{T}^n_\alpha$.

(iii) *There is $i < n$ such that $x_i \leq^\alpha f(\vec{x})$ for all $\vec{x} \in \mathbf{2}^n$, where $\leq^1 = \leq$, $\leq^0 = \geq$.*

Proof: (i) $\to$ (ii) is trivial.

(iii) $\to$ (i): Assume $\alpha = 1$ and $x_{i_0} \leq f(\vec{x})$. If $m \in \mathbb{N}$ and $(a_i^j)_{i<n}^{j<m}$ is such that $\bigvee_{j<m} a_i^j = 1$ for all $i < n$, then in particular $\bigvee_{j<m} a_{i_0}^j = 1$, hence $\bigvee_{j<m} f(a_0^j, \dots, a_{n-1}^j) = 1$ by assumption.

(ii) $\to$ (iii): If (iii) does not hold, let us fix for each $i < n$ a vector $a^i = \langle a_0^i, \dots, a_{n-1}^i\rangle \in \mathbf{2}^n$ such that $a_i^i = \alpha$ and $f(a^i) = \neg\alpha$. Then the matrix $(a_i^j)_{i<n}^{j<n}$ witnesses that $f \not\triangleright r_\alpha^n$: if, say, $\alpha = 1$, we have $a_i^0 \vee \dots \vee a_i^{n-1} \geq a_i^i = 1$ for each $i < n$, but $f(a^0) \vee \dots \vee f(a^{n-1}) = 0$. $\square$

Corollary 3.2 *If $F \subseteq \mathrm{Op}$ and $f \in \mathrm{Op}_n$, then*

$$f \in [F] \iff \forall r \in R_n\,(F \not\triangleright r \text{ or } f \triangleright r). \tag{2}$$

Proof: The left-to-right implication is clear. For the right-to-left implication, if $f \notin [F]$, then one of the completely meet-irreducible clones C as given in Fact 2.2 satisfies $F \subseteq C$ and $f \notin C$. Moreover, if $C = \mathrm{T}^m_\alpha$ for some $\alpha \in \mathbf{2}$ and $m > n$, then also $f \notin \mathrm{T}^n_\alpha \supseteq C \supseteq F$ by Lemma 3.1. Thus, we may assume $C = \mathrm{Pol}(r)$ for some $r \in R_n$, i.e., $F \triangleright r$ and $f \not\triangleright r$. □

Since the criterion in (2) uses the preservation relation $\triangleright$, we need to know its complexity.

Lemma 3.3 *If $f \in \mathrm{Op}$ and $r \in \mathrm{Rel}$ are given by Boolean circuits over any fixed finite basis, we can test whether $f \triangleright r$ in* **coNP**.

Proof: Given circuits computing $f \colon \mathbf{2}^n \to \mathbf{2}$ and $r \subseteq \mathbf{2}^k$ (where we may assume w.l.o.g. that all variables appear in the circuits), we have straight from the definition that $f \not\triangleright r$ iff there exists a matrix $A = (a^j_i)^{j<k}_{i<n} \in \mathbf{2}^{k\times n}$ such that $\langle a^0_i, \dots, a^{k-1}_i \rangle \in r$ for each $i < n$, and

$$\langle f(a^0_0, \dots, a^0_{n-1}), \dots, f(a^{k-1}_0, \dots, a^{k-1}_{n-1}) \rangle \notin r.$$

Here, A is an object of size polynomial in n and k, which are bounded by the size of the input, and the stated properties of A can be checked in polynomial time. □

We can combine Corollary 3.2 and Lemma 3.3 to obtain the desired upper bounds.

Theorem 3.4

(i) $\mathrm{CMP} \in \mathbf{\Theta}^{\mathbf{P}}_2$.

(ii) $\mathrm{CMP}_C \in$ **coNP** *for each clone $C \subseteq \mathrm{Op}$.*

(iii) $\mathrm{CMP}^f \in$ **NP** *for each $f \in \mathrm{Op}$.*

Proof:

(i): Given $F \subseteq \mathrm{Op}$ and $f \in \mathrm{Op}$ of arity n, we can determine if $f \in [F]$ in $\mathbf{P}^{\|\mathbf{NP}}$ by evaluating (2): there are $2n + O(1)$ relations in R_n, and they can be described by efficiently constructible Boolean circuits. Thus, in view of Lemma 3.3, we can ask the **NP** oracle if $F \triangleright r$ and if $f \triangleright r$ for each $r \in R_n$ in parallel, and read the answer off of the oracle responses.

(ii): We use Corollary 3.2 again, but since C is fixed, we can test $C \triangleright r$ in deterministic polynomial time: $\{r \in R_0 : C \triangleright r\}$ is a finite set, and for each $\alpha \in \mathbf{2}$, $\{m \in \mathbb{N}_{>0} : C \triangleright r^m_\alpha\}$ is a downward-closed subset of $\mathbb{N}_{>0}$, i.e., either all of $\mathbb{N}_{>0}$, or a finite set. Thus, (2) can be evaluated in **coNP**.

(iii) is even simpler: since f (hence n) is fixed, R_n is a fixed finite set, and so is $\{r \in R_n : f \not\triangleright r\}$. Thus, we can test if $F \not\triangleright r$ for each r from this finite set in **NP** by Lemma 3.3. □

Remark 3.5 The purported proof of CMP $\in$ **coNP** in Vollmer [30] essentially just states that it follows from a criterion similar to our Corollary 3.2, despite that it involves both positive and negative occurrences of the **coNP** preservation relation.

3.2 Hardness of CMP_C and CMP^f

We turn to lower bounds which will show that Theorem 3.4 is mostly optimal, with a few exceptions in (ii) and (iii). We start with CMP_C and CMP^f, where we can prove (**co**)**NP**-hardness by simple reductions from Boolean satisfiability.

The case of CMP_C was settled by Böhler and Schnoor [7]. For completeness, we include a self-contained proof, employing a reduction function that we will reuse in the argument for CMP^f below.

Theorem 3.6 (Böhler and Schnoor) *Let C be a Boolean clone.*

(i) *If $C \supseteq \mathrm{P}$, then $\mathrm{CMP}_C \in \mathbf{P}$. More precisely, $\mathrm{CMP}_\top$ is trivial, and CMP_C is $\mathbf{P}$-complete for $C = \mathrm{P}_0, \mathrm{P}_1, \mathrm{P}$.*

(ii) *Otherwise (i.e., if $C \subseteq \mathrm{M}$, D, A, T_0^2, or T_1^2), CMP_C is **coNP**-complete.*

Proof:

(i): We have $f \in \mathrm{P}_\alpha$ iff $f(\alpha, \dots, \alpha) = \alpha$, which can be checked in polynomial time. To show the $\mathbf{P}$-hardness of CMP_P and $\mathrm{CMP}_{\mathrm{P}_1}$, let a De Morgan circuit $C(x_0, \dots, x_{n-1})$ and an assignment $\vec{a}$ be given. Put $C_{\vec{a}}(x) = x \wedge C(x^{a_0}, \dots, x^{a_{n-1}})$, where $x^1 = x$, $x^0 = \neg x$. Then $C(\vec{a}) = 1$ iff $C_{\vec{a}} \in \mathrm{P}_1$ iff $C_{\vec{a}} \in \mathrm{P}$. The case of P_0 is dual.

(ii): That C is included in M, D, A, T_0^2, or T_1^2 follows by inspection of Post's lattice (Fig. 1). We have $\mathrm{CMP}_C \in \mathbf{coNP}$ by Theorem 3.4. In order to show **coNP**-hardness, we will provide a reduction from UnSat; it will even be independent of C.

Given a formula φ in variables $\vec{u}$, let

$$f_\varphi(x, y, z, \vec{u}) = \big((x \wedge \varphi) \wedge y\big) \vee \big(\neg(x \wedge \varphi) \wedge z\big).$$

Then

$$\varphi \in \textsc{UnSat} \implies f_\varphi \equiv z \implies f_\varphi \in \bot \subseteq C, \tag{3}$$

$$\varphi \in \textsc{Sat} \implies f_\varphi \notin \mathrm{M}, \mathrm{A}, \mathrm{D}, \mathrm{T}_0^2, \mathrm{T}_1^2 \implies f_\varphi \notin C. \tag{4}$$

Indeed, (3) is obvious. For (4), if $\vec{a}$ is a satisfying assignment to φ, we see that

$$(x \wedge y) \vee (\neg x \wedge z) = f_\varphi(x, y, z, \vec{a}) \in [f_\varphi, 0, 1],$$

thus $f_\varphi \notin \mathrm{M}, \mathrm{A}$. Moreover,

$$f_\varphi(1, 1, 0, \vec{a}) = f_\varphi(0, 0, 1, \neg\vec{a}) = 1,$$
$$f_\varphi(1, 0, 1, \vec{a}) = f_\varphi(0, 1, 0, \neg\vec{a}) = 0,$$

thus $f_\varphi \notin \mathrm{D}, \mathrm{T}_1^2, \mathrm{T}_0^2$. □

Using a similar construction, we can now determine the complexity of CMP^f.

Theorem 3.7 *Let $f \in \mathrm{Op}$.*

(i) *If f is a projection (i.e., $f \in \bot$), then CMP^f is trivial.*

(ii) *Otherwise, CMP^f is $\mathbf{NP}$-complete.*

Proof: Let $\varphi \mapsto f_\varphi$ be the reduction from the proof of Theorem 3.6. It follows from the definition that $f_\varphi \in \mathrm{P}$, while (4) implies that $[f_\varphi] \supseteq \mathrm{P}$ if φ is satisfiable. Thus,

$$\begin{aligned} \varphi \in \textsc{Sat} &\implies [f_\varphi] = \mathrm{P}, \\ \varphi \in \textsc{UnSat} &\implies [f_\varphi] = \bot, \end{aligned}$$

hence $\varphi \mapsto \{f_\varphi\}$ is a reduction from $\textsc{Sat}$ to CMP^f whenever $f \in \mathrm{P} \smallsetminus \bot$. Likewise, the reduction $\varphi \mapsto \{f_\varphi, 0, 1\}$ works whenever $f \notin [0, 1]$, and $\varphi \mapsto \{f_\varphi, \neg\}$ works whenever $f \notin [\neg]$. This covers all cases where f is not a projection. □

Since CMP^f is a special case of CMP, Theorem 3.7 also shows the $\mathbf{NP}$-hardness of CMP, which (conditionally) refutes Vollmer's claim $\mathrm{CMP} \in \mathbf{coNP}$.

Corollary 3.8 $\mathrm{CMP} \notin \mathbf{coNP}$ *unless* $\mathbf{NP} = \mathbf{coNP}$. □

It will take more work to establish the true complexity of CMP, which we will do in the next subsection.

3.3 Hardness of CMP

We aim to prove that CMP is $\mathbf{\Theta}_2^{\mathbf{P}}$-complete in this subsection. First, if we look at how the $\mathbf{\Theta}_2^{\mathbf{P}}$ upper bound was derived in the proof of Theorem 3.4, we see that the only way the complexity of CMP can get as hard as $\mathbf{\Theta}_2^{\mathbf{P}}$ is by interaction of F and f deep inside one of the infinite arms of Post's lattice: otherwise (2) holds with a constant n, in which case the criterion can be evaluated in $\mathbf{P}^{\mathbf{NP}[O(1)]}$, i.e., in the Boolean hierarchy (cf. Lemma 4.1). Thus, we need a flexible supply of functions that generate clones on the infinite arms: we will find a suitable pool of such functions among the threshold functions θ_t^n. We then consider a suitable $\mathbf{\Theta}_2^{\mathbf{P}}$-hard problem based on counting the number of satisfiable formulas from a given list, which we transform into a question about the clone generated by a certain formula that combines the list using a threshold function.

In view of the outline above, our first task is to describe exactly the clones generated by threshold functions. For completeness, the lemma below is stated including various cases that we will not actually need.

Lemma 3.9 *Let $n \geq 1$ and $0 \leq t \leq n+1$. Then*

$$[\theta^n_t] = \begin{cases} \bot & t = n = 1, \\ \mathrm{UP}_1 & t = 0, \\ \bigvee\mathrm{P} & t = 1, \quad n > 1, \\ \mathrm{MPT}_1^{\lfloor (n-1)/(t-1) \rfloor} & 1 < t \leq \frac{n}{2}, \\ \mathrm{DM} & t = \frac{n+1}{2}, \quad n > 1, \\ \mathrm{MPT}_0^{\lceil t/(n-t) \rceil} & \frac{n}{2} + 1 \leq t < n, \\ \bigwedge\mathrm{P} & t = n, \quad n > 1, \\ \mathrm{UP}_0 & t = n+1. \end{cases}$$

Proof: The cases with $t \leq 1$ or $t \geq n$ are straightforward.

Notice that the dual of θ^n_t is θ^n_{n+1-t}. Thus, if $t = (n+1)/2$ (which implies n is odd), then $\theta^n_t \in \mathrm{DM}$. By Fig. 1, DM is a minimal clone, hence $[\theta^n_t] = \mathrm{DM}$ unless θ^n_t is a projection, which only happens when $n = 1$.

Assume that $1 < t \leq n/2$. Clearly, $\theta^n_t \in \mathrm{MP}$. Since $\mathbb{N}_{<n} = \{0, \dots, n-1\}$ has two disjoint subsets of size $\lfloor n/2 \rfloor \geq t$, $\theta^n_t \notin \mathrm{Pol}(r^2_0) = \mathrm{T}^2_0$. Also, $t \geq 2$ implies that θ^n_t is not bounded below by a variable, i.e., $\theta^n_t \notin \mathrm{T}^\infty_1$ by Lemma 3.1. By inspection of Post's lattice, it follows that $[\theta^n_t] = \mathrm{MPT}^k_1$ for some $k \geq 1$. Now, for any $k \geq 1$, we have $\theta^n_t \not\triangleright r^k_1$ iff $\mathbb{N}_{<n}$ can be covered by k subsets of size $< t$ iff $n \leq k(t-1)$, thus

$$\theta^n_t \in \mathrm{T}^k_1 \iff n \geq 1 + k(t-1) \iff k \leq \left\lfloor \frac{n-1}{t-1} \right\rfloor,$$

and consequently $[\theta^n_t] = \mathrm{MPT}_1^{\lfloor (n-1)/(t-1) \rfloor}$.

Finally, assume that $\frac{n}{2} + 1 \leq t < n$. The dual of θ^n_t is $\theta^n_{t'}$, where $t' = n + 1 - t$ satisfies $1 < t' \leq \frac{n}{2}$. Thus, by the case that we just proved, $[\theta^n_{t'}] = \mathrm{MPT}^k_1$ with

$$k = \left\lfloor \frac{n-1}{t'-1} \right\rfloor = \left\lfloor \frac{n-1}{n-t} \right\rfloor = \left\lfloor \frac{t + (n-t-1)}{n-t} \right\rfloor = \left\lceil \frac{t}{n-t} \right\rceil,$$

and $[\theta^n_t]$ is its dual, MPT^k_0. $\square$

We will also need a convenient $\mathbf{\Theta}^{\mathbf{P}}_2$-complete problem to reduce to CMP. In fact, the statement below effectively gives a $\mathbf{\Theta}^{\mathbf{P}}_2$-complete *promise problem* rather than a language. The idea was independently discovered by Wagner [31, Cor. 6.4] and Buss and Hay [10, Thm. 8], but the formulation below is closer to [20, L. 2.1]. We include a self-contained proof for completeness.

Lemma 3.10 *Let $L \subseteq \Sigma^*$ be any language such that $L \in \mathbf{\Theta}^{\mathbf{P}}_2$. Then there exists a logspace function $w \mapsto \langle \varphi_{w,i} : i < 2n_w \rangle$ (where each $\varphi_{w,i}$ is a CNF and $n_w \in \mathbb{N}$) with the following property: for every $w \in \Sigma^*$, there exists $0 < j \leq 2n_w$ such that for all $i < 2n_w$,*

$$\varphi_{w,i} \in \textsc{Sat} \iff i < j,$$

and we have

$$w \in L \iff j \text{ is even.}$$

Proof: L is computable by a polynomial-time Turing machine $M(w)$ that makes $|w|^c$ parallel (non-adaptive) queries to an **NP** oracle. Given a $w \in \Sigma^*$, put $n_w = |w|^c + 1$; for any $i < n_w$, let φ_{2i} be a CNF whose satisfiability is equivalent to the **NP** property "at least i queries made by $M(w)$ have positive answers", φ'_{2i+1} a CNF expressing "there is an accepting run of $M(w)$ with i positive answers to queries, all of which are correct", and φ_{2i+1} a CNF equivalent to $\varphi'_{2i+1} \vee \varphi_{2i+2}$ (where φ_{2n_w} is understood as $\bot$). If k is the true number of positively answered queries made by $M(w)$, then φ_i is satisfiable iff $i < 2k+1$ or $i < 2k+2$ depending on whether $w \in L$. $\square$

The following crucial lemma shows how to leverage threshold functions from Lemma 3.9 to translate the characterization of $\mathbf{\Theta}_2^{\mathbf{P}}$ from Lemma 3.10 into properties of clones. We recall that threshold functions have logspace-computable polynomial-size circuits, and even $O(\log n)$-depth formulas (see e.g. Wegener [32]).

Lemma 3.11 *There exists a logspace function $\langle \varphi_i : i < n \rangle \mapsto f_{\vec{\varphi}}$, and for each n, a sequence of integers $k_{n,0} > k_{n,1} > \cdots > k_{n,n} \geq 2$, with the following property: whenever $\langle \varphi_i : i < n \rangle$ is a sequence of formulas, we have $[\nrightarrow, f_{\vec{\varphi}}] = \mathrm{T}_0^{k_{n,s}}$, where $s = \bigl|\{i < n : \varphi_i \in \textsc{Sat}\}\bigr|$.*

Proof: For a given n, fix $m > n$ (to be specified later) and $t = m - n - 1$. We may assume that the formulas φ_i use pairwise disjoint sets of variables that are also disjoint from $\{x_i : i < m\}$. Put

$$f_{\vec{\varphi}} = \theta_t^m(x_0 \wedge \varphi_0, \dots, x_{n-1} \wedge \varphi_{n-1}, x_n, \dots, x_{m-1}).$$

When φ_i is unsatisfiable, we have $x_i \wedge \varphi_i \equiv 0$. Thus, renumbering w.l.o.g. the φ_is so that each φ_i, $i < s$, is satisfiable,

$$f_{\vec{\varphi}} \equiv \theta_t^{m-n+s}(x_0 \wedge \varphi_0, \dots, x_{s-1} \wedge \varphi_{s-1}, x_n, \dots, x_{m-1}).$$

On the one hand, $x_i \wedge \varphi_i \in \mathrm{T}_0^\infty = [\nrightarrow]$ by Lemma 3.1, thus $f_{\vec{\varphi}} \in [\nrightarrow, \theta_t^{m-n+s}]$. On the other hand, for each $i < s$, we may choose a satisfying assignment a_i to φ_i, substitute $0 \in [\nrightarrow]$ for each variable made 0 by a_i, and substitute x_i for each variable made 1 by a_i. (By our assumptions on variables, we can do this independently for each $i < s$, and it will not affect the $\vec{x}$ variables.) Under this substitution, $x_i \wedge \varphi_i$ becomes equivalent to x_i, and $f_{\vec{\varphi}}$ becomes $\theta_t^{m-n+s}(x_0, \dots, x_{s-1}, x_n, \dots, x_{m-1})$. Thus,

$$[\nrightarrow, f_{\vec{\varphi}}] = [\nrightarrow, \theta_t^{m-n+s}] = \mathrm{T}_0^{k_{n,s}}$$

using Lemma 3.9, where

$$k_{n,s} = \left\lceil \frac{t}{m-n-t+s} \right\rceil = \left\lceil \frac{t}{s+1} \right\rceil,$$

as long as $m/2+1 \leq t$, i.e., $m \geq 2n+4$. In order to satisfy the constraint $k_{n,0} > k_{n,1} > \cdots > k_{n,n}$, it suffices to ensure that

$$\frac{t}{s+1} + 1 \leq \frac{t}{s}$$

for all $s \leq n$, i.e., $t \geq n(n+1)$. This holds if we choose $m = \max\{(n+1)^2, 6\}$. $\square$

We are ready to prove the main result of this section.

Theorem 3.12 CMP *is* $\mathbf{\Theta}_2^{\mathbf{P}}$*-complete.*

Proof: We have CMP $\in \mathbf{\Theta}_2^{\mathbf{P}}$ by Theorem 3.4. In order to show that it is $\mathbf{\Theta}_2^{\mathbf{P}}$-hard, fix $L \in \mathbf{\Theta}_2^{\mathbf{P}}$.

Given w, compute $\langle \varphi_i : i < 2n_w \rangle$ as in Lemma 3.10, and then (abbreviating $n = n_w$) $f_{\mathrm{even}} = f_{\varphi_0, \varphi_2, \dots, \varphi_{2n-2}}$, $f_{\mathrm{odd}} = f_{\varphi_1, \varphi_3, \dots \varphi_{2n-1}}$ as in Lemma 3.11. If $j \le 2n$ is as in Lemma 3.10, then

$$\begin{aligned} \bigl|\{i < n : \varphi_{2i} \in \textsc{Sat}\}\bigr| &= \lceil j/2 \rceil, \\ \bigl|\{i < n : \varphi_{2i+1} \in \textsc{Sat}\}\bigr| &= \lfloor j/2 \rfloor, \end{aligned}$$

thus

$$\begin{aligned} [\nrightarrow, f_{\mathrm{even}}] &= \mathrm{T}_0^{k_{n,\lceil j/2 \rceil}}, \\ [\nrightarrow, f_{\mathrm{odd}}] &= \mathrm{T}_0^{k_{n,\lfloor j/2 \rfloor}}, \end{aligned}$$

where $k_{n,0} > \cdots > k_{n,n} \ge 2$ are as in Lemma 3.11. It follows that

$$f_{\mathrm{even}} \in [\nrightarrow, f_{\mathrm{odd}}] \iff \lfloor j/2 \rfloor = \lceil j/2 \rceil \iff j \text{ is even} \iff w \in L.$$

Thus, $w \mapsto \langle \{\nrightarrow, f_{\mathrm{odd}}\}, f_{\mathrm{even}} \rangle$ is a reduction from L to CMP. □

3.4 Alternative setups

We followed the tradition in the study of Boolean clones—going back to Post—of considering only functions of positive arity, even though the general theory of clones and coclones works more smoothly if nullary functions are also allowed (cf. Behrisch [3]). Let us see now what changes if we take nullary functions into consideration.

First, the number of Boolean clones increases—namely, each non-nullary clone C that includes at least one constant function (i.e., $C \supseteq \mathrm{UP}_0$ or UP_1) splits into two: one consisting only of non-nullary functions as before, and one that also includes nullary functions corresponding to all constant functions of C. In Fact 2.2, we understand the given definitions of meet-irreducible clones so that they include all applicable nullary functions; moreover, there is a new meet-irreducible clone $\mathrm{N} = [\wedge, \neg] = \mathrm{Pol}(r_N)$ of all non-nullary functions, where $r_N = \varnothing \in \mathrm{Rel}_1$. Consequently, we include r_N in R_n for each n. Note that $\mathrm{D} \subseteq \mathrm{N}$ and $\mathrm{P} \subseteq \mathrm{N}$.

Since the set $\{\wedge, \vee, \neg\}$ is no longer functionally complete, we read the definition of CMP and derived problems so that the input is given in the form of circuits over the basis $\{\wedge, \vee, \neg, 0, 1\}$, where 0 and 1 denote nullary constants.

The upper bounds in Theorem 3.4 continue to hold unchanged.

In Theorem 3.6, the main dichotomy still holds: $\mathrm{CMP}_C \in \mathbf{P}$ if $C \supseteq \mathrm{P}$, and CMP_C is **coNP**-complete otherwise. The difference is that now there are more clones $C \supseteq \mathrm{P}$, namely $\top$, N, P_0, NP_0, P_1, NP_1, and P. $\mathrm{CMP}_\top$ is trivial, and CMP_C is **P**-complete for $C = \mathrm{P}_\alpha, \mathrm{NP}_\alpha, \mathrm{P}$. The problem CMP_N amounts to testing if the given Boolean circuit has a nonzero number of input variables: the exact mechanics of this test will depend on syntactic details of the representation of input, but it can be done in $\mathbf{AC}^0$ under any reasonable representation.

The statement of Theorem 3.7 changes so that CMP^f is **NP**-complete if f is neither a projection nor a nullary function. If f is a nullary function, then CMP^f is **P**-complete: if $\alpha \in \mathbf{2}$ is a nullary constant, we have that $\alpha \in [F]$ iff either $\alpha \in F$, or F contains the dual constant $\neg\alpha$ and $F \nsubseteq \mathrm{P}_{\neg\alpha}$ (the second disjunct is only possible if $|F| \geq 2$, thus given a variable-free circuit C, we have $C = \alpha$ iff $\{C\} \in \mathrm{CMP}^\alpha$, showing the **P**-hardness of CMP^f).

All named clones in Lemmas 3.9 and 3.11 need to be intersected with N, so that, e.g., the conclusion of Lemma 3.11 reads $[\nrightarrow, f_{\vec{\varphi}}] = \mathrm{NT}_0^{k_{n,s}}$.

The main Theorem 3.12 still holds.

A different kind of variation of our results concerns the input representation. We defined CMP and friends so that the input functions are represented by Boolean circuits, which is the natural thing to do in a computational context. However, in the context of logic or algebra, it is more natural to represent Boolean functions by Boolean formulas, or equivalently, Boolean terms.

Fortunately, this has a negligible effect on our results. First, all upper bounds hold also for the formula representation, because formulas are special cases of circuits. On the other hand, our main lower bounds continue to hold in this setting as well: we used reductions from Boolean satisfiability (that already works with formulas), and the most complicated tools we employed were threshold functions, which can be written with polynomial-size formulas just as well as circuits.

The only exceptions are problems that were proved **P**-complete by reductions from evaluation of Boolean circuits: namely, CMP_C for $\mathrm{P} \subseteq C \neq \top, \mathrm{N}$ (Theorem 3.6), and CMP^f for f a nullary function (Theorem 3.7 as modified in the first part of this subsection). If we change the input representation to formulas, then all these problems are computable in $\mathbf{NC}^1$.

4 Restricted input bases

The problems CMP, CMP_C, and CMP^f are defined so that the input functions are given by circuits over a fixed functionally complete basis. This is reasonable if we consider these circuits to be just a computing device. However, if we view the problem as "given a circuit over the De Morgan basis, can we rewrite it as a circuit over a given basis F?", it makes perfect sense to also consider the case where the input circuits are expressed in a different basis. That is, for any finite $B \subseteq \mathrm{Op}$, let B-CMP be the following problem:

> **Input:** A finite set of functions $F \subseteq \mathrm{Op}$ and a function $f \in \mathrm{Op}$, all given by circuits over the basis B.
>
> **Output:** YES if $f \in [F]$, otherwise NO.

Similarly, we define the problems B-CMP_C and B-CMP^f.

The complexity of B-CMP_C was thoroughly investigated by Böhler and Schnoor [7], who denote the problem as $\mathcal{M}_C(C \twoheadleftarrow B)$, and its formula version as $\mathcal{M}(C \twoheadleftarrow B)$. For most combinations of C and B, they were able to show that $\mathcal{M}(C \twoheadleftarrow B)$ and $\mathcal{M}_C(C \twoheadleftarrow B)$ are both **coNP**-complete, or both in **P**.

We will not make any effort to classify the complexity of the problems B-CMP^f, as the number of cases is prohibitively large, hence we consider it out of scope of this paper. (The problem has two clone parameters, analogously to B-CMP$_C$, which took Böhler and Schnoor a whole paper to understand, and even then their classification is incomplete.)

We will not obtain a complete classification of the complexity of B-CMP either, nevertheless we present a number of partial results, summarized in Corollary 4.14 at the end of the section.

Unless stated otherwise, the complexity results below also hold for variants of B-CMP where the input functions are represented by formulas (in prefix or infix notation). For simplicity, we disallow nullary functions in this section, but the results below can be easily adapted to a setup that includes them. Our results will be stated for arbitrary bases, but we will often need to express specific functions: to this end, we recall that we can more-or-less freely convert circuits and formulas to different bases by Lemma 2.3. In particular, Lemma 2.3 (ii) applies to polynomial-size CNFs and other constant-depth formulas using unbounded fan-in $\wedge$, $\vee$ gates, as these can be written as $O(\log n)$-depth bounded fan-in formulas.

For the rest of this section, B is a finite subset of Op.

We start with a simple observation.

Lemma 4.1 *If* $B \subseteq \mathrm{T}_0^\infty$, T_1^∞, $\bigwedge$, $\bigvee$, A, *or* D, *then* B-CMP $\in$ **BH**.

Proof: By inspection of Post's lattice, we see that there are only finitely many clones below $[B]$. Thus, if F and f are given by B-circuits, we have

$$f \in [F] \iff \forall r \in R_k \,(F \not\triangleright r \text{ or } f \triangleright r)$$

for some constant k, which gives a **BH** algorithm in view of Lemma 3.3. □

This suggests that there is a major difference between the complexity of B-CMP in cases where $[B]$ has finitely many subclones and in the other cases. We will study the former in Section 4.1, and the latter in Section 4.2.

4.1 The finite subclone cases

We are going to characterize more precisely the complexity of B-CMP for B as in Lemma 4.1, i.e., such that $[B]$ has finitely many subclones. Depending on $[B]$, we will classify it as being either **coDP**-complete or in **P** (with the formula version in $\mathbf{NC}^1$). We are first going to deal with the **P** cases, then show **coDP**-hardness of the remaining cases using reductions from variants of the equivalence problem, and finish by establishing they are indeed in **coDP**.

We start with the tractable cases. The theorem below mostly follows from results of Böhler and Schnoor [7]: since $[B]$ has only finitely many subclones, we have B-CMP $\in$ **P** iff B-CMP$_C \in$ **P** for all clones $C \subseteq [B]$. However, we give a self-contained proof, which also confirms that the complexity drops down to $\mathbf{NC}^1$ for formulas; the basic idea is that the clones in question have so simple structure that we can determine $[f]$ by evaluating f on a specific polynomial-size set of assignments.

Theorem 4.2 *If $B \subseteq \mathrm{MT}_0^\infty$, MT_1^∞, $\bigwedge$, $\bigvee$, A, or DM, then B-CMP $\in \mathbf{P}$. If we represent the input by formulas rather than circuits, B-CMP $\in \mathbf{NC}^1$.*

Proof: Assume that $B \subseteq \bigvee$. The set of n-ary functions in $\bigvee$ is very limited: it consists only of 1 and the functions $f_I(x_0, \dots, x_{n-1}) = \bigvee_{i \in I} x_i$ for $I \subseteq \{0, \dots, n-1\}$ (including $f_\varnothing(\vec{x}) = 0$). Given a B-circuit, we can determine which of these functions it computes by evaluating it on the assignment $\vec{0}$, which detects the 1 function, and for each $i < n$, on $e_i = \langle 0, \dots, 0, 1, 0, \dots, 0 \rangle$ (with 1 at position i), which detects if $i \in I$.

Once we know the function, it is trivial to determine which of the finitely many subclones of $\bigvee$ it generates, and if we know the clones generated by f and by each element of F, we can find out if $f \in [F]$. This gives a polynomial-time algorithm. Moreover, since the algorithm just evaluates the input functions on polynomially many assignments in parallel, and then does $\mathbf{AC}^0$ post-processing, it can be implemented in $\mathbf{NC}^1$ if the functions are given by formulas rather than circuits.

The cases of $B \subseteq \bigwedge$ or $B \subseteq \mathrm{A}$ are completely analogous to $\bigvee$. In particular, the n-ary functions in A are $f_{\vec{\alpha},\alpha}(\vec{x}) = \sum_{i<n} \alpha_i x_i + \beta$ for some $\alpha_0, \dots, \alpha_{n-1}, \beta \in \mathbf{2}$; given a B-circuit computing a function f, we can determine $\vec{\alpha}$ and β such that $f = f_{\vec{\alpha},\beta}$ by evaluating $\beta = f(\vec{0})$ and $\alpha_i = f(e_i) + \beta$.

Assume that $[B] = \mathrm{DM}$. Since DM is a minimal clone, a B-circuit either computes a projection, or it generates DM. Moreover, a self-dual monotone function $f \in \mathrm{Op}_n$ is the projection π_i^n iff $f(e_i) = 1$. Thus, we can again determine $[f]$ by evaluating f at n assignments.

Assume that $B \subseteq \mathrm{MT}_1^\infty$, the case of MT_0^∞ being dual. Given a B-circuit computing a function f, either $f \in \bigvee$, or $[f] = \mathrm{MPT}_1^\infty$, MT_1^∞. By evaluating $f(e_i)$ for all $i < n$, we find the only candidate $I \subseteq \{0, \dots, n-1\}$ such that f could equal f_I. Then we evaluate f at the assignment a such that $a_i = 1$ iff $i \notin I$: if $f(a) = 0$, then $f \equiv f_I$, otherwise $f \notin \bigvee$. In the latter case, we can distinguish MT_1^∞ from MPT_1^∞ by evaluating $f(\vec{0})$. □

We remark that while B-CMP is $\mathbf{P}$-complete (or $\mathbf{NC}^1$-complete in the formula representation) for $[B] = \mathrm{M(P)T}_\alpha^\infty$ or $[B] = \mathrm{DM}$, it is still easier for other classes from Theorem 4.2: for example, it is easy to show that if $[B] \subseteq \bigvee$, we can evaluate B-circuits in $\mathbf{L}$, and B-formulas in $\mathbf{AC}^0$. We will not go into details.

We now turn to the remaining clones below T_α^∞ and D. We will use the following result for lower bounds. Here, the *equivalence problem* B-EQ is to compute if two given Boolean formulas over basis B are equivalent.

Theorem 4.3 (Reith [24]) *If $[B] \supseteq \mathrm{MPT}_0^\infty$, MPT_1^∞, or DM, then B-EQ is $\mathbf{coNP}$-complete.* □

We mention that since the formulas used in Theorem 4.3 are built from DNFs, they can be taken to have depth $O(\log n)$. In particular, this ensures they can be efficiently converted to a different basis by Lemma 2.3.

The constructions in the next lemma mostly come from Böhler and Schnoor [7], who used them to prove $\mathbf{coNP}$-completeness of various instances of B-CMP$_C$. We observe that the lower bound can be improved to $\mathbf{coDP}$ if C is allowed to vary.

Lemma 4.4 *If* $[B] \supseteq \mathrm{PT}_0^\infty$, PT_1^∞, MPT_0^2, MPT_1^2, *or* DP, *then* B-CMP *is* **coDP**-*hard.*

Proof: Assume first $[B] \supseteq \mathrm{DP}$. If f, g are DM-formulas (i.e., formulas over a fixed basis of DM), let $h_{f,g}(\vec{x}, y_0, y_1, y_2) = f(\vec{x}) + g(\vec{x}) + \theta_2^3(\vec{y})$; this can be expressed by a DP-formula as the ternary function $x + y + z$ is in DP. In view of $f, g \in \mathrm{P}$, we have $\theta_2^3(\vec{y}) = h_{f,g}(x, \ldots, x, \vec{y}) \in [h_{f,g}]$, thus $\mathrm{DM} \subseteq [h_{f,g}]$. If $f \equiv g$, then $h_{f,g}(\vec{x}, \vec{y}) \equiv \theta_2^3(\vec{y})$. If $f \not\equiv g$, let $\vec{a}$ be an assignment such that $f(\vec{a}) \neq g(\vec{a})$. Then $h_{f,g}(\vec{0}, \vec{1}) = 1$ (using $f, g \in \mathrm{P}$) and $h_{f,g}(\vec{a}, \vec{1}) = 0$, hence $h_{f,g}$ is not monotone. Since $\mathrm{DM} \subseteq [h_{f,g}] \subseteq \mathrm{DP}$, this implies $[h_{f,g}] = \mathrm{DP}$. Thus,

$$f \equiv g \implies [h_{f,g}] = \mathrm{DM}, \tag{5}$$

$$f \not\equiv g \implies [h_{f,g}] = \mathrm{DP}. \tag{6}$$

Now, since DM-EQ is **coNP**-complete by Theorem 4.3, the language

$$L = \bigl\{\langle f, g, f', g'\rangle : \langle f, g\rangle \in \text{DM-EQ or } \langle f', g'\rangle \notin \text{DM-EQ}\bigr\}$$

is **coDP**-complete. Using (5) and (6),

$$\langle f, g, f', g'\rangle \in L \iff [h_{f,g}] \subseteq [h_{f',g'}],$$

which gives a reduction of L to B-CMP.

If $[B] \supseteq \mathrm{MPT}_1^2$ (the case of MPT_0^2 is dual), we can use in a similar way the MPT_1^2-formula

$$h(\vec{x}, y, z) = \theta_2^3\bigl(f(\vec{x}) \vee g(\vec{x}), y, z\bigr).$$

The dual of h is $\theta_2^3\bigl(f(\vec{x}) \wedge g(\vec{x}), y, z\bigr)$, hence h is self-dual if and only if $f \equiv g$. Moreover, $f(x, \ldots, x) \equiv g(x, \ldots, x) \equiv x$, hence in any case $\theta_2^3 \in [h]$. Thus,

$$f \equiv g \implies [h] = \mathrm{DM},$$
$$f \not\equiv g \implies [h] = \mathrm{MPT}_1^2.$$

This gives a reduction of L to B-CMP in the same way as above.

Finally, assume $[B] \supseteq \mathrm{PT}_1^\infty$ (the case of PT_0^∞ is dual). Given MPT_1^∞-formulas f and g, we put $h(\vec{x}, y) = y \vee \bigl(f(\vec{x}) + g(\vec{x})\bigr)$, which can be expressed by a PT_1^∞-formula. If $f \equiv g$, then $h \equiv y$; otherwise, we check easily that h is not monotone. Thus,

$$f \equiv g \implies [h] = \bot,$$
$$f \not\equiv g \implies [h] = \mathrm{PT}_1^\infty,$$

which yields a reduction of the **coDP**-complete problem

$$\bigl\{\langle f, g, f', g'\rangle : \langle f, g\rangle \in \mathrm{MPT}_1^\infty\text{-EQ or } \langle f', g'\rangle \notin \mathrm{MPT}_1^\infty\text{-EQ}\bigr\}$$

to B-CMP. $\square$

Incidentally, the argument we gave for MPT^2_α also resolves one of the problems left open by Böhler and Schnoor [7]:

Corollary 4.5 *If* $[B] \supseteq \mathrm{MPT}^2_0$ *or* MPT^2_1*, then* B-$\mathrm{CMP}_{\mathrm{DM}}$ *is* **coNP***-complete.* $\square$

It remains to prove **coDP** upper bounds for the B-CMP problems in question. We obtain this by optimizing the number of **NP** oracle calls in the algorithm from Lemma 4.1, exploiting the fact that we can distinguish between most proper subclones of $[B]$ in polynomial time by Theorem 4.2.

Theorem 4.6 *If* $[B]$ *is* T^∞_0*,* PT^∞_0*,* T^∞_1*,* PT^∞_1*,* D*, or* DP*, then* B-CMP *is* **coDP***-complete.*

Proof: **coDP**-hardness was proved in Lemma 4.4. Assume that $B \subseteq \mathrm{T}^\infty_1$, we will refine Lemma 4.1 to show that B-CMP $\in$ **coDP**.

First, given a set of B-circuits F, we can determine $[F]$ in polynomial time using a single **coNP** oracle query, namely $F \overset{?}{\subseteq} \mathrm{M}$: indeed, if $F \nsubseteq \mathrm{M}$, then $[F]$ is PT^∞_1 or T^∞_1, and we can distinguish these two cases by testing if $F \subseteq \mathrm{P}$ (or equivalently, P_0), which we can do by evaluating $g(\vec{0})$ for each $g \in F$. On the other hand, if $F \subseteq \mathrm{M}$, i.e., $F \subseteq \mathrm{MT}^\infty_1$, we can compute $[F]$ using the algorithm in Theorem 4.2 (which again proceeds by evaluation of F at various assignments, hence it does not matter that the circuits are given in a larger basis).

This already shows that B-CMP $\in \mathbf{P}^{\|\mathbf{NP}[2]}$: we can test if $f \in [F]$ using two parallel **coNP** queries, $F \overset{?}{\subseteq} \mathrm{M}$ and $f \overset{?}{\in} \mathrm{M}$.

In order to improve this to **coDP**, we modify the algorithm so that it speculatively explores all computation branches with all possible oracle answers, and only makes the oracle queries needed at the end.

In this way, the algorithm computes two candidate clones $C_0 \subseteq \mathrm{MT}^\infty_1$ and $C_1 \supseteq \mathrm{PT}^\infty_1$ for $[F]$. We have $C_0 \subseteq C_1$: the choice of C_1 as PT^∞_1 or T^∞_1 is made according to if $F \subseteq \mathrm{P}$, and this information is taken into account also when computing C_0. In fact, this means that C_1 is the join $C_0 \vee \mathrm{PT}^\infty_1$ in the lattice of clones.

(Actually, the algorithm may fail to compute C_0 because it runs into an inconsistency that already shows $F \nsubseteq \mathrm{M}$. In this case, we may pick C_0 in an arbitrary way such that the properties $C_0 \subseteq \mathrm{M}$ and $C_1 = C_0 \vee \mathrm{PT}^\infty_1$ hold, say, $C_0 = \bot$ or $C_0 = \mathrm{UP}_1$: if the choice of C_0 became relevant in subsequent computation, it would be dismissed by the oracle as $F \nsubseteq \mathrm{M}$.)

Likewise, we obtain two candidate clones C'_0, C'_1 for $[f]$, with $C'_0 \subseteq \mathrm{MT}^\infty_1$ and $C'_1 = C'_0 \vee \mathrm{PT}^\infty_1$.

Notice that $C'_1 \nsubseteq C_0$ as $C_0 \subseteq \mathrm{M}$, and that $C'_0 \subseteq C_1$ implies $C'_1 = C'_0 \vee \mathrm{PT}^\infty_1 \subseteq C_1$. Thus, there are only the following possibilities:

- $C'_0 \subseteq C_0$ (whence $C'_1 \subseteq C_1$): then $f \in [F]$ if and only if $F \nsubseteq \mathrm{M}$ or $f \in \mathrm{M}$.
- $C'_1 \subseteq C_1$ and $C'_0 \nsubseteq C_0$: then $f \in [F]$ if and only if $F \nsubseteq \mathrm{M}$.
- $C'_0 \nsubseteq C_1$: then $f \notin [F]$.

Consequently, the whole algorithm can be implemented in **coDP**: we have

$$f \in [F] \iff \langle F, f\rangle \in L_0 \text{ or } \langle F, f\rangle \in L_1$$

with

$$L_0 = \{\langle F, f\rangle : C_1' \subseteq C_1 \text{ and } F \nsubseteq \mathrm{M}\} \in \mathbf{NP},$$
$$L_1 = \{\langle F, f\rangle : C_0' \subseteq C_0 \text{ and } f \in \mathrm{M}\} \in \mathbf{coNP},$$

where C_0, C_1, C_0', C_1' are computed from $\langle F, f\rangle$ in deterministic polynomial time as described above.

The case of $B \subseteq \mathrm{T}_0^\infty$ is dual.

The argument for $B \subseteq \mathrm{D}$ uses the same overall strategy: given F, we compute in polynomial time two candidates C_0 and $C_1 = C_0 \vee \mathrm{DP}$ for $[F]$, where $C_0 \subseteq \mathrm{DM}$ or $C_0 \subseteq \mathrm{AD}$. Thus, $[F]$ is C_0 if $F \subseteq \mathrm{M}$ or $F \subseteq \mathrm{A}$, and it is C_1 otherwise. Likewise, given f, we compute candidates C_0' and $C_1' = C_0' \vee \mathrm{DP}$ for $[f]$ satisfying $C_0' \subseteq \mathrm{DM}$ or $C_0' \subseteq \mathrm{AD}$. Then as above, $f \in [F]$ iff $\langle F, f\rangle \in L_0$ or $\langle F, f\rangle \in L_1$, with

$$L_0 = \{\langle F, f\rangle : C_1' \subseteq C_1 \text{ and } F \nsubseteq \mathrm{M} \text{ and } F \nsubseteq \mathrm{A}\} \in \mathbf{NP},$$
$$L_1 = \{\langle F, f\rangle : C_0' \subseteq C_0 \text{ and } (f \in \mathrm{M} \text{ or } f \in \mathrm{A})\} \in \mathbf{coNP}.$$

The candidates are determined as follows. If $g(\vec{0}) = 1$ for some $g \in F$, we put $C_1 = \mathrm{D}$, and C_0 is AD or UD depending on whether there exists $g \in F$ such that $|\{i : g(e_i) \neq g(\vec{0})\}| > 1$.

If $g(\vec{0}) = 0$ for all $g \in F$, then $C_1 = \mathrm{DP}$. Putting $m(g) = |\{i : g(e_i) = 1\}|$, the argument in the proof of Theorem 4.2 shows that if $F \subseteq \mathrm{DM}$, then $m(g) \in \{0, 1\}$ for all $g \in F$, whereas if $F \subseteq \mathrm{AP}$, then $m(g)$ is odd for all $g \in F$; in both cases, $m(g) = 1$ iff g is a projection. Thus, we may define $C_0 = \mathrm{DM}$ if $m(g) = 0$ for some $g \in F$, $C_0 = \mathrm{AP}$ if $m(g) > 1$ for some $g \in F$, and $C_0 = \bot$ otherwise. □

4.2 The hard cases

The question we are mainly interested in is for which bases B is B-CMP a $\mathbf{\Theta}_2^{\mathbf{P}}$-complete problem. We first adapt Theorem 3.12 easily to P-CMP using simple transformations to eliminate constants. Next, we generalize it further to the clones PT_α^k by expressing the threshold functions used in the proof of Lemma 3.11 in a $\{\theta_k^{k+1}\}$ basis; we only manage to do so with a randomized construction, leading to a $\mathbf{\Theta}_2^{\mathbf{P}}$-completeness result under a suitable notion of randomized reductions.

In order to get the result for P-CMP, we use the following translation (for completeness, we formulate it more generally than what we need):

Lemma 4.7 *Let $B, B' \subseteq \mathrm{Op}$ be finite. Assume that $\wedge \in [B']$ and $B \subseteq [B', 0]$, or $\vee \in [B']$ and $B \subseteq [B', 1]$, or $\wedge, \vee \in [B']$ and $B \subseteq [B', 0, 1]$.*

(i) *Given a B-circuit that computes a B'-function f, we can construct in logspace a B'-circuit that computes f.*

(ii) *Given a B-formula of depth $O(\log n)$ that computes a B'-function f, we can construct in logspace a B'-formula that computes f.*

In particular, this applies to arbitrary B if $[B'] \supseteq \mathrm{P}$.

Proof: Assume first $\wedge \in [B']$ and $B \subseteq [B', 0]$. Without loss of generality, $0 \notin [B']$, hence $B' \subseteq \mathrm{P}_1$.

Since $B \subseteq [B', 0]$, Lemma 2.3 implies that we can find a B'-circuit g (or even an $O(\log n)$-depth B'-formula) such that $f(\vec{x}) \equiv g(\vec{x}, 0)$. Then $f(\vec{x}) \equiv g\bigl(\vec{x}, \bigwedge_i x_i\bigr)$: the two expressions agree when $\vec{x} \neq \vec{1}$ as $\bigwedge_i x_i = 0$; they also agree for $\vec{x} = \vec{1}$ as $f(\vec{1}) = 1 = g(\vec{1}, 1)$ on account of $f, g \in \mathrm{P}_1$. Since $\bigwedge_i x_i$ has an $O(\log n)$-depth formula over the basis $\{\wedge\} \subseteq [B']$, it can be written by an $O(\log n)$-depth B'-formula using Lemma 2.3 again.

The case $\vee \in [B']$ and $B \subseteq [B', 1]$ is dual.

Let $\wedge, \vee \in [B']$ and $B \subseteq [B', 0, 1]$. If $0 \in [B']$ or $1 \in [B']$, we are done by one of the previous cases, hence we may assume $0, 1 \notin [B']$, which implies $B' \subseteq \mathrm{P}$. Then we proceed as before: we find a B'-circuit g such that $f(\vec{x}) \equiv g(\vec{x}, 0, 1)$, and we observe that $f(\vec{x}) \equiv g\bigl(\vec{x}, \bigwedge_i x_i, \bigvee_i x_i\bigr)$ because of $f, g \in \mathrm{P}$. □

Theorem 4.8 *If $[B] \supseteq \mathrm{P}$, then B-CMP is $\mathbf{\Theta}_2^{\mathbf{P}}$-complete.*

Proof: The formulas we constructed in the proof of Theorem 3.12 (or rather, Lemma 3.11) compute functions in P_0, and have depth $O(\log n)$ (being build from CNFs and threshold functions), hence we can convert them to P_0-formulas by Lemma 4.7. Thus, in the setting of Lemma 3.11, we map a sequence of formulas $\langle \varphi_i : i < n \rangle$ to a P_0-formula $f_{\vec{\varphi}}(\vec{x})$ such that $[\nrightarrow, f_{\vec{\varphi}}] = \mathrm{T}_0^k$, where $k \geq 2$ is as specified in the lemma.

Using the idea of Lemma 4.7, we can construct a P-formula $g_{\vec{\varphi}}(\vec{x}, y)$ (and therefore a B-formula, using Lemma 2.3) such that $f_{\vec{\varphi}}(\vec{x}) \equiv g_{\vec{\varphi}}(\vec{x}, 0)$. By replacing $g_{\vec{\varphi}}$ with $g_{\vec{\varphi}}\bigl(\vec{x}, y \wedge \bigwedge_i x_i\bigr)$ if necessary, we ensure that

$$g_{\vec{\varphi}}(\vec{x}, y) \equiv f_{\vec{\varphi}}(\vec{x}) \vee \Bigl(y \wedge \bigwedge_i x_i\Bigr). \tag{7}$$

We claim that

$$[x \wedge (y \to z), g_{\vec{\varphi}}] = \mathrm{PT}_0^k, \tag{8}$$

hence we can use $[x \wedge (y \to z), g_{\vec{\varphi}}]$ in place of $[\nrightarrow, f_{\vec{\varphi}}]$ in the proof of Theorem 3.12 to get a reduction from any $\mathbf{\Theta}_2^{\mathbf{P}}$ language to B-CMP.

In order to prove (8), observe that $[x \wedge (y \to z)] = \mathrm{PT}_0^\infty$: clearly $x \wedge (y \to z) \in \mathrm{P}$, and $x \wedge (y \to z) \in \mathrm{T}_0^\infty$ by Lemma 3.1, while $x \wedge (y \to z) \notin \mathrm{M}$. Thus, $\mathrm{PT}_0^\infty \subseteq [x \wedge (y \to z), g_{\vec{\varphi}}] \subseteq \mathrm{P}$ and $[x \wedge (y \to z), g_{\vec{\varphi}}, 0] \supseteq [\nrightarrow, f_{\vec{\varphi}}] = \mathrm{T}_0^k$, which implies

$$[x \wedge (y \to z), g_{\vec{\varphi}}] = \mathrm{PT}_0^l$$

for some $1 \leq l \leq k$. It now suffices to show that $g_{\vec{\varphi}} \in \mathrm{T}_0^k = \mathrm{Pol}(r_0^k)$.

Assume for contradiction that there are assignments $\vec{a}^0, \dots, \vec{a}^{k-1}$ such that $\vec{a}^0 \wedge \cdots \wedge \vec{a}^{k-1} = \vec{0}$, but $g_{\vec{\varphi}}(\vec{a}^0) = \cdots = g_{\vec{\varphi}}(\vec{a}^{k-1}) = 1$. Notice that if $\vec{a}^i = \vec{1}$ for some i, we may leave it out (or rather, replace it with another assignment from $g_{\vec{\varphi}}^{-1}[1]$), as still $\bigwedge_{j \neq i} \vec{a}^j = \vec{0}$. Thus, we may assume that none of the $\vec{a}^i$ is $\vec{1}$. But then $f_{\vec{\varphi}}(\vec{a}^i) = g_{\vec{\varphi}}(\vec{a}^i) = 1$ for each $i < k$ by (7) (more precisely, this holds for the truncation of $\vec{a}^i$ leaving out the last coordinate). This contradicts $f_{\vec{\varphi}} \in \mathrm{T}_0^k$. □

Now we would like to extend Theorem 4.8 to the clones PT^k_α, and for that we need efficient constructions of threshold functions in a PT^k_α basis. This in fact appears to be quite a challenging task. The best deterministic result we found is a construction by Cohen et al. [11], whose special case for the $\{\theta^{k+1}_2\}$ basis is as follows:

Theorem 4.9 (Cohen et al. [11]) *Let $k \geq 2$. There exists a constant c and a polynomial-time algorithm that, given a sufficiently large N, constructs in time $N^{O(1)}$ an $O(\log N)$-depth $\{\theta^{k+1}_2\}$-formula $\psi_N(x_0, \dots, x_{N-1})$ such that*

$$\theta^N_{\lfloor N(k^{-1}+\varepsilon)\rfloor} \leq \psi_N \leq \theta^N_{\lceil N(k^{-1}-\varepsilon)\rceil},$$

where $\varepsilon = c/\sqrt{\log N}$. □

Since the formula ψ_N can only reliably distinguish inputs whose Hamming weights differ by $\Omega(N/\sqrt{\log N})$, it cannot be used to tell apart more than $O(\sqrt{\log N})$ different cases. Thus, in order to distinguish n possible outcomes as in Lemma 3.11, we would need $N = \exp(\Omega(n^2))$, which makes it useless for our purposes. For the special case $k = 2$, they give a better construction that reduces the error of approximation to $2^{-O(\sqrt{\log N})}$, which means we might get away with $N = n^{O(\log n)}$, but this is still insufficient.

In absence of a better idea, we resort to probabilistic constructions following the method of Valiant [29], who used it to prove the existence of short $\{\wedge, \vee\}$-formulas for majority; Gupta and Mahajan [13] modified his construction to produce $\{\theta^3_2\}$-formulas. We will use a similar idea to express suitable threshold functions by short formulas over the $\{\theta^{k+1}_k\}$ basis.

Theorem 4.10 *Let $k \geq 3$. There exist constants $\frac{1}{2} < \sigma_k < 1$ and $c \geq 1$, and a logspace function T with the following properties. The input of T consists of numbers n, t, and e in unary, and $r \in \{0,1\}^*$; the output is a $\{\theta^{k+1}_k\}$-formula $T_{n,t,e,r}(x_0, \dots, x_{n-1})$ of depth $O(\log n + \log e)$. If n is sufficiently large and $\sigma_k n < t \leq n$, then*

$$\Pr_{|r|=(n+e)^c}[T_{n,t,e,r} \equiv \theta^n_t] \geq 1 - 2^{-e}.$$

Proof: Given n and d, let $F_{n,d}$ be a random formula consisting of a complete $(k+1)$-ary tree of θ^{k+1}_k-gates of depth d, where each leaf is a propositional variable independently uniformly chosen from $\{x_i : i < n\}$. If $a \in \mathbf{2}^n$ is an assignment of weight $w = pn$, $p \in [0, 1]$, then $F_{n,d}(a)$ is a Bernoulli random variable that takes value 1 with certain probability p_d. We can describe $F_{n,d}(a)$ as the value of a complete $(k+1)$-ary tree of depth d of θ^{k+1}_k-gates, where each leaf is an independently drawn random element of $\mathbf{2}$ with the probability of 1 being p (which also makes it manifest that—as suggested by the notation—p_d only depends on p and d, not on a). Since the $k + 1$ input subformulas of any gate are independent, this gives a recurrence for p_d:

$$p_0 = p,$$
$$p_{d+1} = f(p_d),$$

where

$$f(x) = x^{k+1} + (k + 1)x^k(1 - x) = (k + 1)x^k - kx^{k+1}.$$

Clearly, f maps $[0,1]$ to $[0,1]$. In order to analyze the behaviour of p_d, we need to locate the fixed points of f in $[0,1]$, i.e., the roots of the polynomial $g(x) = f(x) - x$. The end-points 0 and 1 are roots. Moreover, the derivative

$$g'(x) = (k+1)kx^{k-1}(1-x) - 1$$

satisfies $g'(0) = g'(1) = -1 < 0$, hence there must be another root in $(0,1)$. On the other hand, Descartes's rule of signs implies that g has at most two positive roots. Thus, g has a *unique* root $\sigma_k = \sigma \in (0,1)$; g is negative on $(0,\sigma)$, and positive on $(\sigma,1)$. That is, σ is the unique fixed point of f in $(0,1)$, and we have

$$\begin{aligned} 0 < x < \sigma &\implies f(x) < x, \\ \sigma < x < 1 &\implies x < f(x). \end{aligned}$$

Consequently, for any p, the sequence p_d is monotone (decreasing for $0 < p < \sigma$, and increasing for $\sigma < p < 1$), and as such it has a limit, which must be a fixed point of f. Thus,

$$\begin{aligned} 0 \le p < \sigma &\implies \lim_{d\to\infty} p_d = 0, \\ \sigma < p \le 1 &\implies \lim_{d\to\infty} p_d = 1. \end{aligned}$$

We claim that σ is irrational, hence $p \neq \sigma$: since σ is a root of $1 - (k+1)x^{k-1} + kx^k$, σ^{-1} is a root of the monic polynomial $h(x) = x^k - (k+1)x + k$, and as such, it is an algebraic integer. Thus, if it were rational, it would be an actual integer; however, it is easy to see that $h(x) > 0$ for all $x \ge 2$, hence $1 < \sigma^{-1} < 2$.

Next, we need to analyze the rate of convergence of p_d. In the vicinity of σ, we have

$$f(x) = x + (x-\sigma)g'(\sigma) + O\big((x-\sigma)^2\big),$$

where $g'(\sigma) > 0$: since $g(0) = g(\sigma) = g(1) = 0$, g' has a root in $(0,\sigma)$ and another in $(\sigma,1)$, while it has at most two positive roots altogether by the rule of signs, hence $g'(\sigma) \neq 0$. We cannot have $g'(\sigma) < 0$ as g is negative on $(0,\sigma)$ and positive on $(\sigma,1)$.

Thus, we may fix constants $\varepsilon_0 > 0$ and $\gamma_0 > 1$ such that

$$|x - \sigma| \le \varepsilon_0 \implies |f(x) - \sigma| \ge \gamma_0 \, |x - \sigma| \,,$$

hence

$$|p_d - \sigma| \ge \min\big\{\varepsilon_0, \gamma_0^d \, |p - \sigma|\big\}.$$

By Roth's theorem, σ has irrationality measure 2, i.e., for any $\delta > 0$, all but finitely many pairs of integers $a, b > 0$ satisfy

$$\left|\frac{a}{b} - \sigma\right| \ge b^{-(2+\delta)}.$$

(The weaker theorem of Liouville, bounding the irrationality measure by $k-1$, would also work for our purposes.) Applying this to $p = w/n$, we obtain

$$\log |p - \sigma|^{-1} \le (2 + o(1)) \log n,$$

thus there exists a constant c_0 such that

$$d \geq c_0 \log n \implies |p_d - \sigma| \geq \varepsilon_0 \tag{9}$$

for any sufficiently large n and $p = w/n$.

In the vicinity of the end-points, we have

$$f(x) = (k+1)x^k + O(x^{k+1}), \tag{10}$$

$$f(1-x) = 1 - \binom{k+1}{2} x^2 + O(x^3). \tag{11}$$

Thus, we may fix $\varepsilon_1 > 0$ and $\gamma_1 > 0$ such that

$$0 \leq x \leq \varepsilon_1 \implies f(x) \leq \gamma_1 x^2 \text{ and } f(1-x) \geq 1 - \gamma_1 x^2,$$

hence

$$0 \leq p_d \leq \varepsilon_1 \implies p_{d+d'} \leq \gamma_1^{-1}(\gamma_1 p)^{2^{d'}}, \tag{12}$$

$$1 - \varepsilon_1 \leq p_d \leq 1 \implies p_{d+d'} \geq 1 - \gamma_1^{-1}\big(\gamma_1(1-p)\big)^{2^{d'}}. \tag{13}$$

We may assume $\gamma_1 \varepsilon_1 < 1$. There is a constant d_0 such that

$$|p_d - \sigma| \geq \varepsilon_0 \implies p_{d+d_0} \leq \varepsilon_1 \text{ or } 1 - p_{d+d_0} \leq \varepsilon_1. \tag{14}$$

Putting (9), (12), (13) and (14) together, there is a constant c_1 such that

$$d \geq c_1(\log n + \log e) \implies p_d \leq 2^{-n-e} \text{ or } 1 - p_d \leq 2^{-n-e}$$

for sufficiently large n; that is, for any assignment $a \in \mathbf{2}^n$ of weight w,

$$d \geq c_1(\log n + \log e) \implies \begin{cases} \Pr[F_{n,d}(a) \neq 0] \leq 2^{-n-e}, & w < \sigma n, \\ \Pr[F_{n,d}(a) \neq 1] \leq 2^{-n-e}, & w > \sigma n. \end{cases}$$

Using the union bound over all assignments $a \in \mathbf{2}^n$, we obtain

$$d \geq c_1(\log n + \log e) \implies \Pr[F_{n,d} \equiv \theta^n_{\lceil \sigma n \rceil}] \geq 1 - 2^{-e}.$$

Now, given n and t such that $\sigma n < t \leq n$, put $N = \lfloor \sigma^{-1} t \rfloor \geq n$. Then $\lceil \sigma N \rceil = t$, thus

$$\theta^n_t(\vec{x}) \equiv \theta^N_{\lceil \sigma N \rceil}(\vec{x}, 0, \dots, 0) \equiv \theta^N_{\lceil \sigma N \rceil}\big(\vec{x}, \textstyle\bigwedge_i x_i, \dots, \bigwedge_i x_i\big),$$

and consequently

$$d \geq c_2(\log n + \log e) \implies \Pr\big[F_{N,d}\big(\vec{x}, \textstyle\bigwedge_i x_i, \dots, \bigwedge_i x_i\big) \equiv \theta^n_t\big] \geq 1 - 2^{-e}$$

for some constant c_2. We thus define

$$T_{n,t,e,r} = F_{\lfloor \sigma^{-1} t \rfloor, c_2(\log n + \log e)}\big(\vec{x}, \textstyle\bigwedge_i x_i, \dots, \bigwedge_i x_i\big),$$

where r is the sequence of random coin tosses that determines the leaves of the formula. Notice that $\wedge \in \mathrm{MPT}_0^k = [\theta_k^{k+1}]$, hence $\bigwedge_i x_i$ can be easily expressed by a logspace-computable sequence of $O(\log n)$-depth $\{\theta_k^{k+1}\}$-formulas. It is also straightforward to compute $F_{N,d}$ in logspace given N, d, and the random sequence r.

Finally, to compute $N = \lfloor \sigma^{-1} t \rfloor$, the algebraic number σ^{-1} can be approximated by a $\mathbf{TC}^0$ function (hence a logspace function) by [14], but since we actually need it only with $O(\log n)$ bits of precision, even a brute force approach is sufficient: we can evaluate the polynomial $x^k - (k+1)t^{k-1}x + kt^k$ (which is $t^k h(xt^{-1})$) in parallel for each $x = n, n+1, \dots, 2t$, and let N be the argument where it switches sign from negative to positive. Thus, $T_{n,t,e,r}$ is computable by a logspace function. □

Remark 4.11 The expansion (11) easily implies that for large k, the constant σ_k from Theorem 4.10 is $1 - 2k^{-2} + O(k^{-3})$.

Theorem 4.12 *If $[B] \supseteq \mathrm{PT}_\alpha^k$ for some $k \in \mathbb{N}$ and $\alpha \in \mathbf{2}$, then B-CMP is $\mathbf{\Theta}_2^{\mathbf{P}}$-complete under randomized logspace reductions. More precisely, for any language $L \in \mathbf{\Theta}_2^{\mathbf{P}}$, there exists a constant c and a logspace function $R(w, e, r)$ with e given in unary such that for all strings w of length n, and for all e,*

$$w \in L \implies \Pr_{|r|=(n+e)^c}[R(w,e,r) \in B\text{-CMP}] = 1, \tag{15}$$

$$w \notin L \implies \Pr_{|r|=(n+e)^c}[R(w,e,r) \in B\text{-CMP}] \le 2^{-e}. \tag{16}$$

Proof: We will assume $[B] \supseteq \mathrm{T}_0^k$: we may pass from T_0^k to PT_0^k in the same way as in the proof of Theorem 4.8, and the case of $[B] \supseteq \mathrm{PT}_1^k$ is dual. Without loss of generality, $k \ge 3$.

We use the reduction from Lemma 3.11 and Theorem 3.12 with the following two modifications:

(i) We express the formulas $x_i \wedge \varphi_i$ by B-formulas.

(ii) In place of the threshold function θ_t^m, we use the randomly generated formula $T_{m,t,e,r}$ from Theorem 4.10.

As for (i), recall that the formulas φ_i supplied by Lemma 3.10 are CNFs, hence they may be arranged to have depth $O(\log n)$. We may assume them to be written in the $\{\wedge, \neg\}$ basis. We then write $x_i \wedge \varphi_i$ in the basis $\{\wedge, \not\to\} \subseteq \mathrm{T}_0^\infty$ by replacing each subformula $\neg\psi$ with $x_i \wedge \neg\psi$. Since $\mathrm{T}_0^\infty \subseteq [B]$, we may rewrite the formulas as B-formulas by Lemma 2.3.

Concerning (ii), notice first that in Lemma 3.11, we may assume n to be sufficiently large, and the parameters we take are $m \approx n^2$, $m - t \approx n$, thus $t > \sigma_k m$, justifying the use of Theorem 4.10. We again rewrite the formulas as B-formulas using Lemma 2.3. Crucially, we have to use the same $T_{m,t,e,r}$ formula for constructing both f_{even} and f_{odd}.

It is clear from the construction that (16) holds, but we need more work to establish (15), as it is not obvious that it holds with no error. If $w \in L$, let $\langle \varphi_i : i < 2n \rangle$ and j be as in Theorem 3.12, so that j is even, and put $s = j/2$. Then in the definition of both $f_{\mathrm{even}} = f_{\varphi_0,\varphi_2,\dots,\varphi_{2n-2}}$ and $f_{\mathrm{odd}} = f_{\varphi_1,\varphi_3,\dots\varphi_{2n-1}}$, the first s of the φ_i formulas are satisfiable, and the rest are unsatisfiable.

Going back to Lemma 3.11, let us abbreviate by $T(x_0,\dots,x_{m-1})$ the formula $T_{m,t,e,r}$ we use in place of θ^m_t. Then

$$f_{\text{even}} \equiv T(x_0 \wedge \varphi_0, x_1 \wedge \varphi_2, \dots, x_{s-1} \wedge \varphi_{2s-2}, \underbrace{0,\dots,0}_{n-s}, x_n, \dots, x_{m-1})$$

and

$$[\nrightarrow, f_{\text{even}}] = [\nrightarrow, T(x_0,\dots,x_{s-1},0,\dots,0,x_n,\dots,x_{m-1})]$$

by the argument in Lemma 3.11 (avoiding renumbering of variables or permuting the arguments of T). Since the same T was also used to construct f_{odd}, we obtain likewise

$$[\nrightarrow, f_{\text{odd}}] = [\nrightarrow, T(x_0,\dots,x_{s-1},0,\dots,0,x_n,\dots,x_{m-1})],$$

hence $[\nrightarrow, f_{\text{even}}] = [\nrightarrow, f_{\text{odd}}]$. □

Notice that if the language L we are reducing to B-CMP is in **BH**, the number n in the proof of Theorem 3.12 can be taken as constant, hence also m and t in Lemma 3.11 are constant, and we may just fix a representation of θ^m_t by a B-formula in advance, avoiding the complicated randomized construction from Theorem 4.10.

Corollary 4.13 *If $[B] \supseteq \mathrm{PT}^k_\alpha$ for some $k \in \mathbb{N}$ and $\alpha \in \mathbf{2}$, then B-CMP is **BH**-hard.* □

Let us summarize the results of Section 4 (see Fig. 2):

Corollary 4.14 *Let $B \subseteq \mathrm{Op}$ be finite.*

(i) *If $B \subseteq \mathrm{MT}^\infty_0$, MT^∞_1, $\bigwedge$, $\bigvee$, A, or DM, then B-CMP $\in$ **P**.*

(ii) *If $[B]$ is T^∞_0, PT^∞_0, T^∞_1, PT^∞_1, D, or DP, then B-CMP is **coDP**-complete.*

(iii) *If $[B] \supseteq \mathrm{PT}^k_\alpha$ for some $k \in \mathbb{N}$ and $\alpha \in \mathbf{2}$, then B-CMP is $\mathbf{\Theta}^{\mathbf{P}}_2$-complete under randomized reductions, and even under deterministic reductions if $[B] \supseteq \mathrm{P}$. Also, B-CMP is **BH**-hard.*

(iv) *If $\mathrm{MPT}^k_\alpha \subseteq [B] \subseteq \mathrm{M}$ for some $k \in \mathbb{N}$ and $\alpha \in \mathbf{2}$, then B-CMP is in $\mathbf{\Theta}^{\mathbf{P}}_2$. If $[B] \supseteq \mathrm{MPT}^2_\alpha$, B-CMP is **coDP**-hard.* □

While randomized reductions are a nuisance, the real problem is the last item of Corollary 4.14, where the upper and lower bounds (if any) are far from each other. Notice that since any non-constant monotone function is both 0- and 1-preserving, M-CMP is logspace-equivalent to MP-CMP, and (using also duality) for any $k \geq 1$, the problems MT^k_0-CMP, MPT^k_0-CMP, MT^k_1-CMP, and MPT^k_1-CMP are logspace-equivalent.

Problem 4.15 *What is the complexity of B-CMP for $\mathrm{MPT}^k_\alpha \subseteq [B] \subseteq \mathrm{M}$?*

Our first hunch is that all these problems should be $\mathbf{\Theta}^{\mathbf{P}}_2$-complete just like their non-monotone versions, but on second thought, it is conceivable that, for example, we can learn some properties of monotone functions by a randomized process such as in the proof of Theorem 4.10, hence the expected answer is not as clear-cut.

We note that Böhler and Schnoor [7] left open a similar problem about the complexity of certain cases of B-CMP_C.

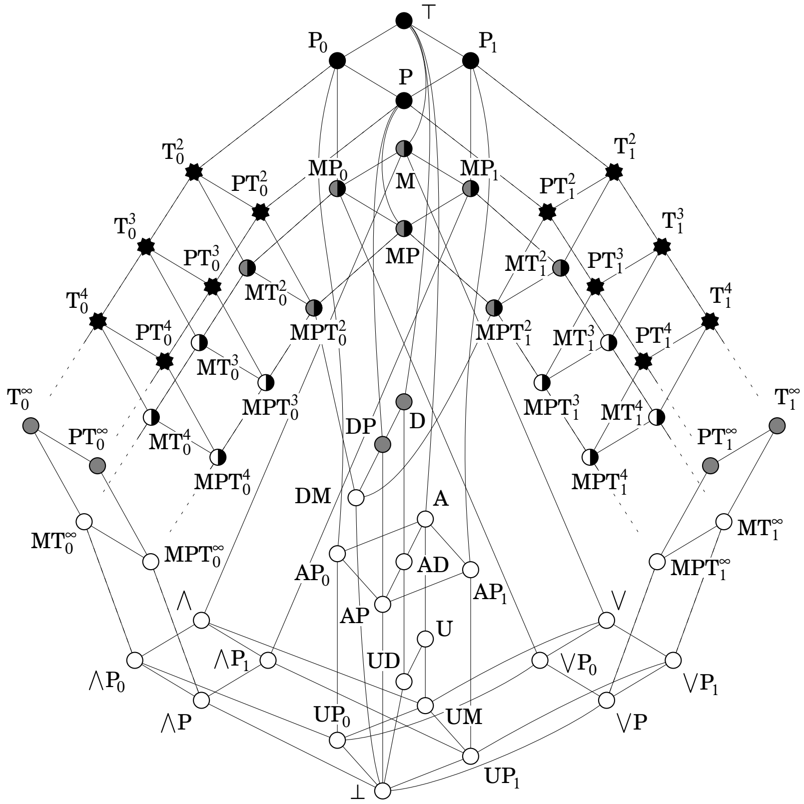

Figure 2: The complexity of B-CMP (○: in **P**; ●: **coDP**-complete; ●: $\mathbf{\Theta}_2^{\mathbf{P}}$-complete; ✸: $\mathbf{\Theta}_2^{\mathbf{P}}$-complete under randomized reductions; ◑: in $\mathbf{\Theta}_2^{\mathbf{P}}$; ◑: in $\mathbf{\Theta}_2^{\mathbf{P}}$, **coDP**-hard).

5 Conclusion

We have undertaken a thorough investigation of the complexity of the Boolean clone membership problem CMP and its variants. Most importantly, we proved that CMP is $\mathbf{\Theta}_2^{\mathbf{P}}$-complete, and in particular, strictly harder than any of the fixed-clone problems CMP_C or CMP^f, barring collapse of the polynomial hierarchy.

Moreover, we obtained a representative (even if incomplete) picture of how the complexity depends on the basis B of gates allowed in the input. As expected, it shows a major dividing line depending on whether $[B]$ has finitely many subclones: in the latter case the complexity drops down inside the Boolean hierarchy—in fact, to **coDP**. However, there seems to be also a more subtle dividing line based on whether B consists of monotone functions only: in the finite subclone case, this makes the complexity of B-CMP go further down to **P** (though this also happens for some non-monotone cases, namely when B consists of affine functions); in the infinite subclone case, it separates the area of more-or-less $\mathbf{\Theta}_2^{\mathbf{P}}$-complete instances of B-CMP from a terra incognita.

Acknowledgements

I would like to thank the anonymous referees for helpful comments.

The research was supported by grant 19-05497S of GA ČR. The Institute of Mathematics of the Czech Academy of Sciences is supported by RVO: 67985840.

References

[1] David A. Mix Barrington, Neil Immerman, and Howard Straubing, *On uniformity within NC^1*, Journal of Computer and System Sciences 41 (1990), no. 3, pp. 274–306.

[2] Michael Bauland, Martin Mundhenk, Thomas Schneider, Henning Schnoor, Ilka Schnoor, and Heribert Vollmer, *The tractability of model checking for LTL: The good, the bad, and the ugly fragments*, ACM Transactions on Computational Logic 12 (2011), no. 2, article no. 13, 28 pp.

[3] Mike Behrisch, *Clones with nullary operations*, Electronic Notes in Theoretical Computer Science 303 (2014), pp. 3–35.

[4] Clifford Bergman, David Juedes, and Giora Slutzki, *Computational complexity of term-equivalence*, International Journal of Algebra and Computation 9 (1999), no. 1, pp. 113–128.

[5] Clifford Bergman and Giora Slutzki, *Complexity of some problems concerning varieties and quasi-varieties of algebras*, SIAM Journal on Computing 30 (2000), no. 2, pp. 359–382.

[6] Olaf Beyersdorff, Arne Meier, Michael Thomas, and Heribert Vollmer, *The complexity of reasoning for fragments of default logic*, Journal of Logic and Computation 22 (2012), no. 3, pp. 587–604.

[7] Elmar Böhler and Henning Schnoor, *The complexity of the descriptiveness of Boolean circuits over different sets of gates*, Theory of Computing Systems 41 (2007), no. 4, pp. 753–777.

[8] Andrei A. Bulatov, *A dichotomy theorem for nonuniform CSPs*, in: Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science, 2017, pp. 319–330.

[9] Samuel R. Buss, *The Boolean formula value problem is in ALOGTIME*, in: Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, pp. 123–131.

[10] Samuel R. Buss and Louise Hay, *On truth-table reducibility to* SAT, Information and Computation 91 (1991), no. 1, pp. 86–102.

[11] Gil Cohen, Ivan Bjerre Damgård, Yuval Ishai, Jonas Kölker, Peter Bro Miltersen, Ran Raz, and Ron D. Rothblum, *Efficient multiparty protocols via log-depth threshold formulae*, in: Advances in Cryptology – CRYPTO 2013. Proceedings, Part II (R. Canetti and J. A. Garay, eds.), Lecture Notes in Computer Science vol. 8043, Springer, 2013, pp. 185–202, full version available at `http://www.wisdom.weizmann.ac.il/~ranraz/publications/Pmajority_mpc-1.pdf`.

[12] Nadia Creignou, Johannes Schmidt, and Michael Thomas, *Complexity classifications for propositional abduction in Post's framework*, Journal of Logic and Computation 22 (2012), no. 5, pp. 1145–1170.

[13] Arvind Gupta and Sanjeev Mahajan, *Using amplification to compute majority with small majority gates*, Computational Complexity 6 (1996), no. 1, pp. 46–63.

[14] Emil Jeřábek, *Root finding with threshold circuits*, Theoretical Computer Science 462 (2012), pp. 59–69.

[15] Dexter Kozen, *Lower bounds for natural proof systems*, in: Proceedings of the 18th Annual Symposium on Foundations of Computer Science, 1977, pp. 254–266.

[16] Marcin Kozik, *A finite set of functions with an EXPTIME-complete composition problem*, Theoretical Computer Science 407 (2008), pp. 330–341.

[17] Richard E. Ladner, *The circuit value problem is log space complete for $\mathcal{P}$*, ACM SIGACT News 7 (1975), no. 1, pp. 18–20.

[18] Dietlinde Lau, *Function algebras on finite sets: A basic course on many-valued logic and clone theory*, Springer, New York, 2006.

[19] Harry R. Lewis, *Satisfiability problems for propositional calculi*, Mathematical Systems Theory 13 (1979), pp. 45–53.

[20] Thomas Lukasiewicz and Enrico Malizia, *A novel characterization of the complexity class Θ^{P}_k based on counting and comparison*, Theoretical Computer Science 694 (2017), pp. 21–33.

[21] Dragan Mašulović, GenClo *and* TermEquiv *are* EXPTIME*-complete*, International Journal of Algebra and Computation 18 (2008), no. 5, pp. 901–909.

[22] Arne Meier and Thomas Schneider, *Generalized satisfiability for the description logic $\mathcal{ALC}$*, Theoretical Computer Science 505 (2013), pp. 55–73.

[23] Emil L. Post, *The two-valued iterative systems of mathematical logic*, Annals of Mathematics Studies no. 5, Princeton University Press, Princeton, 1941.

[24] Steffen Reith, *Generalized satisfiability problems*, Ph.D. thesis, Julius-Maximilians-Universität, Würzburg, 2001.

[25] Walter L. Ruzzo, *On uniform circuit complexity*, Journal of Computer and System Sciences 22 (1981), no. 3, pp. 365–383.

[26] Donald Sannella and Andrzej Tarlecki, *On observational equivalence and algebraic specification*, Journal of Computer and System Sciences 34 (1987), no. 2–3, pp. 150–178.

[27] Thomas J. Schaefer, *The complexity of satisfiability problems*, in: Proceedings of the 10th Annual ACM Symposium on Theory of Computing, 1978, pp. 216–226.

[28] Ágnes Szendrei, *Clones in universal algebra*, Séminaire de mathématiques supérieurs vol. 99, Université de Montréal, 1986.

[29] Leslie G. Valiant, *Short monotone formulae for the majority function*, Journal of Algorithms 5 (1984), no. 3, pp. 363–366.

[30] Heribert Vollmer, *The complexity of deciding if a Boolean function can be computed by circuits over a restricted basis*, Theory of Computing Systems 44 (2009), no. 1, pp. 82–90.

[31] Klaus W. Wagner, *Bounded query classes*, SIAM Journal on Computing 19 (1990), no. 5, pp. 833–846.

[32] Ingo Wegener, *The complexity of Boolean functions*, Teubner, Stuttgart, 1987.

[33] Dmitriy Zhuk, *A proof of CSP dichotomy conjecture*, in: Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science, 2017, pp. 331–342.